UNIVERSIDAD NACIONAL DE INGENIERÍA

FACULTAD DE INGENIERÍA INDUSTRIAL Y DE SISTEMAS
SECCIÓN DE POSGRADO

"UNA HERRAMIENTA DE ANÁLISIS ESTADÍSTICO PARA LA INVESTIGACIÓN AGRÍCOLA"

TESIS

PARA OPTAR EL GRADO ACADÉMICO DE
MAESTRO EN CIENCIAS
CON MENCIÓN EN INGENIERÍA DE SISTEMAS

PRESENTADO POR:
FELIPE DE MENDIBURU DELGADO

LIMA, PERÚ
2009

Dedicatoria:

A mis padres Leonidas y Carmen

mi esposa Ruby

y mis hijos Carmen,

Luis, Mariela y Renzo.

AGRADECIMIENTO

- Al Centro Internacional de la Papa y al Dr. Reinhard Simon, Jefe de la Unidad de Informática para la Investigación, por el apoyo en el desarrollo de la presente investigación.
- A los profesores: Mg. Zalatiel Carranza, Mg. Jorge Guzmán y Mg. Miguel Sierra, por su valiosa ayuda y sugerencia en el desarrollo y redacción del plan de tesis y el trabajo final.
- A los evaluadores de la librería, los doctores:
  Banerjee, Partha. ICRISAT- India
  Ding, Jane. Chinese Academy of Science of the Institute of Botany - China
  Duleep, Samuel. Ind. Inst. of Hort Research, Bangalore - India
  Duyme, Florent. ARVALIS - Institut du Végétal - Francia
  Escriou, Hervé. ITB – Institut technique de la betterave - Francia
  Klein-Gebbinck, Henry. Gobierno de Canadá
  Kozak, Marcin. Warsaw University of Life Sciences - Polonia
  Marcelo Luiz de Laia. Universidade do Estado de Santa Catarina - Brasil
  Salgado Enríquez, Gustavo. Edumetrics Cía. Ltda. - Ecuador
  Streibig, Jens Carl. University of Copenhagen - Dinamarca
  Wyseure, Guido. Department Earth and Environmental Sciences, K.U.Leuven - Bélgica
- A mi hija María del Carmen de Mendiburu por las sugerencias en la redacción.
- A todas las personas que me enviaron la encuesta de satisfacción para la evaluación de la librería agricolae y me alentaron durante el desarrollo del presente trabajo de tesis.

CONTENIDO

RESUMEN

El diseño de experimentos en el campo agrícola y el tratamiento estadístico de los datos provenientes de estos experimentos constituyen la base principal para el análisis y uso de los resultados de una investigación agrícola. En nuestro país las instituciones como la Universidad Nacional Agraria La Molina, el INIA (Instituto Nacional de Innovación Agraria), el CIP (Centro Internacional de la Papa) y otras Universidades del país realizan investigación agrícola para el tratamiento de los cultivos. La construcción de diseños y análisis con programas licenciados, tales como SAS, SPSS y MINITAB son insuficientes. La presente investigación trata el problema con la construcción de funciones para los diseños y análisis estadistico de experimentos, análisis de estabilidad, comparaciones múltiples de tratamientos, análisis de consenso en dendrogramas y otros, conformando una librería de nombre "AGRICOLAE" en el lenguaje R, la cual es sometida a un estudio de calidad según el modelo McCall. El lenguaje R es libre en sistemas como Windows, Linux y MAC, y cuenta con un lenguaje de programación funcional, disponible en INTERNET (http://www.r-project.org). La programación de las funciones de "agricolae" se realizó completamente en R y la evaluacion de la calidad, mediante un programa del mismo sistema, el cual realiza la validación de la sintaxis, instalación, ayuda y documentación, y se complementó el estudio con una encuesta de satisfacción con ítems en la escala de likert (1-5), la cual fue difundida por correo electrónico e INTERNET. La encuesta fue respondida por 37 usuarios del Perú y 11 de otros países, con resultados favorables. El factor de calidad de 0.8 en la escala (0-1), el índice alfa de Cronbach de 0.88, la similitud de los resultados con SAS, SPSS y MINITAB, y la opinión de expertos son indicadores de la calidad del software. La librería agricolae fue aceptada por el Proyecto R. Actualmente la librería se encuentra en INTERNET:

http://cran.at.r-project.org/web/packages/agricolae/index.html

DESCRIPTORES TEMATICOS

AGRICOLAE
Alfa de Cronbach
AMMI
Consenso dendrograma
Diseño afa
Diseño experimental
Escala Likert
Modelo McCall
R-Project
Waller-Duncan

INTRODUCCIÓN

Los centros de investigación agrícola utilizan con frecuencia los diseños de experimentos aplicados en campo e invernadero para estudiar el efecto de tratamientos como: fertilizantes, insecticidas, sistemas de riego, enmiendas, aplicación de químicos, etc., y estudian el comportamiento del cultivo en: rendimiento, características en la planta o resistencia a una enfermedad o fenómeno climático. El Centro Internacional de la Papa (CIP) realiza investigaciones científicas en cultivos de papa, camote y otros cultivos andinos, y dispone de estaciones experimentales en muchos lugares del Perú y el mundo. Los científicos del CIP requieren frecuentemente de herramientas estadísticas para el diseño de experimentos y análisis.

En la presente investigación se plantea desarrollar una librería en R (agricolae) bajo la licencia "General Public License" (GPL), de funciones estadísticas aplicadas a la investigación agrícola no disponible y/o disponible en programas comerciales como SAS, SPSS y MINITAB bajo Windows, Linux y MAC, y que cumplan los estándares de calidad de software.

Las funciones seleccionadas para la librería agricolae son los procedimientos estadísticos más utilizados por los científicos del CIP, tales como diseños de experimentos, comparaciones múltiples paramétricas y no-paramétricas, prueba de estabilidad para genotipos como el AMMI, Shukla's Stability Variance And Kang's, (Kang, 1997), y el consenso de dendrogramas para biodiversidad. La calidad de la librería será evaluada mediante el modelo McCall a través de una encuesta de satisfacción con ítems en la escala de Likert, así como la opinión de expertos. El chequeo del software se realizará con R y será publicado en el proyecto R. Algunas demostraciones del uso de la librería serán presentadas con datos de investigación, las cuales serán incorporadas en la librería como una base de datos, y los resultados serán comparados con programas como SAS, SPSS y MINITAB.

CAPITULO I

PLANEAMIENTO DE LA INVESTIGACIÓN

1.1 DIAGNOSTICO Y ENUNCIADO DEL PROBLEMA

El tratamiento estadístico de experimentos agronómicos es frecuente en instituciones de investigación agrícola. En el Perú existen centros de investigación agrícola como el Centro Internacional de la Papa (CIP), el Centro de Innovación Nacional Agraria (INIA), la Universidad Nacional Agraria La-Molina y otras Universidades que tienen especialidades agropecuarias. El CIP es un centro de investigación sin fines de lucro, orientado mayormente al cultivo de la papa y otras raíces y tubérculos, su sede se encuentra en Lima (La Molina), cuenta con una estación experimental de 10 hectáreas y estaciones en Huancayo, San Ramón y Quito (Ecuador), así como en regiones de Asia y África, y realiza intensa investigación en estos 35 años de funcionamiento. Sus registros de experimentación indican que el CIP en estos últimos 4 años ha realizado más de 300 experimentos, un 85% en campo y un 15% en invernadero y otros medios controlados. Los investigadores del CIP han utilizado varios programas, actualmente utiliza el programa SAS (Statistical Analysis System) que se utiliza desde 1986. El mayor problema radica en el planeamiento de experimentos con los diseños experimentales, el SAS tiene el procedimiento PROC PLAN, el cual que permite realizar algunos diseños básicos, pero no los diseños de bloques incompletos, los cuales son realizados manualmente. En los últimos 323 experimentos, 46 fueron diseños alfa y 27 en latices. Los mayores usuarios son los mejoradores de plantas; el investigador debe probar muchos genotipos o clones, los cuales sobrepasan 200 o más, y debe ser planeados en bloques incompletos. Otras herramientas que no están disponibles en SAS y programas afines son los métodos de cruzamiento en

genética, tales como los diseños Carolina, cruzas línea por probador; en estabilidad el AMMI también utilizado por los genetistas y el consenso en dendrogramas para biodiversidad y genetica con marcadores moleculares. Algunos programas en genética realizan el consenso sólo para marcadores moleculares. El CIP empezó a utilizar el programa R en el año 2003 y la facilidad para programar permitió un uso más frecuente. Las metodologías de análisis en investigación agrícola publicadas en revistas no son actualizadas en los programas con la misma rapidez que se hace para los casos industrial y comercial, y es necesario contar con alguna herramienta que facilite la programación de las metodologías recientemente publicadas. Hoy en día, la presencia de software libre "open source" ha facilitado la programación en este campo. El CIP como centro de investigación, ha considerado conveniente dar prioridad al desarrollo de procedimientos por software libre. Los investigadores en áreas de mejoramiento genético de plantas, biodiversidad, entomología, patología, virológica y conservación de recursos se ven beneficiados por el uso del programa R, ya que les facilita usar nuevos algoritmos para el tratamiento de sus datos; entonces el problema es construir una librería de funciones que facilite el procesamiento de datos de la investigación agrícola.

1.2 DEFINICIÓN DEL PROBLEMA DE INVESTIGACIÓN

Algunos procedimientos, especialmente de diseños experimentales no se encuentran disponibles, y los costos de programas en computadora para fines de investigación son muy altos. Los programas, tales como SAS, SPSS, MINITAB y otros, los cuales podrían ayudar a resolver los problemas de investigación, son difícilmente adquiridos por los centros de Investigación y Universidades Nacionales por ser instituciones sin fines de lucro. Las metodologías recientes sobre investigación, publicadas en revistas, no son actualizadas aún en los programas de uso comercial y es necesario disponer de algún programa que facilite la aplicación de estas metodologías. Hoy en día la presencia de software libre "open source" ha facilitado la programación en este campo. El CIP, como centro de investigación que realiza considerable investigación, ha considerado conveniente dar prioridad al uso del programa R, el cual permite construir funciones estadísticas para procesar sus datos. Tanto los investigadores en áreas de mejoramiento genético de plantas como especialistas afines se ven beneficiados porque este programa les facilita usar

nuevos algoritmos; ahora el problema es construir una librería de funciones que facilite el procesamiento de datos provenientes de la investigación agrícola.

1.3 DEFINICIÓN DE LOS OBJETIVOS

1.3.1 OBJETIVO GENERAL

Construir una librería para el programa R, de nombre "AGRICOLAE", con funciones estadísticas para la investigación agrícola, en plataformas como Windows, Linux y MAC, y que cumpla los estándares de calidad de software.

1.3.2 OBJETIVOS ESPECÍFICOS

- Mostrar el procedimiento para la construcción de la librería "agricolae".
- Construir funciones para el planeamiento de los diseños: completamente aleatorio, bloques, cuadrado latino, greco latino, bloque incompleto balanceado y diseños alfa.
- Construir funciones de análisis estadístico para los modelos de diseños de bloques incompletos y las comparaciones múltiples de tratamientos, así como procedimientos para el análisis de estabilidad en el mejoramiento genético de plantas.
- Incorporar datos experimentales del Centro Internacional de la Papa en la librería para la demostración, prueba y uso de agricolae.

1.4 HIPÓTESIS DE LA INVESTIGACIÓN

1.4.1 HIPÓTESIS GENERAL

La librería "agricolae" satisface las condiciones de un software de calidad el cual cumple los requisitos planteados en el modelo McCall.

1.4.2 HIPÓTESIS ESPECÍFICAS

- La librería "agricolae" es de libre disponibilidad bajo licencia GPL y funciona perfectamente en el sistema R.

- La librería "agricolae" produce resultados válidos comparables a los resultados publicados en libros, y revistas científicas y a programas comerciales, tales como SAS, SPSS y MINITAB.

1.5 JUSTIFICACIÓN Y DELIMITACIÓN DE LA INVESTIGACIÓN

1.5.1 IMPORTANCIA DEL TEMA

Nuestro país en cierta medida es agrícola y los gobiernos de turno están dando énfasis al desarrollo de su producción, impulsando el área a través de convenios y tratados con otros países. En poco tiempo los agricultores deberán enfrentar retos en cuanto a productividad, calidad y mejor condición de vida de los pueblos alejados que viven mayormente de la agricultura. Las Universidades y Centros de investigación, tales como el CIP y el INIA, en donde se realiza la investigación agrícola en nuestro país, deben disponer de herramientas que les permita resolver los problemas y satisfacer las exigencias de la sociedad en alimentación y salud pública. Nuevas tecnologías deberán aplicarse y el uso de herramientas libres de computación en el campo de la estadística podría facilitar esta tarea; por lo tanto, ésta será una alternativa para fomentar la investigación y la enseñanza en las universidades.

1.5.2 JUSTIFICACIÓN

La presente investigación se sustenta en la implementación de funciones estadísticas aplicadas a la investigación agrícola y que no estén al alcance de los investigadores, tales como diseños experimentales, métodos de estabilidad, consenso de dendrogramas, y pruebas comparativas paramétricas y no paramétricas. Estas funciones solamente pueden ser reconocidas trabajando directamente con los investigadores, específicamente con los investigadores del CIP, lugar donde se desarrollará la presente investigación. El Centro cuenta con más de 50 investigadores nacionales y extranjeros que realizan esta actividad; las áreas de su funcionamiento son el mejoramiento genético, el control de plagas y enfermedades, la identificación de material genético con resistencia a plagas, virus y condiciones de ambiente y la conservación y propagación de plantas. Los experimentos son conducidos en ambientes controlados, tales como invernaderos y

laboratorios o en campo experimental y de agricultores. El alcance de herramientas de programación libre, tales como el programa R, que permitan el desarrollo de funciones estadísticas e integrarlas en un paquete, facilitarian la actividad de análisis de los investigadores; naturalmente asociadas a un amplio conocimiento de la metodología estadística para poder construir, usar e interpretar los resultados que generen estas funciones.
La librería debe contemplar ciertas áreas:

- Diseño de experimentos básicos y diseños de bloques incompletos.
- Análisis de estos experimentos hasta las pruebas comparativas de tratamientos.
- Comparación de tratamientos para los casos no paramétricos.
- Análisis y diseño genético como estabilidad y cruzamiento genético entre plantas.

Asimismo, la librería deberá estar correctamente documentado, en un manual de referencia y presentar ayudas con ejemplos demostrativos.

1.5.3 DELIMITACIÓN

La presente investigación se delimita por la creación de una librería de funciones estadísticas para el uso de la investigación agrícola, identificada con el nombre "agricolae" la cual dispondrá de procedimientos para el análisis de datos provenientes de la investigación agrícola, tales como el diseño de experimentos, diseños genéticos, estabilidad de genotipos y biodiversidad. Los programas estarán escritos en el lenguaje R bajo la licencia GPL, que es "open source", y los datos provendrán de investigaciones recientes del CIP que hayan sido publicadas en los últimos años. Dicha información será utilizada para la demostración de la librería "agricolae". El uso de la librería será libre y no podrá ser utilizada con fines lucrativos.

CAPITULO II

MARCO TEÓRICO

2.1 ANTECEDENTES

Sobre los diseños experimentales y sus principios básicos se tiene referencias desde 1923 por R. A. Fisher y sus asociados en "Rothamsted Experimental Station", en Inglaterra. La aplicación de estos principios sólo fue bajo formas y arreglos aleatorios publicados en libros. Una de las principales publicaciones fue de Cochran y Cox, posteriormente editada en varias versiones (Cochran, 1992). En todo este tiempo se publicaron muchos libros sobre diseños de experimentos (Box, 2005). Con el uso de las computadoras se comenzó a programar los diseños de experimentos especialmente en el área industrial, no tanto así para el área agrícola en donde fue su raíz. Actualmente existen programas comerciales de alto costo para estos fines pero no están actualizados de acuerdo al adelanto científico. Las universidades y los centros de investigación necesitan de estas herramientas. CIP e INIA en el Perú son centros de investigación agrícola y usan con frecuencia la planificación de experimentos. La programación libre "Open Source" se inició en 1984 con el proyecto GNU (http://www.gnu.org), el cual presenta la Licencia Pública General de Software para la distribución libre. El objetivo del proyecto fue desarrollar un sistema operativo tipo Unix completo pero de libre uso; bajo esta licencia se encuentra la licencia GPL y se están desarrollando todos los programas de fuente libre. Uno de los programas principales en estadística es R del proyecto R-Project (http://www.r-project.org); es libre bajo los términos de Free Software Foundation's GNU General Public License, el cual tuvo sus inicios en 1997 en el Departamento de Estadística de la Universidad de AUCKLAND en Australia, y a partir del 2000, en el

Departamento de Estadística y Matemáticas de la Universidad WU Wien en Viena, la cual distribuye vía Internet toda la información del proyecto R que comprende el software y la documentación y está enlazado con 58 Universidades en más de 34 países, formando un sistema de espejos que facilita el acceso desde cualquier parte del mundo. A este sistema organizado se le conoce como el CRAN (The Comprehensive R Archive Network) (R Development Core Team, 2008).

2.2 MARCO TEÓRICO

La presente investigación se orienta a la construcción de una librería de funciones aplicadas en la investigación agrícola, accesible desde Internet y que funcione en el programa R. La fuente de los procedimientos, la documentación de la metodología estadística, así como los ejemplos que ilustren su uso serán integrados en la librería, la misma que debe cumplir todos los requerimientos para su instalación en Windows, Linux y Mac, tan igual como todas las librerías en R. En los siguientes puntos se tratará de describir los conceptos teóricos base para la presente investigación.

Proyecto R, programa R, software libre, GNU y la licencia GPL: El proyecto R está descrito enteramente en la página web www.R-Project.org iniciada por Robert Gentleman; estadístico matemático canadiense y Ross Dhaka, estadístico neocelandés, quienes en 1997 proponen un nuevo programa de estadística llamado R. En Febrero del 2000 se publica la versión 1.0.0 del programa y a la fecha se tiene la versión 2.9.1 de junio del 2009, correspondiente a la edición número 31. Este programa está bajo la licencia pública general GPL que permite el libre uso y disponibilidad de sus programas fuentes. El programa R es un software de programación funcional que permite la programación en un lenguaje avanzado llamado S. El lenguaje fue creado por los estadísticos y especialistas en computación, John Chambers, Rick Becker, Douglas Duna, Paul Tukey y Graham Wilkinson, en 1976 en los laboratorios Bell. Este lenguaje tuvo como base el Fortran, C y APL (lenguaje de programación aplicada). El uso de este lenguaje fue para la programación de procedimientos estadísticos. Actualmente el lenguaje S es utilizado por el programa S-Plus y una variante es el lenguaje R, el cual permite la programación, ejecución, desarrollo de procedimientos y funciones estadísticas, así también permite la conexión a datos creados por otros programas, tales como SAS, SPSS, MINITAB y otros medios utilizando la conectividad ODBC (open database Connectivity), tales como Excel, Access, Mysql.

EL lenguaje R: Es un lenguaje interpretativo como Java y no compilado, lo cual significa que los comandos escritos en el teclado son ejecutados directamente sin necesidad de construir ejecutables. Esto significa el uso en la consola. Se utiliza funciones matemáticas y estadisticas como operar con números reales, imaginarios, matrices, derivadas, integrales, probabilidades, funciones de programación como bucles, sentencias de control y generan objetos que son utilizados en la sesión. Dichos objetos pueden ser constantes, vectores, tablas, matrices, listas de objetos y funciones.

Librería agricolae: Será un paquete constituido íntegramente en R en un conjunto de funciones documentadas, integrado al programa R para dar solución a problemas en la investigación agrícola, deberá comprender los diseños de experimentales y sus análisis estadísticos correspondientes.

Un diseño experimental comprende la distribución de tratamientos de estudio en el material experimental según las características propias de cada diseño como por ejemplo: los diseños de bloques completos al azar, todos los tratamientos deben estar distribuidos aleatoriamente en cada bloque, los diseños bloques incompletos balanceados en un grupo pequeño de k < t tratamientos, aleatoriamente distribuidos en cada bloque, de tal forma que un tratamiento cualquiera se encuentre junto a cada uno de los otros en algún bloque un número de veces igual para cada tratamiento, los diseños de bloques incompletos parcialmente balanceados, similar al balanceado, pero con la restricción de que el tratamiento esté con algunos tratamientos en los pequeños bloques y una sola vez. Los diseños conocidos que se tratarán en agricolae son: los latices, alfa y cíclicos.

Los diseños latinos, tales como cuadrado y greco latino, son arreglos que satisfacen restricciones de ubicación. Así, en el cuadrado latino todos los tratamientos están distribuidos aleatoriamente por filas y columnas una sola vez. El greco latino agrega un segundo tipo de tratamientos con igual número que el primer tipo de tratamientos y debe cumplir la ortogonalidad bajo las tres restricciones. Todos los tratamientos deben estar en cada fila, en cada columna y en cada uno de los niveles del otro tipo de tratamiento. En cada diseño la función de agricolae es generar el libro de campo conteniendo la información necesaria para que el investigador pueda registrar los datos observados en las unidades experimentales.

El análisis estadístico: Se refiere al análisis de variancia de cada modelo de diseño experimental que no está disponible en R o que es complejo realizar en un modelo lineal, tales como los diseños genéticos, análisis de estabilidad genética y las

comparaciones múltiples de tratamientos como la prueba de Waller-Duncan u otra no paramétrica como Durbin para bloques incompletos. Los diseños genéticos comprenden los cruces de plantas macho y hembra que realizan los mejoradores de plantas con progenitores para producir otras plantas llamadas progenies y son sembrados juntos. A la manera de cruzar hembras con machos se derivan los diseños genéticos. En agricolae se tratará los diseños Carolina I, II y III. Los análisis de estos diseños consisten en describir mediante un análisis de variancia las diferentes fuentes de variación. Otro tipo de análisis también se debe incluir para la prueba línea por probador (line x tester), el cual determina los indicadores genéticos, tales como índice de heredabilidad y efectos de habilidad combinatoria general y especifica que son vitales en el mejoramiento de plantas. Los análisis de estabilidad corresponden a pruebas sobre material genético sembrados en diferentes ambientes son también incluidos porque forman parte del estudio del mejoramiento de plantas. Así, se tienen el modelo AMMI (método de efectos principales aditivos y de interacción multiplicativa) y la estabilidad genética por efecto del ambiente.

Software estadístico comercial: En el mercado nacional existen programas estadísticos como SAS (Statistical Analysis System), de la compañía SAS Institute Inc USA, el cual utiliza procedimientos como: ANOVA, GLM y MIXED, para análisis de experimentos, y PROC PLAN, para el diseño de experimentos. El MINITAB, de la compañía MINITAB Inc USA, dispone de procedimientos de análisis estadístico especializado en control de la calidad, incluye los diseños de experimentos de naturaleza factorial y se utiliza más en la industria que en el sector agrícola. El SPSS, de la compañía SPSS Inc. USA, es bastante utilizado por algunas universidades, orientado al análisis estadístico con énfasis en muestreo, encuestas y modelos estadísticos, y aplicado en las ciencias sociales y la economía.

Construcción del paquete agricolae: El programa R dispone de la documentación necesaria para construir un paquete "Writing R Extensions" (R Development Core Team, 2008). Aquí se encuentra la terminología, la estructura de una función y dato, la estructura de la documentación de la función, y la configuración para su instalación en plataformas como Windows, Linux y Macintosh (Mac). La escritura de cada función se encuentra integramente en el lenguaje R; ésta contiene parámetros necesarios para producir un resultado final o intermedio que puede ser utilizado por otra función para producir una nueva información. La construcción final del paquete debe hacerse según la plataforma que se utilice para su creación. Así, en Windows, se necesita las herramientas de RTOOLS www.murdoch-sutherland.com/Rtools que

dispone del lenguaje perl, comandos en línea, el compilador MinGW, que permite incorporar programas escritos en C++ dentro del paquete a construir; y para incorporar ayudas en HTML y en PDF se requiere de las aplicaciones hhc.exe (Microsoft HTML Help Workshop) y Latex para PDF.

Revisión de software: Para el chequeo del paquete se utiliza el programa "check" y se ejecuta en línea de comandos con la orden de Rcmd (remote command). El parámetro es el nombre del paquete, así: "RCMD check agricolae". El programa "check" está escrito en lenguaje perl y realiza la validación de características como por ejemplo: "checking R files for syntax errors", el cual es la evaluación de la sintaxis de los archivos de extensión R (programa fuente del paquete). El programa "check" es un software libre de aproximadamente 2600 líneas de programación y en sus primeras líneas se indica la libertad de usarlo:

```
#-*- perl -*-
# Copyright (C) 2000-2008 R Development Core Team
# This program is free software; you can redistribute it and/or modify
# it under the terms of the GNU General Public License as published by
# the Free Software Foundation; either version 2, or (at your option)
# any later version.
```

Los resultados del programa para cada característica son: "OK" si es correcto, "Error" si presenta algún error grave, "Warning", si no es tan grave. La validación del programa es suficiente para dar conformidad al paquete. Este programa se está utilizando por mas de 8 años.

Modelo McCall: Está referido a calidad de software. El termino de calidad define al producto que satisface las condiciones para el cual fue creado y satisface al cliente. En el campo informático, hablar de calidad de software es más complejo, dado que el software es un producto de la mente y no está restringido por las leyes de la física y no se deteriora en el tiempo; el producto puede quedar obsoleto sin necesidad de malograrse, sólo por cambios del medio; por ejemplo, la plataforma del sistema (Windows nueva versión). El modelo McCall trata de cuantificar y analizar la calidad del software sobre la base de tres ejes o puntos de vista: Operación, Revisión y Transición del producto. En estos tres puntos se examina el software en: facilidad de uso, integridad, corrección, fiabilidad, eficiencia, facilidad de mantenimiento, facilidad de prueba, flexibilidad, reusabilidad, interoperabilidad y portabilidad.

Encuesta de satisfacción: Es el medio para captar información del uso del software y estudiar la calidad. Mediante la encuesta de satisfacción del software se podrá medir y cuantificar lo importante, bueno o malo y sobre todo si el cliente está satisfecho y dispuesto a recomendarlo. Las preguntas corresponden a ítems según la escala de Likert (1-5) para la satisfacción del producto; por ejemplo ¿Qué tan satisfecho está usted con las siguientes características de AGRICOLAE?
Calidad, Tiempo de respuesta, Integración con R ,…, etc.

CAPITULO III

METODOLOGÍA DE LA INVESTIGACIÓN

3.1 TIPO DE INVESTIGACIÓN

La presente investigación es del tipo no experimental, descriptiva y aplicada. Es no experimental porque se estudia los métodos aplicados en la investigación agriícola y se trata de construir funciones en una librería de múltiple uso en computadora. Es descriptivo porque se explora las metodologías existentes sobre el uso en investigación agrícola, se observa los casos en dónde y cómo se aplica; por ejemplo, los diseños y análisis de experimentos, los estudios genéticos en el mejoramiento de plantas, los indicadores de biodiversidad y métodos de determinación del tamaño óptimo de parcelas para investigación, y otras actividades relacionadas.
Es aplicado porque tiene relación con la investigación básica, dado que utiliza los resultados y avances científicos publicados de métodos estadísticos que son aplicados al campo agrícola, y se busca construir y/o modificar los procedimientos para ser utilizados en computadora como funciones, disponible en forma libre y que pueda aplicarse en cualquier análisis con posibilidad de actualizar.

3.2 DISEÑO DE LA INVESTIGACIÓN

Comprende el desarrollo de las funciones de aplicación en la investigación agrícola, principalmente los diseños de experimentos con sus análisis correspondientes, funciones de estabilidad fenotípica y genotípica, tales como modelos de estabilidad paramétrica y no-paramétrica. La evaluación de la calidad de la librería se realiza mediante el modelo McCall a través de la funcionalidad en el programa R, una

encuesta de satisfacción basada principalmente en la escala de Likert, un chequeo de toda la librería desde la sintaxis, configuración, documentación e instalación en plataformas como Windows, Linux y Mac, y se compara los resultados de ejemplos presentados con otro software estadístico, además de contar con la opinión de expertos sobre el uso de la librería en R.

3.3 POBLACIÓN Y MUESTRA

La población para el estudio de la calidad del software esta definida por todos los usuarios de R que utilizan la librería agricolae, cuyo número es infinito, su opinión permitirá cuantificar las bondades, veracidad e importancia de la librería. Para determinar el tamaño de muestra se requiere una referencia del posible tamaño de población propuesta, pudiendo ser definida por un número adecuado, dado que la opinión será más importante según el nivel del evaluador; éstos pueden ser Doctores (PhD), investigadores, graduados y estudiantes. Según el número de investigadores del CIP, principales usuarios, profesores y estudiantes que estarían usando la librería, podría considerarse una población de alrededor de 80 usuarios (30 investigadores del CIP, 10 investigadores de INIA, 20 estudiantes y profesores de Universidades y 20 extranjeros). El numero de encuestas según Birnbaum y Sirken (Cochran, 1972), con un margen de error del 10% y un nivel de confianza del 95% sería de 44:

$$n0 = (1.96*0.5 / 0.10)^2 = 96.04$$

$$n= n0/(1+n0/80) = 43.6 , \ n=44$$

3.4 VARIABLES E INDICADORES

Comprende las variables para evaluar la librería según el modelo McCall, para ello se utiliza la encuesta de satisfacción del usuario, medidas de chequeo por el sistema R, índice de 0-1 para calificar el software (Sadana, 2006), coeficiente de Cronbach para la fiabililidad de la encuesta, y la opinión de expertos sobre la funcionalidad de la librería.

Variables de la encuesta:

Variables que miden el conocimiento del usuario:

P1.- ¿Cuánto tiempo hace que utiliza la librería AGRICOLAE?

P2.- ¿Qué funciones estadísticas ha utilizado?

- Diseño de experimentos
- Comparaciones múltiples
- AMMI y estabilidad
- Estadística descriptiva
- Simulación y re-muestreo

P3.- Satisfacción respecto a las áreas de funciones presentadas en P2.

P7.- Importancia de los grupos de funciones en agricolae en P2.

P10.- Funciones que le agradaría incorporar al paquete.

P11.- Sobre productos similares ofrecidos por otros programas.

P17.- ¿Qué software estadístico usa aparte del R?

Variables que miden la satisfacción:

P4.- Satisfacción con características del paquete en:

- Calidad
- Tiempo de respuesta
- Integración con R
- Instalación de la librería
- Uso de ejemplos
- El manual
- Procesar con sus datos

P5.- Importancia de ciertas características en el uso de las funciones

- Calidad
- Presentación de resultados
- Sin costo
- Instalación por primera vez
- Conocimientos de agricultura
- Conocimientos de estadística
- Reproducir con sus datos

P6.- Sobre su más reciente experiencia con la librería.

- Agricolae es funcional
- Agricolae hace lo que dice
- Agricolae hace lo que yo necesito
- Agricolae es fácil de usar
- Agricolae es competitivo

P12.- Uso de agricolae en el futuro

P15.- ¿Recomendaría el uso de agricolae a los colegas o contactos de tu institución?

Variables que describen la satisfacción.

P8.- Qué le agrada de la librería agricolae.

P9.- Qué le desagrada de la librería agricolae.

P13.- ¿Por qué no usaría agricolae en el futuro?

P14.- ¿Por qué usaría agricolae en el futuro?

P16.- ¿Por qué no recomendar agricolae?

Variables complementarias.

P18. Genero (Varón/ Mujer)

P19. Edad

P20. Educación (Pre Grado/Maestría/ Doctorado/Otros)

P21. Ocupación (Estudiante/Profesor/Investigador)

Medidas de chequeo de la librería

El programa R tiene un sistema de chequeo CHECK que realiza 32 pruebas en línea y genera respuesta de la evaluación en:

Dependencias de la librería. La medida indicará si es posible su integración en R. Una librería no trabaja independientemente en una sesión de R si es invocada, ésta se integra automáticamente al sistema.

Prueba de instalación en cualquier sistema (Linux, Windows y Mac). Con esta medida se podrá conocer si no hay errores en su instalación en alguna de estas plataformas.

Ubicación automática en el sistema R para su uso. Al instalar el paquete en R, éste debe ser subido a la memoria sin error, listo para utilizarlo.

Compatibilidad de los nombres de archivos. Los nombres de los archivos que integra el paquete deben ser aceptado por el sistema; cada nombre corresponde a una base de datos, una función o una documentación.

Validación de caracteres, sólo caracteres ASCII en todos los archivos. Los caracteres aceptados por R deben ser sólo ASCII, cualquier otro carácter distinto será rechazado.

Sin error en la documentación. La documentación de un paquete comprende el contenido más la forma de acceso a las ayudas; esto significa que no debe haber ningún error en el manejo.

Sin error en el código fuente. Las funciones están presentadas en código de fuente abierta, el cual puede ser editado en cualquier momento. La forma de usarlo es interpretativa, por lo tanto con esta medida se debe asegurar la funcionalidad del código.

Funcionalidad de los ejemplos. Los ejemplos incorporados en el paquete son de apoyo para el mejor uso de las funciones. Estos deben estar funcionando correctamente.

Funcionalidad del sistema de ayuda en pdf y html. Una medida sobre la funcionalidad de las ayudas en pdf y html es importante para garantizar la integridad en el sistema.

Codificación del manual en látex correctamente y sin error. El manual del paquete es automáticamente generado, entonces se necesita una medida de la codificación para su interpretación.

Índice de calidad. Criterio de aceptación del software (Sadana, 2006):

Malo:	0.00 – 0.25
Regular:	0.25 – 0.50
Bueno:	0.50 – 0.75
Muy bueno:	0.75 – 1.00

La medida corresponde a la evaluación de los diferentes tipos de funciones de agricolae: diseños de experimentos, comparaciones múltiples, estabilidad de genotipos, estadística descriptiva y simulación.

Indice alfa de Cronbach (Hayes, 2008) para determinar la fiabilidad del instrumento, dado por los ítems de las preguntas P4, P5, P6, P12 y P15 con sus respectivos ítems que miden la satisfacción del usuario. La fórmula es:

$$\alpha = \frac{k}{k-1}\left(1-\frac{\Sigma S_i^2}{S_t^2}\right);\ \begin{cases} K: & Items \\ S_i^2: & Variancia.Item-i \\ S_t^2: & Variancia.Total \end{cases}$$

La variancia total corresponde a la variancia de la suma de puntajes. Cuanto más cerca de 1 es el indice del Cronbach más fiable el instrumento.

Opinión del experto. Se utilizará la información de la encuesta llenada por los expertos.

3.5 TÉCNICAS E INSTRUMENTOS DE RECOLECCIÓN DE DATOS

El material necesario para la construcción de la librería corresponde a la base teórica de cada función con su documentación y ejemplos, la base de datos para incluir en la librería, los procedimientos para la construcción de la librería, y el sistema de chequeo y evaluación de la calidad de la librería en funcionamiento.

3.5.1 BASE TEÓRICA DE LAS FUNCIONES

Diseños de experimentos

Los tres principios básicos de un diseño experimental propuesto por Fisher: Repetición, Aleatorización y Control Local, son la base para la construcción de los diseños experimentales (LeClerg, 1962). El principio más importante para la generación sistemática del diseño en computadora es la aleatorización de tal forma que satisfaga las condiciones teóricas del diseño; en segundo lugar, las repeticiones que deben ser un parámetro condicionado por el investigador; y finalmente, el control local basado en la estructura del diseño, que permita el control de factores externos y controlar parcialmente. El error experimental, el control local es dado al asignar el tipo de diseño que debe aplicar el investigador.

En consecuencia, la programación de cualquier diseño experimental en computadora estará basada en los tratamientos por aplicar, el método de aleatorización, el número de repeticiones, y el tipo de diseño.

Parámetros para la función generador del diseño experimental

Se basan en el concepto del diseño experimental que consiste en la aplicación de los tratamientos a las unidades experimentales. Los parámetros por utilizar son:

Tratamientos: Un vector correspondiente al factor de estudio.

Repetición: Un vector o constante si el número es variable o constante para cada tratamiento.

Número: Una constante para iniciar la numeración de las unidades experimentales.

Semilla: Una constante de reproducibilidad del diseño (0=no reproducible, otro valor distinto para reproducir el diseño).

Método de aleatorización: Siendo la generación de números aleatorios por computadora, es necesario escoger un método. Son diferentes métodos según la recurrencia de congruencia aditiva o multiplicativa.

Estos parámetros serán suficientes para la generación de cualquier diseño.

Diseño completamente aleatorio. Se aplica cuando el material es homogéneo (igual de condición relacionado con la medida que se evaluará). Los tratamientos se asignan aleatoriamente en todo el material y pueden tener igual o diferente repetición; esto significa que puede ser una constante o un vector. Así un croquis de 4 tratamientos(a, b, c, d) con repeticiones (4, 3, 4, 2) sería:

```
Plots: 1   2   3   4   5   6   7   8   9  10  11  12  13
Trt  :"a" "c" "b" "c" "a" "d" "a" "b" "d" "c" "a" "b" "c"
```

Diseño bloques completamente aleatorio. Se aplica cuando el material es heterogéneo pero se puede agrupar en grupos homogéneos. Cada grupo homogéneo es un bloque y todos los tratamientos se asignan aleatoriamente en cada bloque. En este diseño los bloques son las repeticiones; esto hace que la repetición sea una constante. Un croquis de 4 tratamientos(a, b, c, d) en 5 bloques seria:

```
     [,1] [,2] [,3] [,4]
[1,] "a"  "c"  "d"  "b"
[2,] "d"  "a"  "b"  "c"
[3,] "c"  "a"  "d"  "b"
[4,] "b"  "d"  "a"  "c"
[5,] "a"  "c"  "d"  "b"
```

Diseño cuadrado latino. Se aplica cuando el material es heterogéneo pero se puede agrupar el material formando cierta homogeneidad por filas y columnas, formando un cuadrado; es decir, si se debe aplicar "t" tratamientos, se debe tener "t" unidades experimentales por columna y por fila; en total serán t*t unidades. La aleatorización debe ser por fila y por columna pero todos los tratamientos deben estar siempre en cada fila y en cada columna. Un croquis de 4x4 con Trt = (a, b, c, d) sería:

```
     [,1] [,2] [,3] [,4]
[1,] "b"  "c"  "d"  "a"
[2,] "d"  "a"  "b"  "c"
[3,] "c"  "d"  "a"  "b"
[4,] "a"  "b"  "c"  "d"
```

Diseño greco latino. Se aplica cuando el material es heterogéneo pero se puede agrupar el material formando cierta homogeneidad por filas y columnas, formando un cuadrado. En este diseño se aplica un segundo tratamiento del mismo número y se debe aleatorizar ambos en el cuadrado, de tal manera que exista una ortogonalidad entre las filas, columnas y el otro tratamiento. Este diseño ha sido muy complejo en

construir;el principal autor,Snedecor y Cochran,en su libro de diseños experimentales (Cochran, 1992), indicó que es imposible construir un diseño de 6x6. Un croquis de 4x4 con T1 = (a, b, c, d) y T2 = (w, x, y, z) sería:

```
     [,1]  [,2]  [,3]  [,4]
[1,] "b x" "a y" "d w" "c v"
[2,] "a w" "b v" "c x" "d y"
[3,] "d v" "c w" "b y" "a x"
[4,] "c y" "d x" "a v" "b w"
```

Diseños bloques incompletos.- Bastante aplicado cuando son muchos tratamientos (>10) y no es posible colocar todos en un bloque, en estudios de degustación el evaluador no puede calificar todos los tratamientos, en mejoramiento genético de plantas, el número de tratamientos a evaluar mayormente sobrepasa 100 tratamientos llegando en algunos casos a mas de 1000. Los diseños de bloques incompletos siguen siendo las alternativas para estos estudios. Entre las posibilidades a escoger esta en los diseños balanceados y parcialmente balanceado.

Bloques incompletos balanceado. Dado un número de tratamientos, se debe hacer una distribución aleatoria en el cual se produzca todas las seleccionadas de un tratamiento frente a otro, un croquis para 4 tratamientos en bloques de tamaño 3 seria:

```
     [,1] [,2] [,3]
[1,] "b"  "d"  "c"
[2,] "d"  "b"  "a"
[3,] "a"  "b"  "c"
[4,] "a"  "d"  "c"
```

Como se puede observar un par de tratamientos se esta comparando dos veces y esto corresponde al parámetro lambda (λ), a su vez cada tratamiento se repite 3 veces.

Para un número grande de tratamientos es imposible de aplicar este diseño porque se necesitaría un gran número de unidades experimentales, para tal fin se requiere de diseños parcialmente balanceados, en la cual solo algunos de los tratamientos están en un semibloque, entonces se debe estimar variancias de la diferencia de tratamientos que están dentro de un bloque y entre bloques diferentes.

Diseños de bloques incompletos parcialmente balanceados.

Diseño latice simple y triple. Estos diseños son bastante utilizados en el mejoramiento de plantas, en donde el número de tratamientos debe ser un cuadrado

perfecto (9, 16, 25, 36, ...). Para números grandes sólo se puede utilizar pocas repeticiones (con 2 repeticiones es un latice simple, y 3, en un latice triple). En la organización de tratamientos en un latice simple, los tratamientos se organizan por filas, y en la segunda repetición, por columnas. En el triple, se agrega una repetición en la que se utiliza el criterio de la distribución de un cuadrado latino y se hace la equivalencia de las letras del cuadrado latino con los tratamientos de la primera repetición. La primera letra es el primer bloque; la segunda, el segundo bloque, y así sucesivamente para los 9 tratamientos. Las tres repeticiones serían:

	[,1]	[,2]	[,3]
[1,]	1	2	3
[2,]	4	5	6
[3,]	7	8	9

	[,1]	[,2]	[,3]
[1,]	1	4	7
[2,]	2	5	8
[3,]	3	6	9

	[,1]	[,2]	[,3]
[1,]	1	7	8
[2,]	2	4	9
[3,]	3	5	7

El cuadrado latino utilizado en este caso es:

	[,1]	[,2]	[,3]
[1,]	A1	B2	C3
[2,]	B4	C5	A7
[3,]	C7	A8	B9

El primer bloque estará formado por los tratamientos 1, 7 y 8 que corresponden a la letra A; el segundo bloque por los tratamientos 2, 4 y 9, correspondientes a la letra B; y el tercer bloque a los tratamientos 3, 5 y 7.

Diseño alfa: Son los diseños alternativos a los latices cuando no se tiene un cuadrado perfecto y para un número más grande de tratamientos los diseños alfa son rectangulares, donde el tamaño del bloque (k) es inferior al número de bloques (s) y el producto kxs es igual al número de tratamientos. Cada rectángulo corresponde a una repetición. Para construir estos diseños se inicia con ciertos arreglos llamados "generador alfa", planteado por Patterson y Williams (Patterson, 1976), los cuales permiten construir arreglos intermedios según el número de repeticiones y estos generan los diseños alfa. Los arreglos planteados por Patterson y Williams bajo las restricciones: $k \leq s$ y $r \leq 4$ y las condiciones para el número de repeticiones son presentados en series como se muestra en la tabla 3.1. Para 2 repeticiones se utilizará la serie I; para 3 repeticiones, la serie II y III; y para 4 repeticiones, la serie IV.

Tabla 3.1 Arreglos básicos para generar el diseño alfa, (Patterson, 1976)

Serie I (r=2, k ≤ s)	
0	0
0	1
0	2
.	.
.	.
0	s-1

Serie II (r=3, k ≤ s, s=impar)		
0	0	0
0	1	s-1
0	2	s-2
.	.	.
.	.	.
0	s-1	1

Serie III (r=3, k ≤ s-1, s=par) s' = s/2		
0	0	0
0	1	s'
0	2	1
.	.	s' + 1
.	.	2
.	.	.
0	s-3	s - 2
0	s-2	s' - 1

Serie IV (r=4, k ≤ s, s=impar, s≠ 0 mod 3) s" = (s + 1)/2			
0	0	0	0
0	1	s - 1	s"
0	2	s - 2	1
.	.	s - 3	s" + 1
.	.	s - 4	2
.	.	.	.
0	s-3	s - 1	s" - 2
0	s-2	s - 2	s - 1
0	s-1	s' - 1	s" - 1

Los arreglos intermedios se construyen en base a una serie. Así, por ejemplo, si se tiene 12 tratamientos (s=4, k=3, r=2), entonces se construye dos arreglos intermedios de orden 3x4; el primero se construye siempre bajo un orden secuencial iniciando en cero; para esto se utiliza la primera columna del generador, en donde todos son valores iguales a cero, resultando la siguiente matriz:

0	1	2	3
4	5	6	7
8	9	10	11

La segunda matriz intermedia se construye sumando la segunda columna del generador a cada una de las columnas de la primera matriz, siempre que no se supere el máximo de la fila; si lo supera, ésta continua con el primer valor de la fila. Para más repeticiones se suma la tercera, cuarta columna del generador alfa a la primera matriz intermedia como se indica:

0	1	2	3
5	6	7	4
10	11	8	9

Luego se debe asignar los tratamientos al azar a cada valor de los arreglos intermedios; por ejemplo, el siguiente orden aleatorio para tratamientos: 8, 4, 2, 9, 12, 5, 7, 11, 1, 6, 3 y 10; al valor 0 le corresponde el tratamiento 8; al valor 1, el tratamientos 4 y ; así sucesivamente hasta el valor 11, al cual le correspondería el tratamiento 10. Luego se transpone estos arreglos y cada uno corresponde a una repetición, así:

	Rep = 1		
1	8	12	1
2	4	5	6
3	2	7	3
4	9	11	10

	Rep=2		
5	8	5	3
6	4	7	10
7	2	11	1
8	9	12	6

Finalmente se procede a aleatorizar dentro de cada bloque y luego entre bloques de cada repetición y se renumera los bloques. Así se tendría el diseño alfa con el siguiente arreglo aleatorio:

	Rep = 1		
1	9	10	11
2	12	8	1
3	6	4	5
4	3	2	7

	Rep=2		
5	10	4	7
6	2	11	1
7	5	8	3
8	9	12	6

Diseño cíclico. Utiliza la rotación cíclica de los tratamientos formando pequeños semi-bloques para un número fijo y no muy grande de tratamientos. El diseño es eficiente y robusto para 6 a 30 tratamientos y con un número de repeticiones menor o igual a 10. Las tablas para la generación de estos diseños están publicadas; el proceso consistirá en el uso de tablas y la aleatorizaciónm.

En la tabla 3.2 se presenta los arreglos para obtener el generador del diseño cíclico para 6 a 15 tratamientos hasta 10 repeticiones para $r = ik$ ($i \geq 1$), siendo k el tamaño del bloque.

El generador es un vector de tamaño k, los k-1 elementos esta formado por los elementos presentados en la tabla para las condiciones (k y r) y el ultimo tratamiento el correspondiente al valor del número de tratamientos.

Vease por ejemplo los generadores para los siguientes casos:

Caso 1: t=6, r= 3, k =3; Generador: 1 2 4

Caso 2: t=6, r= 6, k =3; Generador 1: 1 2 4; Generador 2: 1 3 2

Para el caso 1, el generador crea una secuencia de bloques en donde el primer bloque es el mismo generador; los siguientes se obtienen sumando 1 a cada columna con el módulo número de tratamientos hasta completar un número de bloques igual al número de tratamientos. De continuar con más bloques se reproduce la misma secuencia inicial, lo cual no es correcto, y termina antes de que ocurra, tal y como se presenta en el siguiente arreglo:

Bloque			
1	1	2	4
2	2	3	5
3	3	4	6
4	4	5	1
5	5	6	2
6	6	1	3
Repite	1	2	4

Tabla 3.2. Arreglos para generar el diseño cíclico (Kuehl, 2000)

K	R	Primeros k-1 tratamientos
2	2	1
	4	1
	6	1
	8	1
	10	1
3	3	1 2
	6	1 3
	9	1 2
4	4	1 2 4
	8	1 2 5
5	5	1 2 3 5
	10	1 3 4 5
	10	1 3 4 7
6	6	1 2 3 4 7
7	7	1 2 3 4 5 8
8	8	1 2 3 4 5 7 9
9	9	1 2 3 4 5 6 8 10
10	10	1 2 3 4 5 6 7 10 11

6	7	8	9	10	11	12	13	14	15
2	2	2	2	2	2	2	2	2	2
3	4	4	4	4	4	4	6	5	5
4	3	3	3	3	6	6	3	7	3
6	5	5	5	5	3	3	5	4	8
5	6	6	6	6	5	5	4	6	6
4	4	4	4	5	5	5	5	5	5
2	4	8	7	8	8	6	8	8	9
4	4	5	6	4	4	7	5	7	6
3	7	8	8	7	8	8	10	8	8
3	7	8	9	3	7	7	7	7	7
6	6	8	8	8	8	8	8	10	11
6	6	8	9	10	9				
						8	12	13	11
	6	6	6	6	11	11	11	11	11
		6	6	10	10	10	10	10	11
			6	10	10	10	10	12	12
				9	9	9	11	11	11
					8	8	8	13	13

Para la aleatorización se puede asignar una secuencia aleatoria de tratamientos o aleatorizar los bloques y luego los tratamientos dentro del bloque, y finalmente renumerar los bloques. Mediante la segunda alternativa, un resultado aleatorio sería:

Bloque			
1	3	6	4
2	5	6	2
3	2	5	3
4	1	4	5
5	3	6	1
6	4	2	1

En caso de tener dos generadores, se tendrá una secuencia adicional de bloques aumentando así el número de repeticiones, que es el caso para 6 repeticiones en donde el segundo generador es: 1, 3, 2.

Diseño cíclico con fila y columna. Es un diseño similar al cíclico, pero con la diferencia de que en cada columna se encuentran todos los tratamientos. En este caso, la aleatorización debe ser de los tratamientos (asignación aleatoria de tratamientos); aleatorizar las filas y luego las columnas; y ya no los tratamientos dentro de cada bloque (fila). De esta manera, se tendría todos los tratamientos en cada columna, así:

Bloque			
1	6	1	3
2	3	4	6
3	1	2	4
4	5	6	2
5	2	3	5
6	4	5	1

Análisis del diseño bloques incompletos parcialmente balanceado. (Williams, 1977). El diseño está formado por "r" repeticiones y v=ks tratamientos, son llamados diseños resoluble , donde:

k = tamaño del semibloque (unidades por bloque)

s = número de semibloques por repetición

b = rs bloques

Modelo del diseño experimental:

$Y = R\rho + X\tau + \varepsilon$,

Donde:

Y: Vector nx1 observaciones

R: Matriz de diseño para repeticiones nxr

X: Matriz de diseño para tratamientos de dimensión nv

La matriz X está formada por 1 y 0; 1 si el tratamiento está presente, y 0 si está ausente.

ρ y τ son vectores efecto de la repetición y del tratamiento respectivamente..

ε: vector de efectos aleatorios con $E[\varepsilon] = 0$,

E[εε']= $E[\varepsilon\varepsilon'] = \sigma^2\left(I_n + \phi k^{-1} ZZ'\right)$

donde Z es la matriz diseño del bloque y σ^2 y ϕ, los parámetros a ser estimados.

I_n es una matriz identidad de orden n; k es el tamaño del semibloque y una constante ϕ definida por $\phi = r(E_b - E_e)/ [(r-1)E_e]$.

ϕ es un escalar determinado por la relación de los errores E_e (cuadrado medio residual) y E_b (cuadrado medio residual para bloques dentro de repeticiones).

La matriz de incidencia de un diseño N está definido por el producto de X' y Z, así: N = X'Z para un diseño resoluble. La matriz de la concurrencia del bloque N'N estará formada por submatrices de orden s. N'N puede expresarse, entonces, mediante la siguiente expresión:

$N'N = [\{P_{lm}\}]$, l y m varían de 1 a r,

donde P_{lm} son las submatrices de orden s y mantienen la siguiente relación:

$$P_{lm}1 = P'_{lm}1 = k1 \text{ y } P_{ll} = I_s; \text{ (l, m 1,...,r).}$$

La estructura de la matriz N'N es la matriz del bloque bxb generalmente tiene una dimensión más pequeña que la matriz de tratamientos txt; entonces es mejor estimar tau con la matriz N'N.

Estimación de tau τ.

τ representa el efecto de las medias de tratamiento ajustado.

La estimación por el método de mínimos cuadrados ponderados genera las siguientes ecuaciones normales (Williams, 1977):

$$r\hat{\tau} + N\hat{\beta} = X'y$$

$$N'\hat{\tau} + k(\phi^{-1} + 1)\hat{\beta} = Z'y$$

donde $\hat{\beta}$ corresponde al efecto de los semibloques; es un vector bx1.

La solución para $\hat{\beta}$ es obtenida de la primera y reemplazada en la segunda ecuación, obteniendo la siguiente estimación:

$$\hat{\tau} = r^{-1}\left[X'y - N\left\{k(\phi^{-1} + 1)I_b - r^{-1}N'N\right\}^{-}C_0\right]$$

(-) en el exponente indica inversa generalizada de la expresión:

$$k(\phi^{-1} + 1)I_b - r^{-1}N'N$$

$C_0 = Z'(I_n - r^{-1}XX')$ y .

Se define una nueva matriz con valores 0, 1 y –1 en W de orden b como:

$W=N'N - k I_b - g(J_r - I_r) \otimes J_s$;

donde "g" es una constante que depende de la relación del tamaño del bloque y del número de bloques por repetición.

g=[k/s] si q ≤ ½ s

g=[k/s] + 1 si q > ½ s

donde q = k - [k/s] s

[k/s] es la parte entera de k/s

J_r es una matriz de unos de orden r

⊗ indica producto de Kronecker.

Desde que los efectos de los tratamientos son ortogonales a la media general y a los efectos de las repeticiones, la ecuación de $\hat{\tau}$ (tau) puede ser expresado como:

$$\hat{\tau} = r^{-1} X' y - \lambda N \left(I_b - \lambda W \right)^{-1} c_0$$

Donde λ es:

$$\lambda = \left(rk\left(\phi^{-1} + 1 \right) - k \right)^{-1}$$

Similarmente, la matriz variancia de las medias ajustadas es:

$$Var(\hat{\tau}) = \sigma^2 r^{-1} \left\{ I_v + \lambda N \left(I_b - \lambda W \right)^{-1} N' \right\}$$

Conocida la matriz variancia, ya se puede hallar la variancia estimada de la diferencia de dos medias ajustadas, mediante la siguiente relacion:

$var(\tau_i - \tau_j) = var(\tau_i) + var(\tau_j) - 2 * cov(\tau_i, \tau_j)$

Finalmente estas expresiones pueden ser utilizadas para la comparación de dos tratamientos.

Comparaciones múltiples. Éstas se refieren a la comparación de medias de tratamientos en un estudio experimental con repeticiones o en muestras de poblaciones. En una comparación estadística se necesita conocer cual es la probabilidad (p-valor) mediante la cual se acepte o se rechace una igualdad entre dos medias y también cuáles serían los grupos de tratamientos que no son diferentes significativamente. Para estos casos, se dispone de pruebas paramétricas y no-paramétricas.

Pruebas paramétricas. Son las pruebas que consideran el cumplimiento de ciertos supuestos del análisis de variancia como son la normalidad, la aditividad del modelo, la homogeneidad de las variancias y la independencia de los errores. Entre las

pruebas más utilizadas se encuentren la mínima diferencia significativa de Student (LSD), para pocos tratamientos; la prueba de Tukey, para más de 5 tratamientos; y la prueba de Waller-Duncan, una prueba bayesiana que minimiza los dos tipos de error que se puede cometer en una decisión estadística.

Prueba de la mínima diferencia de significación (LSD). Para esta prueba es necesario que el análisis de variancia sea significativo para los tratamientos. Con los grados de libertad del error y su variancia estimada se construye un valor mínimo para la diferencia significativa:

$$LSD = t(\alpha, gl)sd$$

sd: desviación estándar de la diferencia de promedios

$$sd = \sqrt{\hat{\sigma}^2\left(\frac{1}{n1}+\frac{1}{n2}\right)};$$

n1 y n2 son las repeticiones; $\hat{\sigma}^2$ es la variancia del error, que es estimada por el cuadrado medio del error del análisis de variancia; y t (α, gl), el valor tabular para la significación; α es el nivel de significación y gl es el grado de libertad del error.

Para una comparación, si la diferencia de dos promedios supera el valor LSD, entonces éstos son diferentes estadísticamente. En la comparación estadística se requiere del valor del p-valor, el cual es la probabilidad del valor absoluto en donde la variable t-student sea mayor del cociente dado por la diferencia absoluta de los tratamientos entre su desviación estándar de la diferencia. En otros casos se requiere del agrupamiento de tratamientos que no son diferentes significativamente, asignando letras iguales.

Prueba de Tukey (HSD). Llamada prueba de la diferencia honesta de significación porque puede ser aplicada a un gran número de tratamientos sin perder la precisión en la significación de una comparación, caso que en una prueba LSD se pierde precisión al incrementar el número de tratamientos. En esta prueba se requiere de igualdad de repeticiones porque se utiliza una sola desviación estándar, que es la desviación estándar de promedios; si son diferentes se utiliza el promedio armónico de las repeticiones. Por ser una prueba honesta, no requiere de una significación previa de tratamientos en el análisis de la variancia, (Steel, 1997)

$$HSD = AES.tukey(\alpha, gl, t)Sx$$

AES es el valor tabular de la tabla Amplitudes Estudiantizadas Significativas de Tukey que se determina por el nivel de significación (α), los grados de libertad del error (gl) y el número de tratamientos del experimento (t); el valor Sx es la desviación estándar de promedios de los tratamientos.

$$Sx = \sqrt{\frac{\hat{\sigma}^2}{r}}$$

Las diferencias de los promedios mayores a HSD son diferentes significativamente. En una comparación se requiere también de los p-valores de la comparación, para esto se debe utilizar la función de distribución de Tukey.

Prueba de Waller-Duncan.- Es un método bayesiano para comparaciones múltiples de tratamientos; utiliza los valores de F calculados en el cuadro del análisis de variancia, los grados de libertad de tratamientos, del error y la razón de gravedad del error (k) o peso del error tipo I o II, tales como 50:1, 100:1 y 500:1, equivalentes a los niveles de significación de α=0.10, 0.05 y 0.01. El proceso tradicional es usar la tabla de Waller-Duncan (Steel, 1997) e interpolar si no se encuentra el valor. Para el uso por computadora se debe programar la función para hallar este valor tabular. La prueba es bastante utilizada porque minimiza los dos tipos de riego, del error tipo I, que es el rechazo de una hipótesis verdadera, o del tipo II, de aceptar la hipótesis siendo falsa.

Para la comparación múltiple se calcula el valor de la mínima diferencia significativa obtenida con el valor tabular y la desviación estándar de la diferencia.

$$DMS = waller(k, q, f, Fc)Sd$$

Sd: Desviación estándar de la diferencia de medias

Waller(k,q,f,Fc), el valor tabular de Waller-Duncan.

k: razón de gravedad del error

q: grados de libertad del error

f: grados de libertad de tratamientos

Fc: valor de F calculado a través del Análisis de variancia.

El criterio para decidir la diferencia significativa de dos tratamientos es la diferencia absoluta de sus promedios, de ser superior a la mínima diferencia significativa dada por Waller.

Pruebas no-paramétricas. La utilización de estos métodos se hace recomendable cuando no se puede asumir que los datos se ajusten a una distribución conocida o cuando la medida proviene de valores asignados; las medidas son cambiadas a rangos que son un valor de orden. Para estudios comparativos (más de dos tratamientos) se utiliza las pruebas de Kruskal-Wallis, Friedman y Durbin.

Kruskall Wallis. Es apropiada cuando los datos provienen de un diseño completamente al azar o cuando las muestras provienen de diferentes poblaciones. El estadístico de la prueba está en función de las medidas de orden; éstas pueden tener empates o no. Para el caso sin empates se utiliza el estadístiuco H (Connover, 1999):

$$H = \frac{12}{N(N+1)}\sum_i \frac{R_i^2}{n_i} - 3(N+1)$$; N: total de observaciones;

ni: repeticiones; Ri: la suma de los rangos del tratamiento i.

En el caso de empates, se utiliza el estadístico H (Connover, 1999):

$$H = \frac{1}{S^2}\left(\sum \frac{R_i^2}{n_i} - N\frac{(N+1)^2}{4}\right);\ S^2 = \frac{1}{N-1}\left(\sum r_{ij}^2 - N\frac{(N+1)^2}{4}\right)$$

rij : es el rango de un valor ; S^2: es la variancia del rango;

H se distribuye como una chi-cuadrada con (t-1) grados de libertad. Para realizar una comparación múltiple, se utiliza los promedios de los rangos (valores del orden) y se compara la diferencia absoluta con el valor crítico dado por:

$$t(0.05, N-t)\sqrt{\left(S^2\frac{N-1-H}{N-t}\right)\left(\frac{1}{ni}+\frac{1}{nj}\right)};$$

ni y nj son las repeticiones de los tratamientos por comparar.

Para que exista diferencia significativa entre los tratamientos, el valor absoluto de la diferencia de promedios debe ser mayor que el valor crítico.

Prueba de Friedman. Friedman (1937) propuso una prueba para datos sin distribución conocida cuando estos correspondan a un diseño de bloques completos al azar. En el caso de usar jueces para pruebas de degustación, los jueces representan a los bloques; el proceso consiste en asignar un valor de orden (Rango) a la respuesta de los tratamientos dentro de cada bloque de menor a mayor y con la

suma de los rangos para cada tratamiento se realiza la comparación (Connover, 1999); el estadístico de la prueba es:

$$H = \frac{12}{bt(t+1)} \sum_i r_{i.}^2 - 3b(t+1)$$; caso sin empates

Los valores 12 y 3 son constantes, y para el caso de empates:

$$H = \frac{(t-1)}{A_1 - C_1} \left[\sum_{i=1}^{t} r_{i.}^2 - b C_1 \right],$$

donde:

$$A_1 = \sum_{i=1}^{b} \sum_{j=1}^{t} r_{ij}^2 \quad \text{y} \quad C_1 = bt(k+1)^2/4$$

La variable H sigue una distribución Chi-cuadrada con t-1 grados de libertad.

t = número de tratamientos; b = número de bloques

La significación se da por la magnitud de H. Si ésta supera al valor tabular (crítico), el efecto de tratamientos es significativo.

Ri representa la suma de los rangos del tratamiento "i", que es $r_{i.}$

Entonces la diferencia es significativa si:

$$|R_i - R_j| > LSD = t_{\alpha(b-1)(t-1)} \sqrt{\frac{2(b A_1 - \Sigma R_i^2)}{(b-1)(t-1)}}$$

Los grados de libertad para el valor de t-student son: (b-1)(t-1)

Prueba de Durbin. Se utiliza para comparar muchos tratamientos en un diseño de bloques incompletos balanceados. A la variable respuesta se le asigna el valor del orden estadístico en cada bloque incompleto y con estos nuevos valores se realiza la prueba (Connover, 1999). Para hallar el estadístico se utiliza los siguientes parámetros:

t: número de tratamientos

k: tamaño del bloque

s: número de bloques

r: repeticiones

λ: número de bloques en que un i-ésimo tratamiento y un j-ésimo tratamiento aparecen juntos.

El estadístico de la prueba T1 para el caso sin empates es:

$$T_1 = \frac{12(t-1)}{rt(k-1)(k+1)} \sum_{j=1}^{t} \left(R_j - \frac{r(k+1)}{2} \right)^2$$

En el caso de empates es necesario hallar dos constantes A y un factor de corrección C.

$$A = \sum_{i=1}^{s}\sum_{j=1}^{t}(R(x_{ij}))^2 ;\ C = \frac{sk(k+1)^2}{4}$$

Y el estadístico T1 es:

$$T_1 = \frac{(t-1)\left[\sum_{j=1}^{t} R_j^2 - rC\right]}{A-C}$$

Para la comparación múltiple de tratamientos, el valor de la diferencia mínima es calculado con el valor de t-student al nivel α de significación y grados de libertad igual a sk – s – t + 1.

$$LSD = t(1-\alpha/2)\left[\frac{(A-C)2r}{sk-s-t+1}\left(1-\frac{T1}{s(k-1)}\right)\right]^{1/2}$$

Y para el caso de no empates:

$$LSD = t(1-\alpha/2)\left[\frac{rk(k+1)}{6(sk-s-t+1)}\left(s(k-1)-T_1\right)\right]^{1/2}$$

El valor de λ no es necesario para la prueba, sólo se utiliza para ver el balance del diseño:

$\lambda = \frac{r(k-1)}{(t-1)}$; el valor es entero si hay balance en el diseño.

Para que una diferencia sea significativa, debe cumplir la siguiente desigualdad: $|R_i - R_j| > LSD$.

Diseño genético y análisis.

Corresponde al manejo del material genético como clones, entradas y variedades del cultivo que mediante modalidades de cruzamientos se estudia a los progenitores (padres) y progenies (hijos) sobre la respuesta; por lo general es el rendimiento y se estudia el efecto de la variabilidad genética y en algunos casos la heredabilidad y la heterosis. Para estos estudios se tiene los diseños carolina, la prueba de líneas por

probadores y la estabilidad del material genético en el espacio y tiempo (Singh, 1979).

Diseños genéticos carolina. El diseño comprende la manera de formar y utilizar los progenitores y las progenies. Son tres los diseños carolina (Singh, 1979):

Diseño carolina I. En este diseño, la población es producida por el cruzamiento al azar de dos líneas puras. De esta población se selecciona al azar un individuo y se utiliza como macho, y otro conjunto de "p" plantas como hembras, formando "p" familias de hermanos completos. Del mismo modo se hace para un gran número de machos formando grupos. Las hembras no se utilizan para un segundo apareamiento.

El modelo estadístico es:

$$y_{ikjlt} = m + a_i + b_{ij} + r_{ik} + s_{ikl} + (bs)_{ijkl} + e_{ijklt}$$

i=1,2,...,s; j=1,2,..., r; k=1,2,...,m; l=1,2,...,f; t=1,2,...,n

m: media general

a_i: efecto del i-esimo grupo

b_{ij}: efecto de la j-esima repetición dentro del grupo i,

r_{jk}: efecto del k-esimo macho en el i-esimo grupo,

s_{lki}: efecto del l-esima hembra cruzado con el k-esimo macho en el grupo i.

bs: interacción de las repeticiones con las hembras, y

e_{ilklt}: el error.

Como resultado estadístico se requiere el análisis de variancia y las componentes de variancia.

Fuentes	Grados de libertad:
Grupos	s-1
Rep(grupos)	s(r-1)
Machos(grupos)	s(m-1)
Hembras(machos en grupos)	sm(f-1)
Rep x hembras	s(fm-1)(r-1)
Error	smfr(n-1)
Total	smfrn-1

Componentes de variancia:

Var(machos) = (CM(machos(grupos)) – CM(hembras(machos en grupos)))/ (f r n)

Var(hembras) = CM(hembras(machos en grupos))-CM(rep x hembras))/(n r)

Con estos componentes se calcula la variancia aditiva y dominante.

Var(aditiva) = 4 Var(machos)

Var(dominante) = 4 (Var(hembras) – Var(machos))

Diseño carolina II. En contraste con el diseño I, en este diseño tanto la parte paterna como materna producen medios hermanos; de esta población se selecciona al azar n1 machos y n2 hembras y cada macho se cruza con cada una de las hembras. Así se tiene n1 x n2 progenies, las culaes son analizadas adecuadamente en un experimento. La variación entre las familias se divide en la variación de machos, hembras y la interacción machos x hembras.

El modelo estadístico con un factor de grupos (set) es:

$$y_{ikjl} = \mu + a_i + b_{ij} + r_{ik} + s_{il} + (rs)_{ilk} + e_{ijkl}$$

i=1,2,...,s j=1,2,..., r k=1,2,...,m l=1,2,...,f

μ: media general,

a_i: efecto del i-ésimo grupo,

b_{ij}: efecto de la j-ésima repetición dentro del grupo i,

r_{jk}: efecto del k-ésimo macho en el i-ésimo grupo,

s_{il}: efecto del l-ésimo macho en el i-ésimo grupo,

rs_{ilk}: es la interacción machos x hembras en el i-esimo grupo, y

e_{ijlk}: error.

El resultado estadístico corresponde a un análisis de variancia y componentes de variancia para la interpretación genética.

Fuentes	Grados de libertad:
Grupos	s-1
Rep(grupos)	s(r-1)
Machos(grupos)	s(m-1)
Hembras(grupos)	s(f-1)
Machos x hembras (grupos)	s(m-1)(f-1)
Error	s(mf-1)(r-1)
Total	smfr-1

Componentes de variancia:

Var(machos) = (CM(machos(grupos)) – CM(machos x hembras(grupos)))/ (f r)

Var(hembras) = CM(hembras(machos en grupos))-CM(machos x hembras(grupos)))/(m r)

Var(machos x hembras) = (CM(machos x hembras(grupos)) – CM(error))/r

Con estas variancias se calcula la variancia aditiva y dominante.

Var(aditiva) = 4 Var(machos) ó 4 Var(hembras)

Var(dominante) = 4 Var(machos x hembras)

Diseño carolina III. La población se produce por el cruzamiento de dos líneas puras. El material para la estimación de los parámetros genéticos se produce por el cruce de una nueva generación de individuos machos seleccionados al azar para el cruce con hembras que son las dos líneas puras, el cual produce 2n hermanos. Al igual que en otros diseños, cada conjunto es la unidad de un experimento que se repite r veces en un diseño de bloques al azar.

El modelo estadístico con un factor de grupos (set) es:

$$y_{ikjl} = \mu + a_i + b_{ij} + r_{ik} + s_{il} + (rs)_{ilk} + e_{ijkl}$$

i=1,2,...,s j=1,2,..., r k=1,2,...,m l=1,2,...,f

μ: media general, a_i: efecto del i-ésimo grupo

b_{ij}: efecto de la j-ésima repetición dentro del grupo i,

r_{jk}: efecto del k-ésimo macho en el i-ésimo grupo,

s_{il}: efecto del l-ésimo macho en el i-ésimo grupo,

rs_{ilk}: interacción machos x hembras en el i-ésimo grupo, y

e_{ijlk}: error.

El resultado estadístico corresponde a un análisis de variancia y componentes de variancia para la interpretación genética.

Fuentes	Grados de libertad
Grupos	s-1
Rep(grupos)	s(r-1)
Machos(grupos)	s(m-1)
Hembras(grupos)	s(2-1) = s
Machos x hembras (grupos)	s(m-1)(2-1)= s(m-1)
Error	s(2m-1)(r-1)
Total	2smr-1

Componentes de variancia:

Var(machos) = (CM(machos(grupos)) – CM(machos x hembras(grupos)))/ (2r)

Var(machos x hembras) = (CM(machos x hembras(grupos)) – CM(error))/r

Con estas variancias se calcula las variancia aditiva y dominante.

Var(aditiva) = 4 Var(machos)

Var(dominante) = 2 Var(machos x hembras)

Análisis de cruza Línea x Probador. En un cruzamiento completo entre líneas (l) y probadores (p) se produce lxp progenies de hermanos completos. Estas progenies y sus progenitores son estudiados en un diseño de bloques completos al azar. El resultado estadístico de estos análisis debe permitir construir los análisis de variancia para progenies y cruzas para Línea x Probador, incluyendo los progenitores, efectos de habilidad combinatoria específica y general, sus errores estándar, sus componentes genéticos y su contribución con las líneas, las cruzas y su interacción con la variación total. Los tratamientos (t) de estudio conforman los padres más cruzas, donde los padres son las líneas + probadores.

Fuentes de variación y grados de libertad Padres y cruzas:

Replicaciones	r-1
Tratamientos	t-1
Padres	padres -1
Padres vs. Cruzas	1
Cruzas	cruzas -1
Error	(t-1)(r-1)
Total	rt-1

Fuentes de variación: línea x probador:

Líneas	líneas - 1
Probadores	probadores -1
Interacción	(líneas -1)(probadores-1)
Error	(t-1)(r-1)

Cuadro del análisis completo: líneas, probadores y padres:

Replicaciones	r-1
Tratamientos	t-1
Padres	padres -1
Padres vs. Cruzas	1

Cruzas	cruzas -1
Líneas	líneas - 1
Probadores	probadores -1
Línea x Probador	(líneas -1)(probadores-1)
Error	(t-1)(r-1)
Total	rt-1

Efectos de la habilidad combinatoria general:

Efectos de cada línea:

Promedio(línea) – Promedio general

Efecto de cada probador:

Promedio(probador) – Promedio general

Efectos específicos: Dados por efecto de la interacción de cada línea con cada probador.

Los errores estándares para los efectos de la habilidad combinatoria son:

$\sigma_e^2 = CM(error)$, r=rep, p=probadores, l=líneas

S.E. (gca por líinea)	$: \sqrt{\sigma_e^2/rp}$
S.E. (gca por probador)	$: \sqrt{\sigma_e^2/rl}$
S.E. (sca efecto)	$: \sqrt{\sigma_e^2/r}$
S.E. (gi - gj)línea	$: \sqrt{2\sigma_e^2/rp}$
S.E. (gi - gj)probador	$: \sqrt{2\sigma_e^2/rl}$
S.E. (sij - skl)probador	$: \sqrt{2\sigma_e^2/r}$

Componentes genéticos, covariancias:

Cov H.S. (línea): (CM(líneas) – CM(línea x probador))/(r p)

Cov H.S. (probador): (CM(probador) – CM(línea x probador))/(r p)

Cov H.S. (promedio): [(SC(línea)+SC(probador))/(l + p – 2) – CM(línea x probador)] / (r(2lp-l-p))

Cov F.S. (promedio): (CM(linea)+CM(probador)+CM(linea x probador)- 3CM(error)) / 3r + (6r – r(l+p)) Cov H.S. (promedio)/ 3r

Variancia aditiva genética:

[(1 + F)/4]^2 Var(A) = Cov H.S. (promedio)

(4) por el factor Dialelos (2x2)

Se determina Var(A) para: F = 0 y F= 1.

Variancia debido a la diferencia de la posición dominante para F=0 y F=1.

[(1 + F)/2]^2 Var(D) = Cov F.S. (promedio)

Se determina Var(D) para: F = 0 y F= 1.

Contribución proporcional de líneas, probadores y su interacción:

Contribución de líneas: SC(línea)*100/SC(cruzas)

Contribución de probadores: SC(probador)*100/SC(cruzas)

Contribución de línea x probador: SC(l x p)*100/SC(cruzas)

SC: suma de cuadrados.

Estabilidad de genotipos. Es el estudio sobre la estabilidad de diferentes genotipos en el espacio y el tiempo. La utilización de los mismos genotipos en diferentes épocas y lugares permite comparar el rendimiento u otra característica del genotipo en diferentes ambientes y ver su estabilidad. Para este estudio se consideran dos procedimientos: modelo AMMI y los modelos de estabilidad.

AMMI: Es un modelo aditivo de efectos principales e interacción multiplicativa (Weikai, 2003), ampliamente utilizado para analizar los efectos principales e interacción del genotipo y el ambiente. Se utiliza gráficos de biplot, triplot, análisis de variancia para medir el efecto de interacción y componentes principales para los gráficos. Es una herramienta de mucha utilidad para la estabilidad de los genotipos. Se requiere que los experimentos se realicen en forma similar con los mismos genotipos en diferentes ambientes. Un análisis previo de homogeneidad de variancia permite un análisis combinado. Se considera al ambiente un factor aleatorio.

El modelo planteado es:

$$Y_{ijk} = \mu + A_i + R_{ik} + G_j + AG_{ij} + E_{ijk}$$

$i= 1,2\ldots, a \quad j=1,2,\ldots,g \quad k=1,2,\ldots, r$

Y_{ijk}	: Respuesta de la unidad experimental
μ	: Media general
A_i	: Efecto del ambiente
R_{ij}	: Efecto de las repeticiones en el i-ésimo ambiente
G_j	: Efecto del genotipo
AG_{ij}	: Interacción genotipo x ambiente
E_{ijk}	: Error experimental

Las fuentes de variación y sus grados de libertad del modelo:

Ambientes	a-1
Rep(ambientes)	a(r-1)
Genotipos	g-1
Ambiente x Genotipo	(a-1)(g-1)
Error	a(r-1)(g-1)
Total	agr-1

SVD: Descomposición del valor singular, posiblemente el más utilizado por la técnica multivarial. Esta técnica fue introducida por primera vez en un documento de meteorología en 1956 por Edward Lorenz, en el que se refirió al proceso como función ortogonal empírica. Hoy en día es conocido como análisis de componentes principales. Si A es una matriz de orden (n, m), entonces la función descompone la matriz en 3 matrices:

U, D y V, donde U y V son ortogonales y D es diagonal, así que el producto permite reproducir la matriz original:

A = U D V', donde V' es la transpuesta de V.

Aplicación de la función SVD en AMMI

Con la información de los experimentos se construye una tabla de promedios de la variable observada por ambientes y genotipos. Mediante un modelo lineal se determina la magnitud del error de cada valor medio.

Y = Ambiente + Genotipo

Los errores generados del modelo forman una matriz RESIDUAL del orden número de genotipos por número de ambientes.

La matriz RESIDUAL es descompuesta mediante la función svd(), así:

svd(RESIDUAL) se descompone en U, D y V.

D contiene los valores propios en la diagonal y permite hallar las sumas de cuadrados de las componentes principales.

SC(CP i) = Dii ^2 * rep

el porcentaje de aporte de la i-ésima componente principal es: SC(CP i) *100 / Σ SC(CP i)

Para el análisis de las componentes principales se determina los grados de libertad de cada fuente mediante la relación:

gl = (gen - 1) + (amb - 1) - (2 * i - 1), para i=1 hasta el número de componentes.

Se determina los cuadrados medios (CM) como el cociente de la SC. / gl respectiva. Estas variables aleatorias se distribuyen según la distribución Chi cuadrada central con sus grados de libertad correspondientes.

La prueba estadística sobre las componentes principales se realiza mediante la prueba de F-Snedecor teniendo como denominador a la fuente de error experimental obtenida en el análisis de variancia inicial.

Para el cálculo de los valores de las componentes, se utilizan las matrices ortogonales U y V; éstas se multiplican cada una por la raíz cuadrada de los elementos de la matriz de D.

$$CP\ U = U\ D^{1/2} \text{ de dimensión (gen, n\_comp)}$$
$$CP\ V = V\ D^{1/2} \text{ de dimensión (amb, n\_comp)}$$

En ambas matrices las columnas corresponden a las componentes principales. El análisis gráfico se realiza mediante un biplot, teniendo como ejes a las componentes principales (CP1 y CP2) que son las dos primeras columnas. En el grafico se localizan los genotipos con la matriz CP U y los ambientes con matriz CP V.

En el grafico biplot se visualiza la interacción genotipo por ambiente. Ésto permite distinguir la separación o convergencia de ambientes y la correspondencia entre genotipos y ambientes. La estabilidad se percibe como la no identificación del genotipo con algún ambiente.

Modelo de estabilidad paramétrico. El modelo mas utilizados es el de Shukla y Kang (Kang, 1993) para estudiar la estabilidad del genotipo a través de sus variancias. El procedimiento calcula las estadísticas que permitan la selección según rendimiento y estabilidad. Utiliza los promedios del genotipo por ambiente a través de las diferentes repeticiones conducidas en un diseño experimental. Se asume normalidad de los errores experimentales y homogeneidad de variancia; esto significa que se requiere de un valor común para la variancia del error. El procedimiento de Shuka permite:

- Estimar la contribución de cada genotipo a la interacción GxA mediante la variancia del genotipo,
- Asignar rangos a los genotipos de mayor a menor,
- Calcula las mínimas diferencias de rendimiento LSD para las comparaciones,

- Determinar la significación e importancia de las variancias del genotipo usando las aproximaciones de la distribución F,
- Asignar una calificación de estabilidad (S): -8, -4, -2 y 0 para la variancia del genotipo según el nivel de importancia 0,01, 0,05, 0,10, niveles de probabilidad y 0 para no significativo, (a la variancia más significativa le corresponde menos estabilidad del fenotipo),
- Determinar la sumas de rangos ajustados del rendimiento (Y), y la estabilidad de clasificación (S) para cada genotipo y el estadístico de orden YS, e
- Identificar los genotipos con YS > promedio de YS.

La información requerida son los promedios de cada genotipo por ambientes, un estimado del error experimental común para todos los experimentos. Con esta información se construye un análisis de variancia conjunta con las siguientes fuentes de variación:

Total combinado	ag-1
Genotipos	g-1
Ambientes	a-1
Interacción	(g-1)(a-1)
Heterogeneidad	g-1
Residual	(g-1)(a-2)
Error común	a(g-1)(r-1)

Con la suma de cuadrados del error es estimada la σ_e^2.

Con los promedios de genotipos por ambiente (Y) se calcula las sumas de cuadrados de genotipos, ambientes y su interacción, cuya suma es el combinado genotipo-ambiente.

Yij = promedio del i-ésimo genotipo en j-ésimo ambiente

n: ambientes

k: genotipos

r: repeticiones

Estimación de la suma de cuadrados del residual

Si U es la matriz de efectos de cada genotipo en cada ambiente, entonces:

$\mu_{ij} = y_{ij} - \bar{y}_{.j}$; i=1,2, k; j=1,2,...,n

$\bar{\mu}_{.j} = \bar{y}_{.j} - \bar{y}_{..}$; Efecto del ambiente

$\overline{\mu}_{i.} = \overline{y}_{i.} - \overline{y}_{..}$; Efecto del genotipo

La suma de efectos del ambiente para cada genotipo está dada por:

$$\beta_i = \sum_{j=1}^{n}\left(\mu_{ij} - \overline{\mu}_{i.}\frac{x_j}{\sum x^2}\right)$$ Cuando se tiene una covariable X, por ejemplo precipitación promedio en el ambiente "j".

En el caso que no tener una covariable, el efecto es medido por:

$$\beta_i = \sum_{j=1}^{n}\left(\mu_{ij} - \overline{\mu}_{i.}\frac{\overline{\mu}_{.j}}{\sum \overline{\mu}_{.j}^2}\right)$$

La variancia estimada de los efectos de genotipos es corregida por el efecto ambiental S_i^2. Se determina dos variables para facilitar el cálculo. Así:

$$A_i = \sum_{j=1}^{n}(\mu_{ij} - \overline{\mu}_{i.} - \beta_i x_j)^2$$ con covariable X,

$$A_i = \sum_{j=1}^{n}(\mu_{ij} - \overline{\mu}_{i.} - \beta_i \mu_{.j})^2$$ sin covariable

$L = \sum_{i=1}^{k} A_i$; entonces:

$$S_i^2 = \frac{k}{(k-2)(n-2)}\left(A_i - \frac{L}{k(k-1)}r\right)$$

Y la suma de cuadrados residual se determina como:

$$SC(residual) = \sum_{i=1}^{k} S_i^2 \frac{(k-1)(n-2)}{k}$$

Suma de cuadrados de la heterogeneidad

Corresponde a la diferencia de la suma de cuadrados de la Interacción (genotipo, ambiente) – suma de cuadrado residual, así como sus grados de libertad.

Prueba de F

Para el cálculo del estadístico F se calcula los cuadrados medios divididos por sus grados de libertad y la relación correspondiente de los cuadrados medios para el valor F. Se considera que el factor ambiente es aleatorio y los genotipos, fijos; entonces para la prueba de F cada fuente de variación es hallada como:

Genotipos:	CM(genotipos)/CM(interacción)
Ambientes:	CM(ambientes)/CM(error)
Interacción:	CM(interacción)/CM(error)
Heterogeneidad:	CM(heterogeneidad)/CM(residual)
Residual:	CM(residual)/CM(error)

Para el cálculo de las variancias de cada genotipo σ_i^2 se determina la suma de cuadrados dentro de genotipos:

$$G_i = \sum_{j=1}^{n} (\mu_{ij} - \bar{\mu}_{i\cdot})^2$$

Entonces la variancia σ_i^2 es:

$$\sigma_i^2 = \frac{r(k(k-1)G_i - \sum G_i)}{(n-1)(k-1)(k-2)}$$

Se determina el efecto cuadrático de interacción para cada genotipo, denominado como la ecovalencia del genotipo como la suma de cuadrados de los efectos de interacción del genotipo en todos los ambientes:

$$W_i = \sum_{j=1}^{n} r(y_{ij} - \bar{y}_{i\cdot} - \bar{y}_{\cdot j} + \bar{y}_{\cdot\cdot})^2$$

Los estimadores de las medidas de estabilidad son probados estadísticamente al nivel de significación de 0.05 y 0.01. Estas medidas corresponden a cada genotipo según:

σ_i^2: la variancia de estabilidad del genotipo, grados de libertad (a-1), y

S_i^2: la variancia de estabilidad del genotipo corregido por el efecto ambiental

con grados de libertad (a-1-1). Se resta un grado de libertad por efecto de la covariable X.

Mediante la prueba de F se evalúa la estabilidad de cada genotipo. El valor de F es:

$$\frac{\sigma_i^2}{\sigma_e^2} \text{ y } \frac{S_i^2}{\sigma_e^2}$$

En ambos casos se trata de medir el efecto de estabilidad. Si es significativo, entonces el genotipo es inestable y su variabilidad, muy alta.

Selección simultánea Rendimiento-Estabilidad.

Mediante el análisis de estabilidad se determina los genotipos que tienen un comportamiento más estable. Sin embargo, el interés del investigador está también en los rendimientos máximos. La segunda parte de este análisis trata de hacer una selección considerando además la estabilidad y el rendimiento.

Con los promedios de cada genotipo $y_{i.}$, el promedio general $y_{..}$ y la diferencia mínima de significación al nivel de 0.05

$$LSD = t(0.95, glerror)\sqrt{2CM(error)/rn}$$

Se determina un valor de ajuste para los valores de rango de los genotipos:

+1 si $\bar{y}_{i.} > \bar{y}_{..}$

+2 si $\bar{y}_{i.} \geq \bar{y}_{..}$ + LSD

+3 si $\bar{y}_{i.} \geq \bar{y}_{..}$ + 2 LSD

-1 si $\bar{y}_{i.} < \bar{y}_{..}$

-2 si $\bar{y}_{i.} \leq \bar{y}_{..}$ - LSD

-3 si $\bar{y}_{i.} \leq \bar{y}_{..}$ - 2 LSD

Si ρ_i es el rango de los genotipos, entonces el rango ajustado es:

ρ_i – valor de ajuste

Calificación de la estabilidad

Se determina valores de F calculados por la razón de la variancia de estabilidad y la variancia del error dado por: $Fc = \frac{\hat{\sigma}_i^2}{\hat{\sigma}_{error}^2}$ y se les compara con los valores tabulares de significación (F con los grados de libertad respectiva)

F0 = F(1-α, n-1, n(g-1)(r-1))

F05 = F(0.95, n-1, k(n-1)(r-1))

F01 = F(0.99, n-1, k(n-1)(r-1))

Se obtiene un valor de calificación de estabilidad

Los valores de calificación son:

0 si Fc < F0; **-2** si Fc ≥ F0; **-4** si Fc ≥ F05; **-8** si Fc ≥ F01

Valor simultáneo de rango ajustado y nivel de estabilidad.

A los valores de rango ajustado para cada genotipo se les suma el valor de calificación de estabilidad, obteniendo un valor que permitirá seleccionar genotipos con más rendimiento y con aceptable estabilidad. Para decidir sobre esta selección, se utiliza como parámetro de decisión el promedio del valor de rango ajustado y nivel de estabilidad. La decisión final es: Si el valor del genotipo supera al promedio, entonces éste es considerado como genotipo de buen rendimiento y estabilidad.

Modelo de estabilidad no paramétrico. Cuando las observaciones corresponden a valores ordinales o no satisfacen las condiciones para una prueba paramétrica se debe utilizar una prueba no paramétrica. Para estabilidad se tiene la prueba de Haynes (Haynes, 1998), utilizada para evaluar el AUDPC (área bajo la curva del progreso de enfermedad) que corresponde a porcentajes de enfermedad, presentes en diferentes tiempos. Un valor que representa es el AUDPC. Se supone que estos valores no cumplen las condiciones de normalidad y tal vez de homogeneidad porque los estudios se efectúan en diferentes ambientes donde la enfermedad es muy diferente. Haynes en su artículo presenta la metodología. La información base para el estudio es el promedio del genotipo en diferentes ambientes, presentada en una matriz (genotipos x ambientes).

Para la comprensión de la metodología se utilizará la variable Y de promedios con elementos Yij , i=1,2,..., k genotipos y j=1,2,..., n ambientes y el efecto genotipo:

$$g_i = \bar{y}_{i.} - \bar{y}_{..}$$

Se calcula la matriz E de desviaciones de Y respecto del efecto genotipo:

$$e_{ij} = y_{ij} - g_i, \qquad i=1,2,..., k; \quad j=1,2,..., n.$$

Para cada ambiente "j" se determina el ranking (orden) de la respuesta de Y y de E.

Rango(Y): ρ, matriz k.n

Rango(E): ε, matriz k.n

Con la matriz ρ se determinan los promedios de los genotipos de todas sus diferencias de los rangos entre los ambientes como una medida de variación (S1) o su variancia (S2).

$$S1_i = \sum_{j=1}^{n-1} \sum_{m=j+1}^{n} \frac{\left|\rho_{ij} - \rho_{im}\right|}{n(n-1)/2}$$

$$S2_i = \sum_{j=1}^{n} \frac{(\rho_{ij} - \bar{\rho}_{i.})^2}{n-1}$$; i=1,2,...,k

Los valores esperados y variancias de las variables aleatorias S1 y S2 son:

$E[S1]=(k^2 - 1)/(3k)$

$E[S2]= (k^2 - 1)/12$

$V(S1)= k^2 ((k^2 - 4) (n + 3) + 30)/(45 k^2 n(n-1))$

$V(S2)= (k^2-1) (2(k^2- 4) (n-1) + 5 (k^2-1))/(360n (n - 1))$

Para las pruebas de hipótesis de estabilidad de los genotipos, las variables S1 y S2 se estandarizan y al elevar al cuadrado estas variables siguen una distribución de chi-cuadrada con 1 gl.

$z1 = (S1 - E[S1])^2/V(S1)$

$z2 = (S2 - E[S2])^2/V(S2)$

Los valores pueden servir para probar individualmente la estabilidad de los genotipos. Si el nivel de significación es de 0.05, el α que se considera para la prueba es α =0.05/k; así, el estadístico de la prueba chi.cuadrada es con α y un grado de libertad. En el caso de probar en conjunto si existe al menos un genotipo inestable, se suma cada variable Z1 y Z2 y mediante una prueba de chi cuadrada con α=0.05 y k grados de libertad se puede aceptar que todos son estables o que al menos existe un genotipo inestable. De aceptarse esta segunda alternativa, tienen sentido las pruebas individuales de Z1 y Z2.

El uso de estas medidas de variación es para probar la estabilidad de los genotipos. Si éstos tienen una variación significativa, entonces no son estables.

Índice de biodiversidad. Indica la presencia de especies en un área determinada. Cada índice de biodiversidad trata de expresar algo distinto como riqueza o abundancia de lo que ocurre. Existen muchos índices como: Margalef, Simpson.Dom, Simpson.Div, Berger.Parker, McIntosh, Shannon, entrte otros

(Magurrani, 1988). El índice es un valor y como tal se puede estimar. Lo importante de la estimación es encontrar un intervalo de confianza para el índice. Cuando se desconoce la distribución de probabilidades de estos índices se puede utilizar el método de Bootstrap, un método de re-muestreo.

Bootstrap aplicado. Son muchas las formas de hacer el re-muestreo; en este caso se obtiene muestras con reemplazo de la muestra observada cientos o miles de veces, lo cual permite construir una distribución empírica, mediante la cual se halla los intervalos a la confianza deseada en base a los cuantíles de la distribución (Efron, 1993).

Limite de confianza. Para un 95%, los límites serían:

LIC = cuantil((1 - 0.01 * 95)/2) y LSC= cuantil(1-(1 - 0.01 * 95)/2)

Cálculo de los índices:

n= número de especies diferentes

X=cantidades por especie

Margalef:

$$I = \frac{(n-1)}{Log(\sum x)};$$

Simpson.Dom:

$$I = \sum \left(\frac{x}{\sum x} \right)^2$$

Simpson.Div:

$$I = 1 - \sum \left(\frac{x}{\sum x} \right)^2$$

Berger.Parker:

$$I = \frac{\max(x)}{\sum x};$$

McIntosh:

$$I = \frac{\sum x - \sqrt{\sum x^2}}{\sum x - \sqrt{\sum x}}$$

Shannon:

$$I = -\sum \frac{x}{\sum x} \log \left(\frac{x}{\sum x} \right)^2$$

Consenso en dendrogramas. Medido por la magnitud en que las ramas del dendrograma mantienen su presencia. Se expresa en porcentaje; el valor indica que individuos de un agrupamiento jerárquico tienen mayor o menor porcentaje como un grado de importancia. En un estudio de clasificación multivarial, el uso de dendrogramas ayuda a la formación de grupos con similar característica. No existe prueba alguna para validar cada una de las ramas o cuántos grupos se podrían formar a partir del dendrograma. Con el consenso se puede conocer la importancia de las ramas y se tendría una mejor apreciación de los posibles grupos que se podrían formar en una clasificación jerárquica.

Bootstrap aplicado. Dado que es una sola muestra con la que se construye el dendrograma, mediante bootstrap (Efron, 1993) se puede tener muchas muestras con los mismos individuos. Para que todos los individuos participen, las variables serán sometidas al proceso de bootstrap, de tal manera que se pueda construir miles de dendrogramas y pueda contabilizarse para hallar el porcentaje de cada rama del dendrograma original (construido con la muestra observada).

Re-muestreo para modelos lineales. Las técnicas de re-muestreo involucran métodos que trabajan con muestras extraídas de la muestra original. Los métodos conocidos son Bootstrap (Efron, 1993) y Permutación (Good, 2001). Estos dos métodos se diferencian en que en bootstrap las muestras son con reemplazo, y en permutación sin reemplazo. Las técnicas de re-muestreo son aplicadas a modelos lineales para estimar la distribución de probabilidades empíricas y determinar la significancia de las fuentes de variación. El método para este proceso es permutación, asumiendo la nulidad en los parámetros. Con la permutación de los datos se puede asignar respuestas a otras unidades, generando distribuciones empíricas que permitirían calcular la probabilidad de que el parámetro es distinto de cero. El modelo propuesto debe ser lineal y la hipótesis nula (parámetros = 0) como cierta. Bajo estos supuestos se realiza la prueba estadística.

En el modelo lineal se asume una variable aleatoria "Y" dependiente de variables "X" independientes (factores o variables) lineales en los parámetros.

$$Y = f(X_1,\ldots,X_n)$$

Las fuentes de variación corresponderán a un modelo de regresión o diseño experimental en el cual se tiene una descomposición de fuentes correspondientes a los factores de estudio, así como sus interacciones.

El proceso de permutación es válido en el supuesto que todos los parámetros son cero; entonces los datos de la variable dependiente (Y) pueden ser permutados y no se alteraría el resultado con respecto a las variables independientes (X); caso contrario, se rechazaría la hipótesis de nulidad, la cual está basada en los mismos conceptos teóricos del análisis de la variancia, y las sumas de cuadrados correspondientes al producto de X e Y cambiaran constantemente por la permutación de Y. Esto permitirá generar distribuciones empíricas de las variables aleatorias correspondientes a las fuentes de variación bajo estudio (SC, CM y F).

Finalmente con los datos de la muestra original se puede realizar las pruebas estadísticas en base a las distribuciones empíricas generadas por la permutación; entonces la probabilidad (p.valor) será obtenida como:

número de casos (Fcalculado > F empírico) / k,

donde k es el número de re-muestreos y el p.valor corresponderá al nivel de significación para el rechazo de la hipótesis de nulidad del parámetro.

Montecarlo. Método de simulación más utilizado cuando la variable aleatoria no tiene una distribución conocida. En este caso, se genera una distribución empírica con los datos observados y luego se genera valores aleatorios que sigan esta distribución muestral. El procedimiento utiliza la distribución relativa acumulada (ojiva) de la función de densidad empírica. Se generar números aleatorios uniformes entre 0 y 1 y se establece una correspondencia entre el valor aleatorio y la variable mediante la ojiva, como se aprecia en la figura 3.1.

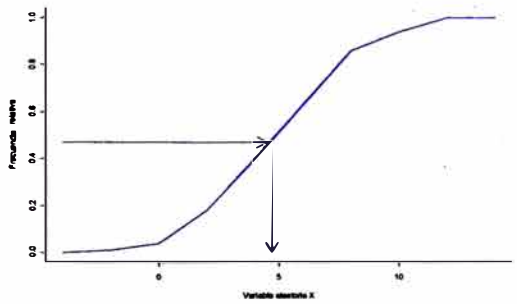


Figura 3.1 Generación por Montecarlo, uso de la ojiva

Simulación para modelos lineales bajo normalidad. El supuesto mas importante en las pruebas de hipótesis de los modelos lineales es la Normalidad. Se supone que los errores están distribuidos en forma independiente y con una variancia común. Si realmente se cumple este supuesto, las probabilidades de utilizarlo en las decisiones estadísticas son correctas. En algunas situaciones este supuesto no se cumple o está cercano a a no cumplirse porque el p.valor es cercano al 0.05. Frente a una incertidumbre, lo mejor es hacer un proceso de simulación utilizando la variancia del error del modelo, generando miles de grupos de errores normales por cada grupo, y reproducir el análisis de variancia. Se espera que más de un 50% de las veces sobre la decisión tomada para cada fuente de variación sea válido tanto en la aceptación como en el rechazo.

Métodos de aleatorización para los diseños. Los métodos utilizados por computadoras son llamado métodos de congruencia. Utilizan una semilla (valor inicial) y a partir de este valor se genera los demás valores, utilizando la operación del resto de una división.

R dispone de 8 métodos para generar números aleatorios: "Wichmann-Hill", "Marsaglia-Multicarry", "Super-Duper", "Mersenne-Twister", "Knuth-TAOCP", "user-supplied", "Knuth-TAOCP-2002" y "default". Cualquiera de estos métodos puede ser utilizado. No hay un método recomendado; por defecto no se menciona y el programa R asume el método "Super-Duper", (R Development Core Team, 2008).

El método "Super-Duper", propuesto por Marsaglia como el mejor método de congruencia multiplicativa, es definido por los parámetros 2^{32} y 69069, que es el producto 3*7*11*13*23. El método recursivo es $Xi+1 = 69069Xi \pmod{2^{32}}$; Xi es el valor inicial, $Xi+1$= resto de dividir $69069Xi$ entre 2^{32}. Así, si x1 = 73, x2 =5042037, x3 = 356102577, etc.

3.5.2 BASE DE DATOS DE LA LIBRERÍA

La investigación del CIP se orienta al tratamiento del cultivo de papa, camote y tubérculos andinos para el mejoramiento, conservación y prevención de enfermedades. El trabajo se realiza en campo e invernadero y en algunos casos con agricultores. Los tratamientos de estudio van dirigidos al control de plagas, tales como la polilla, mosca minadora, gorgojo, control de enfermedades como el tizón tardío, marchites bacteriana y virus; al mejoramiento del material genético en búsqueda de resistencia a enfermedades, tolerantes a sequías y heladas; y también mejoraramiento de la capacidad nutritiva del tubérculo. Así las áreas que producen datos para estudios estadísticos son:

Genética. Es el estudio de genotipos mediante marcadores moleculares para identificar grupos de genotipos con características similares, como por ejemplo resistencia a una enfermedad. Un caso de estudio se presenta en los datos **"markers";** estos comprenden 23 genotipos y 27 marcadores moleculares. La tabla está constituida de valores "0" y "1", que representan la ausencia y presencia del marcador en la banda del genotipo. Para otros archivos registrados esta **"pamCIP"**, con más de 40 variedades nativas de papa y sus correspondientes a 107 marcadores moleculares para identificar grupos genéticamente similares o parecidos

y posteriormente para relacionar con sus características morfológicas como las características del tubérculo en su forma, tamaño, color de piel, etc.

Biodiversidad. Es el estudio de los insectos en localidades con cultivos de papa, por ejemplo del orden de los coleópteros "escarabajos", familia CHRYOMELIDAE, y de otras órdenes y familias. Mediante los índices de biodiversidad se estudia la importancia de su presencia en el lugar. Así, se tiene los datos del lugar **"paracsho".** El estudio sobre distribución poblacional de los insectos en zonas productivas de papa permite relacionar la presencia o ausencia de insectos dañinos al cultivo.

Enfermedades. Son producidas por plagas como hongos, bacterias y virus. Las enfermedades más frecuentes son: el tizón tardío, la marchites bacteriana y enfermedades producidas por virus en la planta y tubérculos. Los estudios se realizan en campo e invernadero; algunos datos de estos estudios se presentan en el archivo **"clay",** que corresponde a estudios de suelo, la textura del suelo como el efecto de la arcilla en la disminución de la población de la bacteria Ralstonia en el tiempo, también en suelos donde se cultive papa y se presenten marchites. El uso de enmiendas (aplicación de cal y manejo con rotación de cultivos) en zonas como Carhuas y Huánuco pueden controlar de alguna manera la enfermedad. El contenido de estos datos comprende los objetos **"Hco2006"** y **"Chz2006"**. Las mediciones son porcentaje de marchites examinadas en periodos de 7 días durante dos meses, además de la observacion y rendimiento del cultivo al final de la campaña; observación de otros datos de suelo con diferentes características físico-químicas **"soil",** y el comportamiento de la bacteria. Existen estudios de mayor riesgo , tales como la enfermedad del tizón tardío en **"comasOxapampa"**. Aquí se presenta los datos de un diseño de bloques completos al azar con 3 repeticiones en parcelas de 11.9 x 18.5 m^2 con 30 cm. entre surcos y 90 cm. entre parcelas en las localidades de Mariscal Castilla y Oxapampa. La concentración de esporangios fue de 5000/ml a 49 días de la siembra. Se midió el porcentaje de infección foliar yse estimó visualmente cada 3 días durante 8 veces en Oxapampa y cada 7 días durante 12 veces en Comas. Para un análisis conjunto se calculó la variable audpc relativo de la curva de progreso de la enfermedad. Otros estudios están registrados en los datos **"wilt"** y **"cic"** también para **el** tizón tardío o "late blight" en papa; para virus, en **"PLRV"** para estudiar el efecto en papa o **"sweetpotato"** para el efecto en camote.

"Huasahuasi" contiene otros datos del cultivo de papa bajo la presencia del tizón tardío.

Recursos genéticos. Son estudios sobre del comportamiento de plantas por efecto de las sequías y heladas, sobre la estabilidad del rendimiento de las plantas en diferentes ambientes, sobre el mejoramiento de plantas en busca de nuevas variedades que generan cantidades de datos que deben ser analizados estadísticamente, así se tiene datos sobre **"LxT"** línea por probador donde se presenta el rendimiento de los padres y sus progenitores y se estudia los diferentes efectos del cruzamiento genético, tales como la variabilidad genética y los efectos aditivos generales y específicos. **"genxenv"** presenta datos para el estudio de la estabilidad en el rendimiento de papa en diferentes años y localidades, los métodos de análisis estadístico son los métodos de estabilidad y el modelo AMMI. **"Heterosis"** contiene datos de cruzamientos para ver los efectos de ganancia de las progenies en el rendimiento cuando el cultivo es sometido a estrés hídrico; se analiza mediante un modelo aditivo lineal en un análisis de variancia. Los estudios también comprenden métodos de producción, como por ejemplo los datos de **"greenhouse"**, que contienen el rendimiento por métodos de aeroponia, hidroponía, camas y macetas. Una lista completa de los datos del CIP, incorporada en la librería agricolae se presenta en el Anexo. 7

Revisión de datos de otras fuentes. En el estudio de procedimientos aplicados a la investigación agrícola, fue necesario incorporar datos provenientes de otras fuentes para mostrar las diferentes funciones estadísticas. Así, se tiene los datos de **"haynes"** para mostrar el método de estabilidad no paramétrica propuesto por Haynes en su publicación para analizar la estabilidad fenotípica al tizón tardío en 7 ambientes en EE.UU. Los datos **"corn"**, publicados por Connover (Connover, 1999), sobre el rendimiento de maíz para mostrar la aplicación de la prueba de Kruskal-Wallis. Este experimento fue completamente aleatorio. Los datos **"melon"** corresponden a datos de un experimento en cuadrado latino publicado en la Tesis: "Evaluación del sistema de riego por exudación utilizando cuatro variedades de melón, bajo modalidad de siembra, SIMPLE HILERA", de Alberto Angeles L. Universidad Agraria la Molina - Lima Peru. Datos de gramíneas **"grass"** , publicados por Connover (Connover, 1999) para mostrar la prueba de Friedman el experimento fue en bloques. Datos de arroz **"rice"**, publicado por Kwanchai (Kwanchai,1984)

para mostrar la metodología para determinar el tamaño y forma de la parcela en experimentos de campo.

Diseños y análisis más frecuentes. El CIP realiza con frecuencia experimentos de campo e invernadero. En estos últimos 4 años (2004-2008) registro en su base de datos 323 experimentos como se indica en la tabla 3.3:

Tabla 3.3. Distribución según tipo de diseño experimental en el CIP (2004-2208)

bloques	alfa	Aleatorio	Latice	split plot	strip plot	otros	Total
182	46	30	27	18	11	9	323

En invernadero desde el 2006 se realizaron 43 experimentos, de los cuales más del 90% son diseños completamente aleatorios.

En campo se utilizó el diseño de bloques completos al azar en un 56%, diseños alfa y completamente aleatorio en 14% y 9% respectivamente; pero también se utilizó otros tipos de diseño experimental que deben ser contabilizados para los análisis. Los análisis de los datos comprenden en general el análisis comparativo de tratamientos con uno o más factores. Esto significa utilizar herramientas estadísticas como el análisis de variancia, prueba de supuestos como la normalidad y homogeneidad de variancia, prueba comparativa entre tratamientos mediante pruebas paramétricas y no paramétricas y gráficos. En algunos casos, los experimentos son utilizados para fines de estabilidad, lo que requiere del uso de experimentos en diferentes ambientes.

El programa R dispone de los análisis básicos como análisis de variancia, gráficos y prueba de supuestos, pero no dispone de las pruebas comparativas entre tratamientos para variables de tipo cuantitativas y cualitativas ordinales ni los análisis de estabilidad para estos tipos de variable. Los métodos que se requiere son Kruskal-Wallis, Friedman, Waller-Duncan, LSD, Tukey y las probabilidades ajustadas.

Para estudios en recursos genéticos se requiere de herramientas para los diseños genéticos con cruzas de líneas por probador y los diseños carolina.

Para los diseños experimentales, R no dispone de ninguna función, y son necesarios los diseños de bloques, completamente aleatorios, diseños alfa, diseños de bloques incompletos o cuadrado latino, que son frecuentemente utilizados como se observa en la tabla 3.3 de la distribución de experimentos del CIP.

En biodiversidad se trabaja con dendrogramas para determinar grupos homólogos de clones registrados en el banco. El programa R dispone de los métodos para hacerlo, pero no se puede determinar el consenso, que es la importancia que tienen las ramas en el dendrograma.

3.5.3 PROCEDIMIENTOS PARA CONSTRUIR LA LIBRERÍA

Ambiente R (R Development Core Team, 2008)

Está conformado por una consola y un sistema de menú básico para el manejo administrativo de los archivos de datos y paquetes. La consola es la plataforma de R para la escritura de procedimientos; en ella se ejecuta y se presenta los resultados de lo propuesto. El proceso es interpretativo. Las instrucciones, de nivel alto. Esto permite interactuar con lenguajes de bajo nivel como el lenguaje C y FORTRAN. El lenguaje de programación utiliza instrucciones de control, repetición y secuencia, facilitando el uso de funciones matemáticas y estadísticas. Estas permiten derivar, integrar, generar números aleatorios y hallar la probabilidad para la mayor cantidad de funciones de probabilidad; también utiliza valores imaginarios, facilita la creación de funciones e interactúa con diferentes tipos de datos que se generan en un proceso como tablas, vectores, matrices, listas de objetos y otras estructuras complejas. Utiliza funciones genéricas y dependiendo del tipo de objeto se puede obtener resultados diferentes. El objeto asume un rol importante en la formación de la respuesta; así, por ejemplo en un modelo de regresión, la función plot() de R en el modelo presenta un gráfico de análisis residual. Si el objeto es resultado de un análisis cluster jerárquico, la función plot() genera un dendrograma y si el objeto es un par de vectores (x, y), es un gráfico de puntos o líneas según el caso. Los gráficos resultantes son presentados en otras ventanas especialmente para gráficos en dos o tres dimensiones; hasta puede expresarse gráficos animados movidos por el mouse. La fuente de datos para R puede ser creada en el mismo ambiente R o provenir de otros medios como dBase, Access, Excel, MySql, Oracle, textos, SAS, MINITAB, SPSS u otros programas estadísticos. Con respecto a las funciones, R

facilita la creación de librerías personales integradas por funciones, datos y documentos y con la posibilidad de ser un contribuyente al proyecto R haciendo pública la librería en R (R Development Core Team, 2008).

Preparación de los datos para la librería. Lo usual para R es que los datos estén organizados en tablas con encabezamientos por columna o una matriz. Los datos por su contenido pueden pertenecer a un tipo de variable como numérico, caracter, factor, lógico o fecha. Cada tipo de dato es reconocido y puede ser convertido en otro tipo; por ejemplo si una variable corresponde a un factor conformado por niveles, éste puede ser convertido en textos formando una variable de tipo caracter con la instrucción as.character(objeto). Si es numérico, corresponde a un factor con la orden as.factor(objeto). En el caso de fechas, el manejo está en el contexto del tiempo como días, meses, años, horas, minutos y segundos.

Previo a la creación de datos para una librería, la memoria asignada para el proceso de datos debe estar vacía de todo tipo de objetos. Se debe retirar todo objeto de la memoria con la ejecución de la instrucción: rm(list=ls()).

Se procede a la lectura de datos para formar tablas, matrices y listas de objetos según el criterio como debe ser reconocido en la librería. La lectura es directa desde la consola, invocando los datos de otros medios tales como textos, tablas de Excel, Access, etc., y darles un nombre apropiado, de preferencia en letras minúsculas y en inglés a fin de ser usados en un ambiente más amplio. Suponiendo que se tiene dos objetos, uno de camote, en texto, y otro de análisis de suelo, en Excel, el procedimiento que se sigue es:

```
> library(RODBC)
> canal <- odbcConnectExcel("suelo.xls")
> soil <- sqlFech(canal, "area")
> odbcCloseAll()
> sweetpotato <- read.table("camote.txt",header=T)
```

La librería RODBC contiene todas las funciones para importar y exporta datos de bases de datos. La función odbcConnectExcel() sirve para conectarse con Excel, versión antes del 2007, la función sqlFetch() sirve para extraer datos de un área seleccionada del Excel, y la función odbcCloseAll() para cerrar la conexión al archivo "suelo.xls".

La función read.table() es propia de la base R y permite la lectura de datos en ASCII. Con estos procedimientos se crearon dos objetos de datos: "soil" y "sweetpotato". En otros casos se junta varios objetos de datos para formar una lista. El procedimiento continúa con la creación de más objetos de datos para ser integrados en la librería.

Preparación de las funciones. El propósito de una función en la librería es facilitar los procesos en los que la función se puede utilizar cuantas veces se la invoque. En cada caso proporciona un resultado o un objeto para reutilizar en otras funciones o procedimientos. Así por ejemplo un objeto "modelo" resultante de un análisis de regresión puede ser utilizado por otras funciones para producir diferentes resultados:

Modelo de regresión,

```
> modelo <- lm(y ~ x1 + x2, data=datos)
```

Análisis de variancia,

```
> anova(modelo)
```

Grafica la línea de cuantiles para la normalidad,

```
> plot(modelo, 2)
```

Prueba la normalidad del error,

```
> shapiro.test(modelo$residuals)
```

Valores predecidos,

```
> predict(modelo).
```

Resumen del análisis de regresión,

```
> summary(modelo)
```

coeficientes del modelo,

```
> coefficients(modelo)
```

etc., muchas funciones se encargarán de sacar resultados de un solo objeto y a su vez producir nuevos resultados. La función construida debe tener la facultad de re-usar los resultados producidos por una función. Para la construcción de una función, considere el siguiente caso: Para hallar el valor tabular de la función de Waller-Duncan según el procedimiento planteado por Waller (Waller, 1976), se requiere los siguientes parámetros: grados de libertad del error, variancia estimada del error, valor de la variable F calculado en el análisis de variancia y el valor de significancía k (equivalente a los niveles de significación α). Con esta información, aplicada a la función matemática w(t) se puede hallar el valor de "t" raíz de la función. La función de w (t) tiene la siguiente configuración:

$$W(t) = k - \frac{\int^{\infty} \sqrt{x-1}\, g(x) h(tb(x)) d(x)}{\int^{\infty} \sqrt{x-1}\, g(x) h(-tb(x)) d(x)}$$

Donde:

$$g(x) = x^{(-(q+3)/2)} (f + q*F/x)^{(-(f+q-1)/2)}$$

$$b(x) = \sqrt{\frac{(q+f)(x-1)}{f*x+q*F}} \; ; \qquad h(z) = \frac{f+q+z^2}{f+q-1}c(z) + zC(z)$$

c(z) y C(z) son las funciones de densidad y acumulativa de la distribución t-student con (q+f) grados de libertad.

La solución de la ecuación w(t)=0 es el valor "t" de la tabla de Waller-Duncan. La ecuación no lineal es resuelta por el método de bisección.

El procedimiento de la función creada en R es:

```
1    waller<- function (K, q, f, Fc)
Inicio del procedimiento
2    {
Intervalos iniciales para el algoritmo de bisección
3    a0 <- 1
4    b0 <- 20
Ciclo de iteraciones de 1 a un máximo de 50
 5   for (i in 1:50) {
Cálculo del punto medio como primera aproximación a "t"
 6   t <- (b0 + a0)/2
Generación de las funciones
 7   g <- function(x, q, f, Fc) x^(-(q + 3)/2) * (f + q * Fc/x)^(-(f + q - 1)/2)
 8   b <- function(x, q, f, Fc) sqrt((f + q) * (x - 1)/(f *x + q * Fc))
 9   h <- function(z, q, f) ((f+q + z^2)/(f + q - 1))*dt(z, q + f) + z * pt(z, q+f)
10   n0 <- function(x) sqrt(x - 1) * g(x, q, f, Fc) * h(t * b(x, q, f, Fc), q, f)
11   d0 <- function(x) sqrt(x - 1) * g(x, q, f, Fc) * h(-t *b(x, q, f, Fc), q, f)
12   n1 <- function(x) sqrt(x - 1) * g(x, q, f, Fc) * h(a0 * b(x, q, f, Fc), q, f)
13   d1 <- function(x) sqrt(x - 1) * g(x, q, f, Fc) * h(-a0 * b(x, q, f, Fc), q, f)
Aplicación de las integrales hasta el valor infinito
14   IN0 <- integrate(n0, 1, Inf)$value
15   ID0 <- integrate(d0, 1, Inf)$value
16   IN1 <- integrate(n1, 1, Inf)$value
17   ID1 <- integrate(d1, 1, Inf)$value
Búsqueda del intervalo para localizar la raíz
18    if ((K - IN0/ID0) * (K - IN1/ID1) <= 0)
19   b0 <- t
20   if ((K - IN0/ID0) * (K - IN1/ID1) > 0)
21   a0 <- t
Control de proceso iterativo con un error absoluto de 5x10-4
22   if (abs(b0 - a0) <= 5e-04)
23   break
24   }
Retorno del valor aproximado a la raíz de la función waller()
25   return(round(t, 3))
26   }
```

La función waller() recibe los parámetros en su expresión como función en R, waller(k, q, f, Fc). El algoritmo de bisección requiere un intervalo donde se encuentra la raíz. Los valores tabulares de Waller-Ducan son valores positivos y pequeños

(<20), siendo un intervalo [1,20] y 50 iteraciones, como máximo, suficiente para la convergencia a la raíz. El criterio para controlar el proceso iterativo es el error absoluto menor de 0.0005.

La construcción de funciones para los diseños de experimentos es similar en cuanto a los parámetros de entrada, ya que el diseño de experimentos se fundamenta en tres principios: aleatorización, repetición y control local. La aleatorización se asegura por la generación de números aleatorios que en realidad son seudo-aleatorios puesto que utilizan procedimientos congruentes y deben tener la posibilidad de reproducir el diseño. La repetición debe especificar si es constante o variable. En el caso del diseño completo al azar las repetición puede ser variable y el control local se determina por la asignación de los tratamientos a sus correspondientes unidades experimentales según el diseño. Todas las funciones requieren los mismos parámetros:

trt: vector de tratamientos

r: vector de repeticiones

number: inicio de la numeración de unidades experimentales.

seed: un valor asignado para repetir la misma aleatorización o el valor cero para cualquier secuencia aleatoria.

kinds: criterio de aleatorización.

Así, por ejemplo para el diseño cuadrado latino, el número de repeticiones es igual al número de tratamientos; entonces los parámetros trt y r son los mismos, y sólo se requiere de: trt, number , seed y kinds.

El procedimiento de esta función en R es:

```
design.lsd <- function (trt, number = 1, seed = 0, kinds = "Super-Duper")
{
r <- length(trt)
if (seed != 0)  set.seed(seed, kinds)
a <- rep(0,r*r)
dim(a) <- c(r, r)
for (i in 1:r) {
      for (j in 1:r) {
         k <- i + j - 1
         if (k > r)
            k <- i + j - r - 1
         a[i, j] <- k
      }
}
```

```
15 m <- sample(2:r, r - 1)
16 a <- a[c(1, m), ]
17 m <- sample(1:r, r)
18 a <- a[, m]
19 trat <- trt[a]
20 columna <- rep(gl(r, 1), r)
21 fila <- gl(r, r)
22 plots <- number + 1:length(trat) - 1
23 book<-data.frame(plots, row=as.factor(fila), col=as.factor (columna), trat =
   as.factor(trat))
24 names(book)[4] <- c(paste(deparse(substitute(trt))))
25 return(book)
26 }
```

La interpretación de cada instrucción numerada es la siguiente:

(1); Define la función de nombre design.lsd, con los argumentos trt (vector de nombres de los tratamientos), number = 1 (numeración inicial de las parcelas; se asigna el valor 1 por defecto), seed = 0 (constante para iniciar la generación aleatoria, un valor distinto de cero repetirá la misma secuencia aleatoria); kinds = "Super-Duper" (método de recurrencia para la generación aleatoria).

(2) y (26): Llaves que abren y cierran la función.

(3): Inicia el número de repeticiones.

(4): Criterio para reproducir la generación aleatoria.

(5) y (6): Inicializan una matriz de orden r x r.

De la línea (7) a la (14) se construye la matriz básica de este tipo de diseño. Para el caso de 4 tratamientos sería:

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

la cual satisface los requerimientos del diseño cuadrado latino.

(15) y (16): Permutan las filas, menos la primera. Un caso aleatorio sería:

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

(17) y (18): Permuta las columnas así:

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

De la línea (19) a la (22) se asigna nombres a los tratamientos del diseño así como a las filas y columnas que identificaron a las unidades experimentales. Así, si los nombres de "trt" corresponde a las letras: a, b, c y d; entonces:

```
> trt
[1] "a" "b" "c" "d"
> trat<- trt[a]
> trat
 [1] "a" "d" "b" "c" "c" "b" "d" "a" "d" "c" "a" "b" "b" "a"
"c" "d"
> fila
 [1] 1 1 1 1 2 2 2 2 3 3 3 3 4 4 4 4
Levels: 1 2 3 4
> columna
 [1] 1 2 3 4 1 2 3 4 1 2 3 4 1 2 3 4
Levels: 1 2 3 4
> plots
 [1] 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16
```

(23): Construye la tabla de las columnas pre-calculadas formando el libro de campo.

(24): Define al vector tratamientos como un factor.

(25): Retorna el libro de campo.

Empaquetado de la librería

El empaquetado es un proceso de consolidación de las funciones, datos y la documentación del mismo en un solo producto, llamado librería. En el ambiente de la consola se debe tener todos los objetos que se van a acoplar. Para estó, es recomendable hacer el programa efectuando todos los pasos necesarios:

limpiar la consola:

> rm(list=ls())

cargar los datos. Si éstos provienen de una base de datos o de Excel, utilizar la librería RODBC, y si son datos en texto, usar la función read.table().

Si ya se realizó una vez y debe actualizar el paquete, entonces debe cargar la librería, y activar los datos como sigue:

```
library(????)
#
data(A1)
data(A2)
        ...
source("F1.R")
source("F2.R")
...
source("F3 ")

package.skeleton(name="agricolae",list=ls(), names=TRUE,
path="d:/", force=T)
```

La función package.skeleton() construye la estructura del paquete en la dirección "d", creando una carpeta de nombre "agricolae" con 3 sub-carpetas. La estructura se presenta en la figura 3.2.

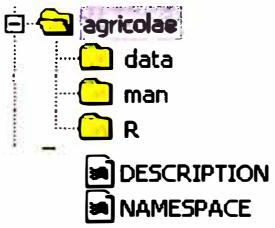


Figura 3.2 Orden de las carpetas en el explorador de Windows.

"DESCRIPTION" sin extensión es un archivo creado con un contenido básico para documentar según la descripción del paquete.
El contenido es:

```
Package: agricolae
Type: Package
Title: What the package does (short line)
Version: 1.0
Date: 2009-03-11
Author: Who wrote it
Maintainer: Who to complain to <yourfault@somewhere.net>
Description: More about what it does (maybe more than one line)
License: What license is it under?
LazyLoad: yes
```

"NAMESPACE" sin extensión es un archivo adicional creado para el uso de funciones genéricas como plot() y summary(). Por ejemplo el contenido es:

```
exportPattern("^[[:alpha:]]+")
S3method(summary,graph.freq)
S3method(plot,graph.freq)
```

- La carpeta "data" contiene todos los archivos correspondientes a los datos comprimidos con el nombre con el que fueron creados: A1, A2, etc. Un archivo es asignado para cada objeto, con la extensión "rda". Estos no pueden ser modificados.
- La carpeta "man" contiene todos los archivos de extensión "Rd" y un archivo por objeto creado (dato o programa). Cada uno de estos contiene un modelo para redactar y documentar. Los archivos están en modo ASCII y pueden ser modificados con cualquier editor de texto. El formato debe ser llenado tal y como se indica en los manuales de documentación. La sintaxis es importante para su

funcionamiento. Estos documentos son parte del sistema de ayuda del paquete en PDF y en HTML; por este motivo, la escritura sigue las reglas de LATEX.
La carpeta "R" contiene todos los programas fuente: F1, F2, etc. Todos los archivos tienen la extensión "R" y su contenido es el mismo como fue creado.

Documentación de la librería. La documentación en R sigue formatos diseñados por el proyecto R, (R Development Core Team , 2008) en la preparación inicial de la librería. Mediante la función package.sheleton se construye el prototipo de la librería.

Prototipo para una función. La documentación del prototipo tiene los elementos necesarios que debe tener toda función aplicada: un nombre, un titulo, una descripción, la forma de uso, argumentos, otros detalles, valor del argumento, referencia bibliográfica de la función, autor, algunas notas aclaratorias, relación con otras funciones, ejemplos e identificación del tipo de función para ser reconocida por todo el sistema R. Así, por ejemplo para una función "design.graeco":

```
\name{design.graeco}
\alias{design.graeco}
%- Also NEED an '\alias' for EACH other topic documented here.
\title{ ~~function to do ... ~~ }
\description{
  ~~ A concise (1-5 lines) description of what the function does. ~~
}
\usage{
design.graeco(trt1, trt2, number = 1, seed = 0, kinds = "Super-Duper")
}
%- maybe also 'usage' for other objects documented here.
\arguments{
  \item{trt1}{ ~~Describe \code{trt1} here~~ }
  \item{trt2}{ ~~Describe \code{trt2} here~~ }
  \item{number}{ ~~Describe \code{number} here~~ }
  \item{seed}{ ~~Describe \code{seed} here~~ }
  \item{kinds}{ ~~Describe \code{kinds} here~~ }
}
\details{
  ~~ If necessary, more details than the description above ~~
}
\value{
  ~Describe the value returned
  If it is a LIST, use
  \item{comp1 }{Description of 'comp1'}
  \item{comp2 }{Description of 'comp2'}
  ...
```

```
}
\references{ ~put references to the literature/web site here ~ }
\author{ ~~who you are~~ }
\note{ ~~further notes~~

 ~Make other sections like Warning with \section{Warning }{....} ~
}
\seealso{ ~~objects to See Also as \code{\link{help}}, ~~~ }
\examples{
##---- Should be DIRECTLY executable !! ----
}
% Add one or more standard keywords, see file 'KEYWORDS' in the
% R documentation directory.
\keyword{ ~kwd1 }
\keyword{ ~kwd2 }% __ONLY ONE__ keyword per line
```

Completada la documentación, se procede a la creación de la librería.

Requerimientos para la creación de la librería. En Windows, se instala los programas libres:

- ActivePerl. Se utilizó el archivo: ActivePerl-5.8.8.817-MSWin32-x86-257965.zip. El archivo se descomprime en C:\ y la carpeta interna con el nombre "perl" se copia a la raíz principal, quedando como: C:\perl.
- TOOLS, en http://www.murdoch-sutherland.com/Rtools/. El archivo descargado de Internet fue renombrado a: tools.zip. Al descomprimirlo en C:\, queda como C:\tools.
- MikTex proveniente de la web http://www.miktex.org/setup.html. Se configurará el acceso directo y las variables de ambiente: PATH=c:\tools\bin;c:\Perl\bin; TMPDIR c:\windows\temp

En ambiente DOS.

Copiar la carpeta del paquete; por ejemplo "agricolae" dentro de la carpeta BIN del folder R con toda la documentación actualizada. En la figura 3.3 se muestra el orden.

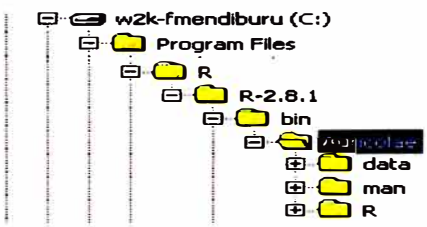


Figura 3.3. Orden de las carpetas de "agricolae"

3.5.4 CHEQUEO Y GENERADOR DE INSTALADORES

Ejecutar las siguientes órdenes en línea de comandos de DOS en el fólder BIN, asumiendo que el paquete se denomina "agricolae". Las órdenes para realizar el chequeo del paquete y la generación de los instaladores con su manual son las siguientes:

Chequeo de las funciones y datos
RCMD check agricolae
Preparación para Linux
RCMD build agricolae
Preparación para Windows
RCMD build --binary agricolae
Preparación del manual
RCMD Rd2dvi --pdf agricolae
Prueba de instalación del paquete en el entorno R
RCMD INSTALL agricolae

3.5.5 EVALUACIÓN DE LA CALIDAD DE LA LIBRERÍA

Considerando que los usuarios del programa no están presentes en forma física, la forma de encuestarlos es mediante correo electrónico e INTERNET, donde el público tenga acceso con facilidad:
http://cran.at.r-project.org/web/packages/agricolae/index.html
y en la página: http://tarwi.lamolina.edu.pe/~fmendiburu
o indirectamente a la encuesta:
http://tarwi.lamolina.edu.pe/~fmendiburu/index-filer/survey.htm.
La encuesta fue enviada por correo al personal local CIP, INIA y Universidad Agraria para que el usuario pueda opinar sobre la satisfacción de la librería agricolae. La encuesta completa se presenta en el Anexo 2. Los análisis de la encuesta serán:

Análisis de frecuencia para determinar el tipo de usuario mas frecuente.
Distribución de la antigüedad de los usuarios.
Análisis de satisfacción según la escala de Likert

La escala de Likert recomendado es de 5 niveles, del 1 al 5, de menor a mayor satisfacción. Los ítems que intervienen en el estudio son: importancia, comparación, satisfacción, uso y conformidad como se muestra en la tabla 3.4:

Tabla 3.4. Calificación según la escala de Likert

Puntaje	Importancia	Comparación	Satisfacción	Uso	Conformidad
5	Muy importante	Mejor	Muy satisfecho	Definitivamente	Totalmente de acuerdo
4	Importante	Algo mejor	Satisfecho	Probablemente	De acuerdo
3	Neutral	Es lo mismo	Neutral	No está seguro	Neutral
2	Algo importante	Algo peor	Insatisfecho	Probablemente no	En desacuerdo
1	Nada importante	No sabe	Muy insatisfecho	Definitivamente no	Totalmente en desacuerdo

- Análisis de frecuencia por tipo de función de agricolae: Diseño de experimentos, Comparaciones múltiples de tratamientos, Análisis de estabilidad, Estadística descriptiva y Simulación.
- Análisis por tipo de variable de satisfacción y determinación de los ítems más importantes para la calificación de la satisfacción. Para la selección se halla la correlacion de spearman de cada item con el puntaje total como una medida de importancia y las diferencias entre las evaluaciones altas y bajas. Los ítems selectos son los mas correlacionados y los que presentan las diferencias mas altas.
- Determinación del índice alfa de Cronbach en las variables de satisfacción de las preguntas 4, 5, 6, 12 y 15 correspondientes a 21 ítems.
- Opinión de los encuestados
- Opinión de expertos.
- Medición de la calidad en ítems representativos según el análisis de Likert con los datos de expertos.

Sobre la conformidad de la librería en el programa R, el paquete será sometido a las pruebas de chequeo por el programa "CHECK" de R, mediante la orden:
RCMD check agricolae.

Los resultados se mostrarán en un fólder con la documentación del chequeo para su evaluación. Aceptación y publicación de la librería en el proyecto R.

CAPITULO IV

ANÁLISIS Y RESULTADOS DE LA INVESTIGACIÓN

4.1 ANÁLISIS Y TRATAMIENTO DE DATOS

Para la construcción de la librería agricolae se seleccionó datos de las investigaciones agrícolas, se efectuo la programación de funciones, se documentó los datos y funciones, y se realizó el empaquetado.

4.1.1 DATOS PARA LA LIBRERÍA AGRICOLAE

Los datos utilizados en la librería corresponden a resultados de investigaciones en el Centro Internacional de la Papa, cuyas descripciones se encuentran en el Anexo 7.

CIC: Estudio de Tizón Tardío en plantas de papa de las localidades de Comas y Oxapampa, 2005. Para Comas existen 395 registros y para Oxapampa, 337.

Chz2006: Enmiendas orgánicas y rotación de cultivos para el control de la marchites Bacteriana en Carhuas, Perú en el 2006. Contiene 1620 registros para la presencia de enfermedad y 48 registros para la evaluación del rendimiento.

ComasOxapampa: 168 registros correspondientes a 53 variedades de papa resistentes al tizón tardío fueron evaluados en dos localidades de Perú Comas y Oxapampa, en el año 2002.

Glycoalkaloids: 25 registros corresponden a medidas para el glycoalcaloides de la papa. La medición se realizo con dos instrumentos: HPLC y el espectrofotómetro, para una comparación del mismo.

Hco2006: Son enmiendas orgánicas y rotación de cultivos para el control de la marchites bacteriana en Huanuco, Perú en el 2006. Contiene 1347 registros para la presencia de enfermedad y 27 registros para la evaluación del rendimiento.

RioChillon: Corresponde a clones avanzados de papa en el Valle del Rió Chillón, Perú en el 2004, en un diseño mamá-bebé. Contiene 30 registros de la parcela mamá y 90 registros de las parcelas bebés.

Clay: Población de Ralstonia (bacteria) que produce la marchites bacteriana en papa. Son 69 registros en campo.

Greenhouse: Producción de tuberculillo de papa en 3 invernaderos bajo diferentes métodos: hidroponía, aeroponía, maseta y camas. Se midió el rendimiento y el número de tuberculillo. Son 480 registros en dos invernaderos y 48 en un tercer invernadero en Lima, Perú, en el 2004.

Heterosis: Estudio genético en papa para determinación la heterosis (ganancia genética de los hijos a los padres), la capacidad combinatoria general (ACG) y específica en la materia seca, azúcares reductores y características agronómicas en familias. Contiene 324 registros.

Huasahuasi: Estudio de la vaporización de fungicidas sobre la base de los umbrales de lluvia acumulada en el control del tizón tardío en Huasahuasi, Perú, en el 2003, son dos tipos de registros correspondientes al progreso de la enfermedad en 450 registros y 45 para el rendimiento.

Markers: Resultados de marcadores moleculares para el estudio de la variabilidad en la distancia de similaridad entre 23 clones de papa.

Natives: Rendimientos de 24 papas nativas. Son 876 registros para peso y número de tubérculos.

Paracsho: Estudio de la biodiversidad en la localidad de Paracsho provincia de Tarma, departamento de Junín, Perú. Diversidad de insectos que afecta a los cultivos de papa después de la aplicación de insecticida. Son 110 registros.

Plrv: Estudio de resistencia al PLRV (Potato Leaf Roll Virus), la enfermedad es el enrollamiento de las hojas por el efecto del virus. Se experimentó con 28 genotipos en 6 localidades del Perú, siendo el total de registros 504.

Ralstonia: Registro de la población de Ralstonia solanacearum en el suelo después de 48 horas de ingestación con evaluaciones a los 15, 29, 43, 58 y 133 días en 13 localidades distintas el 2004.

Soil: Registro del análisis de suelo de las 13 localidades donde se estudió la Ralstonia en el 2004.

Sweetpotato: Experimento realizado en camote, variedad costanero en la localidad de Tacna, al sur del Perú. Se estudió el efecto de dos virus (spfmv y spcsv). Se inoculó plantas y fueron llevadas al campo. Hubo un total de 50 plantas por parcela; se estudió 4 tratamientos: los dos virus, una mezcla de ambos y un testigo. El total de parcelas fue 12.

Wilt: Estudio del avance de la enfermedad bacteria wilt (marchites bacteriana) en varias localidades en el 2004.

Yacon: (Smallanthus sonchifolius) Es una planta nativa tuberosa de los andes peruanos, que contiene una fuente natural de FOS, que es un tipo de carbohidratos con propiedades beneficiosas para el cuerpo humano.

4.1.2 FUNCIONES EN AGRICOLAE Y EJEMPLOS

La descripción completa de las funciones se encuentra en anexo 6; sus funciones, en el anexo 8; y el manual de referencia en el anexo 9.

Las funciones para diseños experimentales llevan por nombre "design", con un agregado del tipo de diseño. Así; "design.crd" corresponde al diseño completamente aleatorio, del inglés: "completely randomized design":

crd: "Completely Randomized Design"
rcbd: "Randomized Complete Block Design"
lsd: "Latin Square Design"
bib: "Balanced Incomplete Block Design"
lattice: "Lattice Design"
alpha: "Alpha Design"
cyclic: "Ciclic Design"
graeco: "Graeco-Latin square Design "
ab: "Factorial pxq in block Design"

design.crd().-Diseño completamente al azar.

Su uso es muy frecuente en ambientes controlados como laboratorio, invernadero y campo cuando el material experimental es homogéneo (uniforme). Se aplica para igual o diferente repetición y la aleatorización es completa.

Parámetros. (trt, r, number = 1, seed = 0, kinds = "Super-Duper"). "trt" para indicar el vector de tratamientos, "r" para el vector de repeticiones, "number" para el inicio de la identificación de la parcela, "seed=0" para no reproducir el mismo diseño, y un valor para generar un único plan. Corresponde a la semilla de aleatorización el método respectivo; por defecto, es "Super-Duper".

Aplicación: 5 tratamientos con 4 repeticiones:

```
> trt <- c("T1","T2","T3","T4","T5")
> r <- 4
> plan <- design.crd(trt,r,number=101,seed=5)
el plan generado es:
> plan
   plots trt r
1    101  T3 1
2    102  T2 1
...
20   120  T4 4
```

Significa que en la parcela 101 se aplicará el tratamiento 3 y corresponde a la primera repetición; la parcela 102 se aplicará el tratamiento 2 y tendrá su respectiva repetición, y así sucesivamente.

design.rcbd() - Diseño bloques completos al azar. Es Ampliamente utilizado en campo. El material es homogeneizado por bloques y se aplica con igual repetición que hace el papel de bloque, el factor de bloque es aleatorio y controlable y el factor

tratamiento es aleatoriamente distribuido en cada bloque. La función que se aplica es design.rcbd().

Parámetros. (trt, r, number = 1, seed = 0, kinds = "Super-Duper"). "trt" para indicar el vector de tratamientos; "r", el número de bloques; "number", elindicar de la identificación de la parcel; "seed=0" para no reproducir el mismo diseño, y un valor para generar un único plan. Corresponde a la semilla de aleatorización el método respectivo; por defecto, es "Super-Duper".

Aplicación: 5 tratamientos con 4 bloques:

```
> trt <- c("A1","A2","A3","A4","A5")
> r <- 4
> plan <- design.rcbd(trt,r,number=11,seed=10)
```

El plan generado:

```
   plots block trt
1     11     1  A1
2     12     1  A4
...
20    30     4  A3
```

significa que en la parcela 11 se aplicará el tratamiento A1 y le corresponde el primer bloque, y así sucesivamente.

design.lsd() - Diseño cuadrado latino. Es aplicado en campo y en la industria, utiliza dos criterios de homogeneidad por filas y columnas, llamado doble bloqueo. Los tratamientos deben estar por fila y columna aleatoriamente en forma competa, formando un cuadrado. La función por aplicar es design.rcbd().

Parámetros. (trt, number = 1, seed = 0, kinds = "Super-Duper"). "trt" para indicar el vector de tratamientos; "number" para indicar el inicio de la identificación de la parcela; "seed=0" para no reproducir el mismo diseño, y un valor para generar un único plan. "seed" corresponde a la semilla de aleatorización con el método respectivo. En este caso, por defecto es "Super-Duper".

Aplicación: 4 tratamientos:

```
> trt <- c("D1","D2","D3","D4")
> plan <- design.lsd(trt,number=111,seed=15)
```

El plan generado:

```
   plots row col trt
1    111   1   1  D1
2    112   1   2  D2
...
16   126   4   4  D4

> t(matrix(plan$trt,4,4))
```

```
     [,1] [,2] [,3] [,4]
[1,] "D1" "D2" "D4" "D3"
[2,] "D3" "D4" "D2" "D1"
[3,] "D4" "D1" "D3" "D2"
[4,] "D2" "D3" "D1" "D4"
```

significa que en la parcela 111 se aplicará el tratamiento D1 y le corresponde la primera fila y primera columna, y así sucesivamente.

design.graeco() - Diseño greco latino.- Es aplicado en campo y la industria. Utiliza dos criterios de homogeneidad por filas y columnas, Se aplica dos tipos de tratamientos, llamado doble bloqueo. Los tratamientos deben aplicarse aleatoriamente por fila y columna y en forma ortogonal; con respeto al otro tratamientos en forma completa, formando un cuadrado. La función que se aplica es design.graeco(). Por la complejidad del diseño esta función sólo es posible en tamaños específicos. La función programada permite números impares y pares (4, 8,10 y 12).

Parámetros.- (trt1, trt2, number = 1, seed = 0, kinds = "Super-Duper"). "trt1" para indicar el vector de tratamientos tipo 1; y "trt2" para el tipo 2; "number" para indicar el número de parcela de inicio; "seed=0" para no reproducir el mismo diseño, y un valor para generar un único plan. "seed" corresponde a la semilla de aleatorización con el método respectivo. En este caso, por defecto es "Super-Duper".

Aplicación: 4 tratamientos:

```
> trt1 <- c("1","2","3","4")
> trt2 <- c("A","B","C","D")
> plan <- design.graeco(trt1,trt2,number=101,seed=55)
```

El plan generado:

```
   plots row col trt1 trt2
1    101   1   1    1    C
2    102   1   2    4    D
...
16   116   4   4    1    A
> trt <- paste(plan$trt1,plan$trt2,sep="")
> t(matrix(trt,4,4))
     [,1] [,2] [,3] [,4]
[1,] "1C" "4D" "3A" "2B"
[2,] "4A" "1B" "2C" "3D"
[3,] "3B" "2A" "1D" "4C"
[4,] "2D" "3C" "4B" "1A"
```

significa que en la parcela 101 se aplicará el tratamiento tipo 1 ("1") y el tratamiento tipo 2 ("C"), y le corresponde la primera fila y primera columna, y así sucesivamente.

Para enviar el plan a Excel se ejecuta la orden:

> write.table(table, "plan.txt", sep="\t", row.names=FALSE)

Factorial pxq en diseños en bloques .

Es similar al diseño de bloques completos al azar, donde los tratamientos corresponden a una combinación de dos factores.

design.ab().-**Factorial pxq**.

Parámetros.- (A, B, r, number = 1, seed = 0, kinds = "Super-Duper"). "A" y "B" son vectores con sus niveles, y "r", los bloques.

Aplicación. Factorial 2x3 con 4 repeticiones en bloques.

> design.ab(A=1:2, B=1:3, r=4)

Diseños bloques incompletos

Estos diseños son especiales porque utilizan un número grande de tratamientos y no es posible aplicarlos en un bloque balanceado. Por ejemplo, en el campo agrícola no se puede disponer de bloques muy grandes que sean homogéneos, o en pruebas de degustación no se puede permitir que un mismo juez pruebe todos los tratamientos. Los diseños pueden ser balanceados; esto significa que cualquier par de tratamientos está presente en algún bloque un número igual de veces, y parcialmente balanceado, cuando un par de tratamientos está en algunos bloques.

design.bib() - Diseño balanceado. Es ampliamente utilizado en pruebas de degustación cuando se tiene muchos evaluadores y sólo el evaluador puede degustar un número reducido de tratamientos. La aleatorización debe ser aplicada de tal manera que el diseño sea balanceado y con un número de repeticiones igual para todos los tratamientos. La función que se aplica es design.bib().

Parámetros: (trt, k, number = 1, seed = 0, kinds = "Super-Duper"). "trt" para indicar el vector de tratamientos; el valor "k" corresponde al tamaño del semibloque, "number", al inicio de la numeración parcelaria; y "seed" al proceso aleatorio. Si "seed=0" la aleatorización es no reproducible, y un valor diferente es reproducible; "method" corresponderá al método de aleatorización. Por defecto es "Super-Duper".

Aplicación: 5 tratamientos y 3 tamaños del semi bloque:

```
> trt <- c("A","B","C","D", "E")
> plan <- design.bib(trt,k=3,number=101,seed=7)
Parameters BIB
==============
Lambda     : 3
treatmeans : 5
```

```
Block size : 3
Blocks     : 10
Replication: 6
Efficiency factor 0.8333333
<<< Book >>>
> plan
   plots block trt
1    101     1   A
2    102     1   E
3    103     1   C
...
30   130    10   C
> t(matrix(plan$trt,3,10))
      [,1] [,2] [,3]
 [1,] "A"  "E"  "C"
 [2,] "D"  "A"  "B"
 [3,] "D"  "B"  "C"
 [4,] "B"  "C"  "A"
 [5,] "D"  "B"  "E"
 [6,] "C"  "E"  "B"
 [7,] "E"  "B"  "A"
 [8,] "A"  "D"  "E"
 [9,] "C"  "D"  "A"
[10,] "E"  "D"  "C"
```

El diseño está conformado por 10 semibloques con 3 tratamientos por semibloque. Son 30 unidades experimentales (5 tratamientos por 6 repeticiones). Así, el par de tratamientos "A" y "E" se está comparando 3 veces (3 representa el valor de lambda). De igual forma, para cada par de tratamientos. Además, cada tratamiento se repite 6 veces de la misma manera para todos.

Diseños parcialmente balanceados

design.lattice(), alpha.design(), design.cyclic(). Es ampliamente utilizados en programas de mejoramiento genético. Debido al gran número de tratamientos (material genético), es imposible formar bloques completos y, por número bajo de repeticiones sólo se logra un balance parcial; es decir un par de tratamientos sólo se compara en un semibloque; las comparaciones no son ortogonales, entonces hay tratamientos que se comparan con una medida de variación menor (dentro del semibloque), y otros, con la medida de variación mayor (entre los semibloques). Esto hace que el nivel de comparación no sea igual para todos, y en la generación del diseño se busca un diseño con una mejor eficiencia relativa.

design.lattice() – Diseño latice. Los más aplicados son latice simple y triple para un número de tratamientos igual a un cuadrado perfecto: 9, 16, 36,…, etc.

Parámetros. (k, type = "triple", number = 1, seed = 0, kinds = "Super-Duper"). "k" corresponde al nivel del latice, y "type", al tipo simple o triple; las definiciones de "number", "seed" y "kinds" han sido indicadas anteriormente.

Aplicación: design.lattice(k=3, type="triple", seed=5, kinds="Super-Duper")

```
Lattice design,  triple   3 x 3
$square1
     [,1] [,2] [,3]
[1,]    6    2    8
[2,]    9    7    3
[3,]    4    1    5
$square2
     [,1] [,2] [,3]
[1,]    2    7    1
[2,]    6    9    4
[3,]    8    3    5
$square3
     [,1] [,2] [,3]
[1,]    2    9    5
[2,]    6    3    1
[3,]    8    7    4


$plan
   plots sqr block trt
1      1   1     1   6
....
27    27   3     9   4
```

Los semibloques corresponden a las filas 1 a la 3 en cada cuadrado y a una repetición. En este caso, cada tratamiento se repite 3 veces, pero la comparación del tratamiento 1 se compara con la misma eficiencia con los tratamientos 2, 3, 4, 5, 6 y 7, y en otro nivel de eficiencia con los tratamientos 8 y 9 porque se encuentran en otros semibloques.

design.alpha() – Diseño alfa. Es el más recomendado cuando no se tiene un cuadrado perfecto para el número de tratamientos; pero si requiere la formación de rectángulos de "s" filas por "k" columnas con "r" repeticiones. Cada rectángulo es una repetición y a su vez que satisface la condición de un diseño de bloques parcialmente balanceado. Para la generación se recurre a los arreglos planteados por Patterson y Williams (Patterson, 1976), quienes idearon una forma simple de generar estos diseños. Los rectángulos formados son el producto de s*k, donde "s", es el número de semibloques, y "k", el tamaño del semibloque. Esto hace que el número de tratamientos sea siempre igual a s*k y que tenga un máximo de 4 repeticiones bajo las reglas que presentan Patterson y Williams.

Parámetros: (trt, k, r, number = 1, seed = 0, kinds = "Super-Duper")

"trt" es el vector tratamientos; "k", el tamaño del bloque; "r", el número de repeticiones; "number", "seed" y "kinds" como lo señalado en otros diseños.

Aplicación: 12 tratamientos, el tamaño del bloque igual a 3, 3 repeticiones y la semilla 5 siendo igual al reproductor de la aleatorización.

design.alpha(trt=1:12, k=3, r=3, seed=5, kinds = "Super-Duper")

La salida de esta función presenta los parámetros del diseño:

```
alpha design (0,1) - Serie  III
Parameters Alpha design
=======================
treatmeans : 12
Block size : 3
Blocks     : 4
Replication: 3
Efficiency factor
(E ) 0.7096774
El diseño
$design$rep1
     [,1] [,2] [,3]
[1,] "1"  "4"  "3"
[2,] "7"  "9"  "10"
[3,] "2"  "11" "12"
[4,] "8"  "5"  "6"
$design$rep2
     [,1] [,2] [,3]
[1,] "10" "5"  "2"
[2,] "1"  "8"  "7"
[3,] "11" "3"  "6"
[4,] "4"  "9"  "12"
$design$rep3
     [,1] [,2] [,3]
[1,] "10" "1"  "6"
[2,] "7"  "5"  "12"
[3,] "11" "4"  "8"
[4,] "9"  "2"  "3"

El libro de campo,
$book
   plots cols block 1:12 replication
1      1    1     1    1           1
...
36    36    3    12    3           3
```

design.cyclic() - Diseño cíclico. Esta conformado por un gran número de semibloques, donde el factor de repeticiones no es controlable, pero debe cumplir un número igual de repeticiones por tratamiento. La generación se basa en la rotación cíclica de los tratamientos formando pequeños bloques de un número sugerido. Puede ser aplicado a estudios de degustación donde no es posible hacer un balance completo. La comparación entre tratamientos se ve afectada por la formación de los

semibloques, entonces la eficiencia cambia a los que se comparan dentro de un mismo bloque con aquellos que están en otros bloques. Estos diseños son eficientes y robustos para 6 a 30 tratamientos y con un número de repeticiones menor o igual a 10. La generación de estos diseños es viable a partir de matrices básicas propuestas por John, J.A. (Kuehl, 2000).

Parámetros: (trt, k, r, number = 1, rowcol = FALSE, seed = 0, kinds = "Super-Duper")

"trt" es el vector tratamientos; "k", el tamaño del bloque; "r", el número de repeticiones; "number", "seed" y "kinds" como lo señalado en otros diseños. A su vez el parámetro rowcol=FALSE o TRUE indica si el diseño debe cumplir la condición de un diseño latinizado (TRUE) o no (FALSE).

Aplicación. Un diseño con 8 tratamientos, con bloques de tamaño 4, 4 repeticiones, y un valor de la semilla igual a 5 para reproducir el diseño.

```
> plan <- design.cyclic(trt=letters[1:8],k=4, r=4, seed=5, kind= "Super-Duper")

cyclic design
Generator block basic:
1 2 4 8
Parameters
==================
treatmeans : 8
Block size : 4
Replication: 4
> plan$design[[1]]
     [,1] [,2] [,3] [,4]
[1,] "e"  "f"  "d"  "h"
[2,] "f"  "c"  "b"  "d"
[3,] "d"  "a"  "h"  "b"
[4,] "a"  "c"  "h"  "g"
[5,] "g"  "d"  "c"  "e"
[6,] "b"  "c"  "e"  "a"
[7,] "f"  "g"  "h"  "b"
[8,] "a"  "f"  "g"  "e"
```

Cada fila corresponde a un semi bloque de 4 tratamientos. Como se observa, cada tratamiento de los 8 se repite 4 veces.

El libro de campo generado comprende un registro de 32 líneas para la documentación de cada unidad experimental observada:

```
> plan$book
   plots group block letters[1:8]
1      1     1     1            e
2      2     1     1            f
...
32    32     1     8            e
```

Comparación múltiple de tratamientos

Los procedimientos más usados que utilizan estimación de parámetros para las pruebas son: LSD (diferencia mínima de significación) mediante t-student, con sus ajustes de probabilidad por Bonferroni; la prueba de Tukey y Waller-Duncan; y para procedimientos no-paramétricos las pruebas de Kruskal-Wallis, Friedman y Durbin. Las pruebas paramétricas requieren de la variancia y grados de libertad del error. Para ello es necesario hacer un análisis de variancia. En las pruebas no paramétricas no es necesario; el proceso es a una escala de orden (rangos), que son valores asignados a un orden según la asignación de rango estadístico, y con esta nueva escala se hace los cálculos. Las funciones programadas para estas pruebas permitirán hallar el valor del P-valor de la comparación así como la asignación de letras para agrupar tratamientos que son diferentes estadísticamente.

Para la aplicación de las pruebas paramétricas se utiliza los datos de camote incluido en la librería:

```
> library(agricolae)
> data(sweetpotato)
```

LSD.test() – Diferencia mínima de significación. Requiere del análisis de variancia para obtener los grados de libertad, y la variancia del error del experimento, dada por el cuadrado medio del error:

```
> modelo <- aov(formula, data= datos experimentales)
> gl_error <- df.residual(modelo)
> cm_error <- deviance(modelo)
```

El objeto modelo dispone de toda la información del análisis estadístico; entonces, se puede presentar el análisis de variancia, obtener los errores experimentales, hacer un análisis de residuos, hallar el coeficiente de variación, etc., con las siguientes instrucciones:

```
> anova(modelo) # realiza el análisis de variancia
> residuals(modelo) # halla los errores
> plot(modelo) # análisis grafico de residuos
> cv.model(modelo) # coeficiente de variación.
```

Para el uso de las funciones, es necesario que los datos estén organizados en una tabla por columna (respuesta y tratamientos).

Parámetros: (y, trt, DFerror, MSerror, alpha = 0.05, p.adj = c("none", "holm", "hochberg", "bonferroni", "BH", "BY", "fdr"), group = TRUE, main = NULL)

y: variable dependiente y respuesta del experimento

trt: tratamientos

DFerror: gl_error, grados de libertad del error

MSerror: CM_error. Es el estimado de la variancia del error.

alpha: nivel de significación de la prueba.

p.adj: método de ajuste de la probabilidad. El más utilizado es Bonferroni.

group: TRUE o FALSE, para indicar si se debe formar grupos de tratamientos no significativos o, en su defecto, que presente la probabilidad de rechazo de la comparación (riesgo en la comparación de un par de tratamientos, es decir, que son diferentes cuando no lo son).

main: para describir el título del análisis comparativo.

Aplicación. El experimento de camote corresponde a un diseño completamente aleatorio con un factor virus que corresponde a los tratamientos y a la variable respuesta rendimiento (yield)

```
> library(agricolae)
> data(sweetpotato)
> model<-aov(yield~virus, data=sweetpotato)
> DF<-df.residual(model)
> MSE<-deviance(model)/DF
> attach(sweetpotato)
> compara <-
 LSD.test(yield,virus,DF,MSE,main="Camote ")
> detach(sweetpotato)
```

Resultado:

```
Study: Camote

LSD t Test for yield

                          ......
Alpha                      0.050000
Error Degrees of Freedom   8.000000
Error Mean Square         22.489167
Critical Value of t        2.306004

Treatment Means
  virus    yield  std.err replication
1    cc 24.40000 2.084067           3
2    fc 12.86667 1.246774           3
3    ff 36.33333 4.233727           3
4    oo 36.90000 2.482606           3

Least Significant Difference 8.928965
Means with the same letter are not significantly different.
Groups, Treatments and means
a        oo      36.9
a        ff      36.33333
 b       cc      24.4
  c      fc      12.86667
```

Interpretación. Las letras iguales no son significativamente diferentes.

El objeto "compara" puede ser utilizado por otras funciones, dado que éste contiene información de la prueba realizada. Así, las funciones bar.group y bar.err de

agricolae son funciones complementarias y permiten mostrar los resultados gráficamente, así:

```
> bar.group(compara, density=4, main="Comparación del
efecto\ndel virus en camote", ylim=c(0,45))
> bar.err(compara, std=FALSE, col=0,main="Std error y el
rendimiento\nde camote afecto al virus", horiz=TRUE,
xlim=c(0,45))
```

Resultados, en la figura 4.1

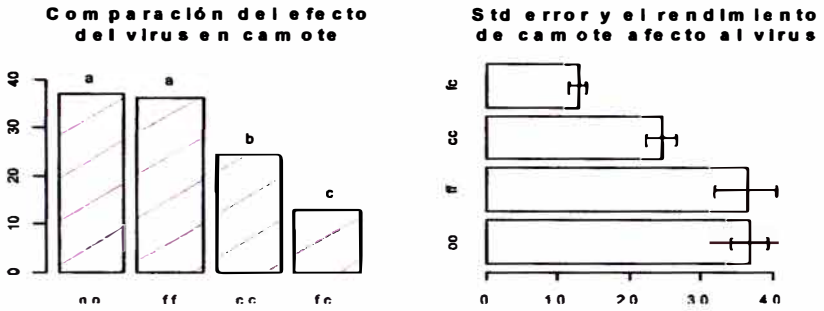


Figura 4.1 Comparación significativa y su error estándar del virus en el rendimiento de camote.

Con la opción del parámetro group=FALSE se puede hallar el p.valor que mide el riesgo de decir que son diferentes cuando no lo son. La significancia de la diferencia de los tratamientos se mide por el p.valor. Si es menor de 0.05, la diferencia es significativa.

```
> attach(sweetpotato)
> compara <- LSD.test(yield,virus,DF,MSE,main="Camote",
group=FALSE)
> detach(sweetpotato)

Study: Camote

LSD t Test for yield

                              ......
Alpha                       0.050000
Error Degrees of Freedom    8.000000
Error Mean Square          22.489167
Critical Value of t         2.306004

Treatment Means
  virus    yield  std.err replication
1    cc 24.40000 2.084067           3
2    fc 12.86667 1.246774           3
3    ff 36.33333 4.233727           3
4    oo 36.90000 2.482606           3
```

```
Comparison between treatments means

  tr.i tr.j       diff pvalue
1    1    2 11.5333333 0.0176
2    1    3 11.9333333 0.0151
3    1    4 12.5000000 0.0121
4    2    3 23.4666667 0.0003
5    2    4 24.0333333 0.0003
6    3    4  0.5666667 0.8873
```

Interpretación. Adicional a los resultados, con la opción group=FALSE se logra las probabilidades de cada comparación. Así, al virus 1 le corresponde "cc", y al virus 2, "fc". Entonces la comparación 1 vs. 2 es equivalente a "cc" vs. "fc" la diferencia es 11.53, y el p.valor es 0.017, lo cual es significativo. Entonces "cc" difiere de "fc" significativamente en el rendimiento obtenido del camote. Así también "ff" vs. "oo", y el p.valor 0.88 que no significativo, no difieren en el rendimiento de camote.

HSD.test(). Es la diferencia honesta de significación. Conocida como la prueba de Tukey, esta prueba requiere la variancia del error y su grado de libertad, el cual es obtenido del análisis de variancia.

Parámetros: (y, trt, DFerror, MSerror, alpha = 0.05, group = TRUE, main = NULL)

Estos parámetros son los mismos de la función LSD.test()

Aplicación. Para el caso de virus en camote. Los cálculos previos corresponden al error, dado por DF y MSE, que son los grados de libertad y el cuadrado medio del error. Así, el procedimiento es:

```
compara <- HSD.test(yield,virus,DF,MSE)
```

Los resultados son similares al caso LSD.test(), con las diferencias en el valor crítico y la diferencia mínima:

Critical Value of Studentized Range 4.52881

Honestly Significant Difference 12.39967

```
Groups, Treatments and means
a         oo      36.9
ab        ff      36.33333
 bc       cc      24.4
  c       fc      12.86667
```

Para el caso de group=FALSE, las comparaciones de tratamientos se presentan par a par, similar a la función LSD.test().

waller.test() – Prueba de Waller-Duncan. Esta prueba requiere el valor F calculado de los tratamientos, la variancia del error y sus grados de libertad.

Parámetros: (y, trt, DFerror, MSerror, Fc, K = 100, group = TRUE, main = NULL)

El valor "Fc" corresponde a F-calculado de tratamientos.

"K" es el nivel de gravedad del error. Para el caso k=100, es equivalente al nivel 0.05 de significación.

Aplicación. Primero se realiza el análisis de variancia como en los casos anteriores y se extrae los valores de F calculado de tratamientos, el cuadrado medio del error y sus grados de libertad:

```
Fc<-anova(model)[1,4]
compara <- waller.test(yield,virus,DF,MSE, Fc)
```

La estructura de los resultados es similar a las pruebas anteriores, con algunas diferencias en la respuesta:

```
Critical Value of Waller   2.23600
Minimum Significant Difference 8.657906
Groups, Treatments and means
a        oo      36.9
a        ff      36.33333
 b       cc      24.4
  c      fc      12.86667
```

Para el caso de group=FALSE, la respuesta es diferente. No se calcula la probabilidad pero se indica su significancia:

```
compara <- waller.test(yield,virus,DF,MSE, Fc, group=F)

  tr.i tr.j       diff significant
1    1    2 11.5333333        TRUE
2    1    3 11.9333333        TRUE
3    1    4 12.5000000        TRUE
4    2    3 23.4666667        TRUE
5    2    4 24.0333333        TRUE
6    3    4  0.5666667       FALSE
```

Significa que los tratamientos 3 y 4 no son diferentes, y corresponden a "ff" y "oo" respectivamente.

kruskal(). Realiza las comparaciones múltiples no paramétricas mediante la prueba Kruskal-Wallis.

Parámetros. (y, trt, alpha = 0.05, group = TRUE, main = NULL)

Es similar a las otras pruebas y no requiere ninguna prueba inicial.

Aplicación. Se aplica con los datos de Carhuaz del 2006 sobre enmiendas inorgánicas para el control de la marchites bacteriana.

```
> data(Chz2006)
> wilt<-Chz2006$wilt
> means <- tapply.stat(wilt[,5],wilt[,1:3],function(x)
mean(x,na.rm=TRUE))
> names(means)[4]<-"wilt_percent"
> attach(means)
> kruskal(wilt_percent, amendment, main="Marchitez")
> compara<-kruskal(wilt_percent, amendment, main="Marchitez")
```

Resultado:

```
Study: Marchitez
Kruskal-Wallis test's

Value: 31.09584
degrees of freedom: 3
Pvalue chisq  : 8.114507e-07
pKruskalWallis: 3.208616e-10

Mean of the ranks
  amendment wilt_percent replication
1        0C     41.95833          12
2        3C     20.54167          12
3      3C1Z     23.58333          12
4        6C     11.91667          12

t-Student: 2.015368
Alpha     : 0.05
LSD       : 6.729857

Means with the same letter are not significantly different

Groups, Treatments and mean of the ranks
a        0C      41.95833
 b       3C1Z    23.58333
 b       3C      20.54167
  c      6C      11.91667
```

Con la opción group=FALSE se obtiene la probabilidad en la comparación de tratamientos.

```
Comparison between treatments mean of the ranks
  tr.i tr.j      diff pvalue signf  LSD
1    1    2 21.416667 0.0000     * 6.73
2    1    3 18.375000 0.0000     * 6.73
3    1    4 30.041667 0.0000     * 6.73
4    2    3  3.041667 0.3674    ns 6.73
5    2    4  8.625000 0.0132     * 6.73
6    3    4 11.666667 0.0010     * 6.73
```

El tratamiento "OC" es el testigo, no se aplicó Cal. Los tratamientos con aplicación de cal de 3 y 6 kilos sí tienen un efecto significativo en el control de enfermedad en la planta, con un menor porcentaje de daño.

friedman(). Realiza las comparaciones múltiples no paramétricas mediante la prueba de Friedman.

Parámetros. (judge, trt, evaluation, alpha = 0.05, group = TRUE, main = NULL)

Aplicación. Se utilizará los datos de "ComasOxapampa". El análisis será comparar el efecto de la enfermedad tizón tardío en los cultivares sembrados en Comas. El diseño se realizó en bloques completos, siendo el procedimiento en R el siguiente:

```
> data(ComasOxapampa)
> attach(ComasOxapampa)
> compara <- friedman(replication, cultivar, comas, main=
main="Tizon Tardio")
> detach(ComasOxapampa)
```

Resultado:

```
Study: Tizon Tardio
Sum of the ranks
                cultivar comas replication
1          Amarilis-INIA 131.0           3
2               Andinita  64.0           3
...
56           Yayla-Kizi 158.0           3

Means with the same letter are not significantly different.


Group, Treatment and Sum of the ranks
a        Heera           167
ab       Tahuaquea       160
abc      Yayla-Kizi      158
...
    x    LBr-40            7
```

Con la opción group=FALSE se obtiene la probabilidad y significación.

```
Comparison between treatments
Sum of the ranks

     tr.i tr.j  diff pvalue signf   LSD
1       1    2  67.0 0.0000     * 32.41
2       1    3  28.5 0.0842    ns 32.41
...
1540   55   56  59.5 0.0004     * 32.41
```

durbin.test(). Realiza las comparaciones múltiples no-paramétricas mediante la prueba de Durbin para el diseño de bloques incompletos balanceado.

Aplicación. Con los datos de libro Conmover (1982), pág. 431, el siguiente procedimiento realiza la prueba:

```
> person<-gl(7,3)
> variety<-c(1,2,4,2,3,5,3,4,6,4,5,7,1,5,6,2,6,7,1,3,7)
> preference<-c(2,3,1,3,1,2,2,1,3,1,2,3,3,1,2,3,1,2,3,1,2)
> comparison<-durbin.test(person,variety,preference,
  group=TRUE, main="Seven varieties of ice cream manufacturer")
```

Resultado:

```
Study: Seven varieties of ice cream manufacturer
```

```
Sum of ranks
  variety preference
1       1          8
2       2          9
3       3          4
4       4          3
5       5          5
6       6          6
7       7          7

Durbin Test
===========
Value      : 12
Df 1       : 6
P-value    : 0.0619688
Alpha      : 0.05
Df 2       : 8
t-Student  : 2.306004

Least Significant Difference
between the sum of ranks:  2.824267

Parameters BIB
Lambda     : 1
treatmeans : 7
Block size : 3
Blocks     : 7
Replication: 3

Groups, Treatments and sum of the ranks

a        2       9
ab       1       8
abc      7       7
 bcd     6       6
  cde    5       5
   de    3       4
    e    4       3
```

Con el parámetro group=FALSE, se tiene:

```
Comparison between treatments sum of the ranks
   tr.i tr.j diff pvalue signf
1     1    2    1 0.4378    ns
2     1    3    4 0.0114     *
...
21    6    7    1 0.4378    ns
```

En ambos resultados, con o sin grupos, la información de la comparación estadística es completa. Con la información adicional del p-valor se puede ver el nivel de diferencia entre los tratamientos. En el primer caso se compara 1 vs. 2 (variety 1 vs. variety 2), el p-valor es 0.4778 y no es significativo. E en el segundo caso se compara 1 vs. 3, el p-valor es 0.0114 y es significativa la diferencia. La función

también reporta el parámetro lambda; en este caso es igual a 1 y significa que un par de tratamientos está comparado una sola vez en un determinado bloque del experimento y cada tratamiento se repite 3 veces, donde la diferencia mínima de significación de la suma de rangos entre dos tratamientos es de 2.8242.

PBIB.test().- Análisis de bloques parcialmente balanceado.

Aplicable a los diseños latices y alfa (Williams, 1977)

Parámetros. (block, trt, replication, y, k, method = "lsd", alpha = 0.05)

block: Columna que identifica a los bloques. La numeración es correlativa.

trt: Columna de tratamientos.

Replication: Columna que identifica a las repeticiones.

y: Variable respuesta

k: Tamaño del bloque

method: Método de comparación de medias.

Aplicación: sección 4.2.5, ejemplo 5.

Diseños y análisis genético

El diseño genético corresponde a la manera en que los mejoradores de plantas cruzan los genotipos y los llevan al campo o invernadero. Los tratamientos están formados por padres e hijos, y el análisis corresponde a los datos obtenidos de estos experimentos. Las funciones propuestas corresponden a los análisis.

carolina() . Análisis de los diseños carolina I, II y III

Parámetros. (model, data)

model: Valor numérico (1,2 ó 3) para indicar el tipo de diseño.

data: Tabla de datos en una estructura definida.

model=1. El orden de los datos es: set, male, female, progeny, replication, response.

model=2,3. El orden de los datos es: set, male, female, replication, response.

Set: Representa el experimento. Si son 2 set, significa que son 2 experimentos y estos se combinan en un solo análisis. male(machos), female(hembras), progenie(descendientes), replication(repetición) y response(la medida obtenida).

Aplicación. Datos (DC) aplicados al diseño carolina I.

```
> data(DC)
> carolina1 <- DC$carolina1
> output<-carolina(model=1,carolina1)
```

Resultado:

```
Response(y):  yield
Analysis of Variance Table

Response: y
                             Df  Sum Sq Mean Sq F value    Pr(>F)
set                           1  0.5339  0.5339  7.2120 0.0099144 **
set:replication               2  2.9894  1.4947 20.1914 4.335e-07 ***
set:male                      4 22.1711  5.5428 74.8743 < 2.2e-16 ***
set:male:female               6  4.8250  0.8042 10.8630 1.311e-07 ***
set:replication:male:female  10  3.2072  0.3207  4.3325 0.0002462 ***
Residuals                    48  3.5533  0.0740
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

CV: 8.286715    Mean: 3.283333
```

Este análisis de variancia presenta la siguiente información:

Set: Efecto del set

set:replication: Efecto de las repeticiones dentro de set.

set:male: Efecto de los machos dentro de set.

set:female: Efecto de las hembras dentro de set.

set:male:female: Efecto de la interacción de machos y hembras dentro de set.

set:replication:male:female: Efecto de la interacción de las repeticiones con el combinado de machos y hembras dentro de set.

Residual: Error del experimento.

El mejorador sólo toma información de los esperados cuadrados medios de las fuentes de variación para realizar cálculos en la estimación de las variancias de las diferentes componentes genéticas de machos, hembras, su aditividad y dominancia.

output[][-1]

Resultado de las variancias:

```
$var.m
`[1] 0.3948843
$var.f
[1] 0.08057407
$var.A
[1] 1.579537
$var.D
[1] -1.257241
```

var.m y var.f corresponden a las variancias de machos y hembras.

Var.A y var.D corresponden a ganancia o pérdida en variabilidad de la aditividad (A) y dominancia(D). En este caso se tiene una variabilidad aditiva de 1.579 y una pérdida de –1.25 en la variancia dominante de los machos respecto a las hembras.

lineXtester() línea por probador es el estudio del cruzamiento en búsqueda de mejoras.

Parámetros. (replications, lines, testers, y)

Corresponde a: repeticiones, líneas puras genéticamente, probadores y la variable respuesta "Y".

Aplicación. Corresponde a los estudios de heterosis (ganancia genética en los cruzamientos)

```
> data(heterosis)
> site1<-subset(heterosis,heterosis[,1]==1)
> site1<-subset(site1,site1[,4]!= "Control")
> attach(site1)
> output1<-lineXtester(Replication, Female, Male, v2)
```

Resultado:

```
ANALYSIS LINE x TESTER:  v2
ANOVA with parents and crosses
==============================
                    Df       Sum Sq     Mean Sq F value Pr(>F)
Replications         2  0.002047619 0.001023810   0.066 0.9362
Treatments          34 29.093205714 0.855682521  54.785 0.0000
Parents             10 20.937951515 2.093795152 134.055 0.0000
Parents vs. Crosses  1  1.933676421 1.933676421 123.804 0.0000
Crosses             23  6.221577778 0.270503382  17.319 0.0000
Error               68  1.062085714 0.015618908
Total              104 30.157339048

                   Df    Sum Sq    Mean Sq F value Pr(>F)
Lines               7 4.3497111 0.62138730   9.078 0.0003
Testers             2 0.9135444 0.45677222   6.673 0.0092
Lines X Testers    14 0.9583222 0.06845159   4.383 0.0000
Error              68 1.0620857 0.01561891

ANOVA for line X tester analysis including parents
==================================================
                    Df       Sum Sq     Mean Sq F value Pr(>F)
Replications         2  0.002047619 0.001023810   0.066 0.9362
Treatments          34 29.093205714 0.855682521  54.785 0.0000
Parents             10 20.937951515 2.093795152 134.055 0.0000
Parents vs. Crosses  1  1.933676421 1.933676421 123.804 0.0000
Crosses             23  6.221577778 0.270503382  17.319 0.0000
Lines                7  4.349711111 0.621387302   9.078 0.0003
Testers              2  0.913544444 0.456772222   6.673 0.0092
Lines X Testers     14  0.958322222 0.068451587   4.383 0.0000
Error               68  1.062085714 0.015618908
Total              104 30.157339048

GCA Effects:
============
Lines Effects:
Achirana  LT-8    MF-I    MF-II  Serrana    TPS-2   TPS-25    TPS-7
0.143   -0.346  -0.134   -0.169    0.394   -0.016    0.321   -0.194

Testers Effects:
TPS-13 TPS-67  TS-15
 0.104  0.052 -0.156
```

```
SCA Effects:
===========
          Testers
Lines      TPS-13 TPS-67  TS-15
  Achirana -0.010  0.002  0.008
  LT-8      0.186 -0.042 -0.144
  MF-I     -0.059  0.070 -0.011
  MF-II    -0.021  0.055 -0.034
  Serrana  -0.241  0.008  0.233
  TPS-2    -0.044  0.101 -0.057
  TPS-25    0.136 -0.259  0.123
  TPS-7     0.054  0.064 -0.118

Standard Errors for Combining Ability Effects:
==============================================
S.E. (gca for line)    : 0.04165854
S.E. (gca for tester)  : 0.02551054
S.E. (sca effect)      : 0.07215471
S.E. (gi - gj)line     : 0.05891408
S.E. (gi - gj)tester   : 0.03607736
S.E. (sij - skl)tester: 0.1020422

Genetic Components:
==================
Cov H.S. (line)    : 0.0614373
Cov H.S. (tester)  : 0.01618003
Cov H.S. (average): 0.004651843
Cov F.S. (average): 0.1221949
F = 0, Aditive genetic variance : 0.07442949
F = 1, Aditive genetic variance : 0.01860737
F = 0, Variance due to Dominance: 0.07044357
F = 1, Variance due to Dominance: 0.01761089

Proportional contribution of lines, testers
 and their interactions to total variance
==========================================
Contributions of lines  : 69.91331
Contributions of testers: 14.68349
Contributions of lxt    : 15.40320
```

Los resultados corresponden a la significancia de líneas y cruzas mediante el análisis de variancia, según la significancia estadística de las fuentes en estudio, se procede a estimar los efectos de habilidad combinatoria general y específica, sus componentes genéticos y la contribución a la variación total. Según estos indicadores se procede a la selección del material genético.

Estabilidad de los genotipos. Mediante métodos estadísticos se trata de examinar la importancia de la estabilidad en el espacio y tiempo, Para esto se dispone de tres funciones: stability.par(), stability.nonpar() y AMMI.

AMMI(). Esta función bajo un modelo aditivo, estudia la interacción genotipo-ambiente, utilizando el método de componentes principales.

Parámetros. (ENV, GEN, REP, Y, MSE = 0, number = TRUE, graph = "biplot", ...)

ENV: ambientes

GEN: genotipos

REP: repeticiones

Y: variable respuesta

MSE: el cuadrado medio del error, "0" si no se tiene.

Number: valor lógico. Si es TRUE, los genotipos son expresados por números y si es FALSE por propios nombres.

Graph: tipo de grafico es "biplot" o "triplot". Para el caso triplot se utiliza la librería klar del R.

..., significa que puede escribirse mas parámetros de la función de gráficos de R

Aplicación. Se aplica datos de "plrv"

```
> library(agricolae)
> library(klar)
> data(plrv)
> model<- AMMI(plrv[,2], plrv[,1], plrv[,3], lrv[,5], xlim=c(-
3,3),ylim=c(-4,4),graph="biplot")
> model<- AMMI(plrv[,2], plrv[,1], plrv[,3], plrv[,5], xlim=c(-
3,3),ylim=c(-4,4), graph="triplot")
```

Resultado: En texto e imagen, ver la figura 4.2.

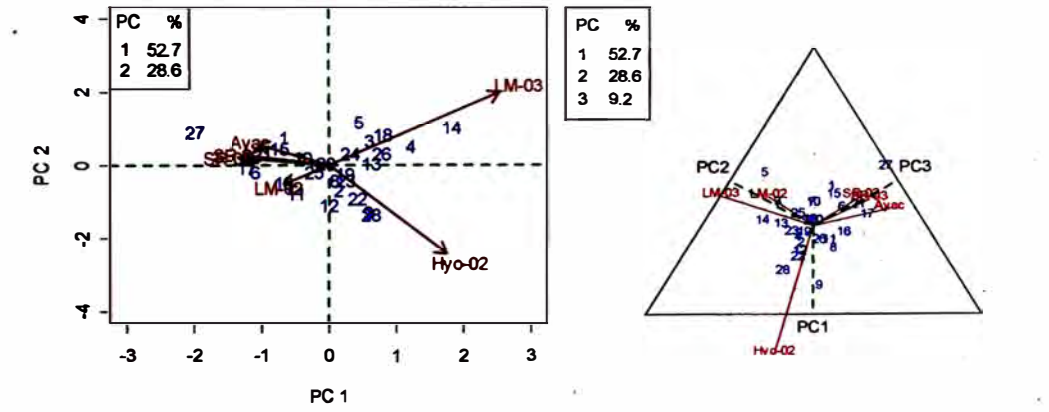


Figura 4.2 Biplot y Triplot del análisis AMMI

```
ANALYSIS AMMI:  plrv[, 5]
Class level information

ENV:  Ayac LM-02 SR-02 Hyo-02 LM-03 SR-03
GEN:  102.18 104.22 121.31 141.28 157.26 163.9 221.19 233.11
235.6 241.2 255.7 314.12 317.6 319.20 320.16 342.15 346.2
351.26 364.21 402.7 405.2 406.12 427.7 450.3 506.2 Canchan
Desiree Unica
```

```
REP:  1 2 3

Number of observations:  504

model Y: plrv[, 5]  ~ ENV + REP%in%ENV + GEN + ENV:GEN
Random effect REP%in%ENV

Analysis of Variance Table
Response: Y
           Df Sum Sq Mean Sq  F value    Pr(>F)
ENV         5 9607.4  1921.5 284.6352 4.957e-12 ***
REP(ENV)   12   81.0     6.8   2.7313   0.00154 **
GEN        27 1367.4    50.6  20.4904 < 2.2e-16 ***
ENV:GEN   135 1764.8    13.1   5.2891 < 2.2e-16 ***
Residuals 324  800.8     2.5
---
Signif.codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Coeff var       Mean plrv[, 5]
20.07525         7.831188

Analysis
   Percent acum Df    Sum.Sq   Mean.Sq F.value   Pr.F
PC1  52.7  52.7 31 929.89935 29.996753   12.14 0.0000
PC2  28.6  81.3 29 503.95903 17.377898    7.03 0.0000
PC3   9.2  90.5 27 161.61565  5.985765    2.42 0.0002
PC4   5.8  96.3 25 102.55756  4.102303    1.66 0.0264
PC5   3.8 100.1 23  66.74147  2.901803    1.17 0.2699
PC6   0.0 100.1 21   0.00000  0.000000    0.00 1.0000
```

El primer informe muestra el análisis de variancia y la variabilidad experimental (CV=20%). La interacción genotipo-ambiente debe ser significativo, para proceder al estudio de estabilidad. El análisis continua con el objeto creado "model", el cual tiene la siguiente información:

"genXenv" , matriz numérica con residuos para otros análisis.

"análisis", una tabla con los análisis de variancia de las componentes principales.

"Means", una tabla con los promedios del genotipo por ambiente.

"biplot", datos de las componentes principales.

El uso puede ser graficar una componente con la variable respuesta. Este resultado puede verse en la figura 4.3.

```
> par(cex=0.6)
> bplot<-model$biplot[,1:4]
> plot(bplot[,2],bplot$PC1,cex=0.5,xlab="Y", ylab="PC1")
> A <- subset(bplot,bplot[,1]=="ENV")
> G <- subset(bplot,bplot[,1]=="GEN")
> text(A[,2],A$PC1,labels=row.names(A),col="red")
> text(G[,2],G$PC1,labels=row.names(G),col="blue")
> abline(h=0,v= mean(bplot[,2]),lty=2,col="green")
```

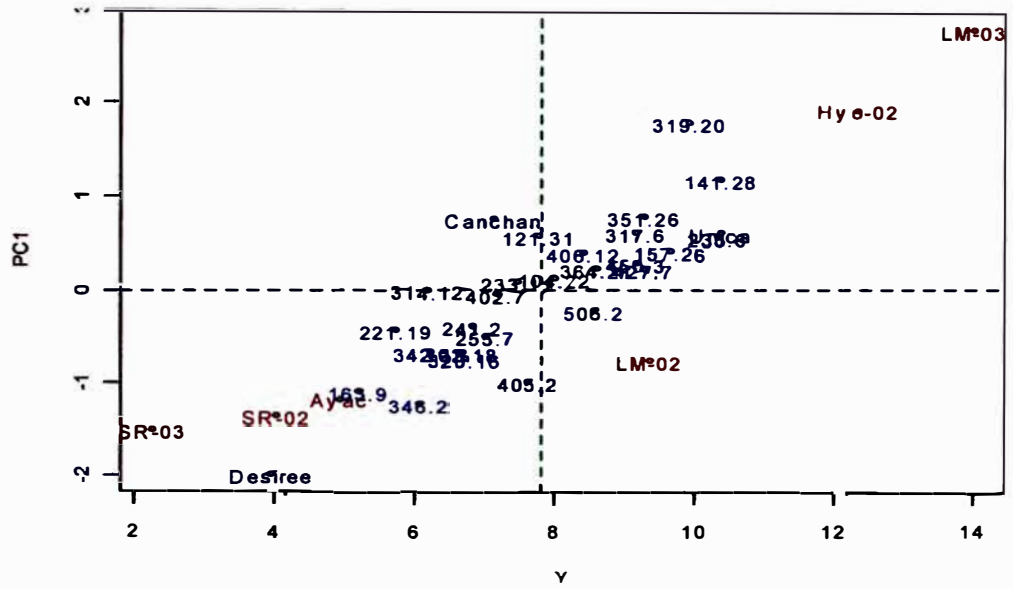


Figura 4.3 La variable respuesta y la componente principal en AMMI.

Con el objeto encontrado puede realizarce otros análisis, por ejemplo componentes principales:

```
pc<- princomp(model$genXenv, cor = FALSE)
pc$loadings

Loadings:
       Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
Ayac    0.278 -0.168  0.490  0.557 -0.422 -0.408
Hyo-02 -0.462  0.749  0.240                -0.408
LM-02   0.176  0.168 -0.823  0.264 -0.163 -0.408
LM-03  -0.669 -0.611                       -0.408
SR-02   0.322                -0.785 -0.320 -0.408
SR-03   0.355         0.114          0.831 -0.408

               Comp.1 Comp.2 Comp.3 Comp.4 Comp.5 Comp.6
SS loadings     1.000  1.000  1.000  1.000  1.000  1.000
Proportion Var  0.167  0.167  0.167  0.167  0.167  0.167
Cumulative Var  0.167  0.333  0.500  0.667  0.833  1.000

summary(pc)

Importance of components:
                          Comp.1    Comp.2      ...
Standard deviation     3.3271956 2.4493902      ...
Proportion of Variance 0.5269229 0.2855659      ...
Cumulative Proportion  0.5269229 0.8124880      ...
```

En el gráfico de AMMI se examina el comportamiento entre genotipos y ambientes. El análisis es descriptivo.

stability.par(). Método de análisis de estabilidad mediante un modelo paramétrico.

Parámetros. (data, rep, MSerror, alpha = 0.1, main = NULL, cova = FALSE, name.cov = NULL, file.cov = 0)

data: Matriz promedio de genotipos y ambientes.

rep: Constante que indica el número de repeticiones.

MSerror: Cuadrado medio del error, que es un estimado de la variancia del error experimental.

alpha: Nivel de probabilidad para hallar el valor F-Snedecor para el nivel de estabilidad en la variancia para cada genotipo.

Main: Título del estudio

Cova: Título de la covariable.

File.cov: Objeto con el contenido de la covariable, por ejemplo precipitación..

Aplicación: Con los datos publicados en el artículo de referencia (Kang, 1993) se realizó el análisis de estabilidad.

```
> V1 <- c(10.2,8.8,8.8,9.3,9.6,7.2,8.4,9.6,7.9,10,9.3,8.0,10.1,9.4,10.8,6.3,7.4)
> V2 <- c(7,7.8,7.0,6.9,7,8.3,7.4,6.5,6.8,7.9,7.3,6.8,8.1,7.1,7.1,6.4,4.1)
> V3 <- c(5.3, 4.4, 5.3, 4.4, 5.5, 4.6, 6.2, 6.0, 6.5, 5.3, 5.7, 4.4,
4.2,5.6,5.8,3.9,3.8)
> V4 <- c(7.8, 5.9, 7.3, 5.9, 7.8, 6.3, 7.9, 7.5, 7.6, 5.4, 5.6, 7.8,
6.5,8.1,7.5,5.0,5.4)
> V5 <- c(9, 9.2, 8.8, 10.6, 8.3, 9.3, 9.6, 8.8, 7.9, 9.1, 7.7, 9.5,
9.4,9.4,10.3,8.8,8.7)
> V6 <- c(6.9, 7.7, 7.9, 7.9, 7, 8.9, 9.4, 7.9, 6.5, 7.2, 5.4, 6.2,
7.2,8.8,7.3,7.1,6.4)
> V7 <- c(4.9, 2.5, 3.4, 2.5, 3,2.5, 3.6, 5.6,3.8, 3.9, 3.0, 3.0, 2.5,2.6,3.8,2.8,1.6)
> V8 <- c(6.4, 6.4, 8.1, 7.2, 7.5, 6.6, 7.7, 7.6, 7.8, 7.5, 6.0, 7.2,
6.8,7.6,6.9,7.2,7.3)
> V9 <- c(8.4, 6.1, 6.8, 6.1, 8.2, 6.9, 6.9, 9.1, 9.2, 7.7, 6.7, 7.8,
6.5,5.2,8.3,6.8,7.1)
> V10 <-c(8.7, 9.4, 8.8, 7.9, 7.8, 7.8, 11.4, 9.9, 8.6, 8.5, 8.0, 8.3,
9.1,11.0,8.1,7.8,8.0)
> V11 <-c(5.4, 5.2, 5.6, 4.6, 4.8, 5.7, 6.6, 6.8, 5.2, 4.8, 4.9, 5.4,
4.5,5.6,7.0,6.0,5.6)
> V12 <-c(8.6, 8.0, 9.2, 8.1, 8.3, 8.9, 8.6, 9.6, 9.5, 7.7, 7.6, 8.3,
6.6,9.5,9.0,9.0,8.5)
> data<-data.frame(V1,V2,V3,V4,V5,V6,V7,V8,V9,V10,V11,V12)
> rownames(data)<-LETTERS[1:17]
> stability.par(data, rep=4,MSerror=1.8,alpha=0.1, main="Genotype")
> precipitation<- c(1000,1100,1200,1300,1400,1500,1600,1700,1800,1900,2000,2100)
> stability.par(data, rep=4, MSerror=1.8, alpha=0.1, main="Genotype", cova=TRUE,
name.cov="Precipitation", file.cov=precipitation)
```

Resultado:

```
INTERACTIVE PROGRAM FOR CALCULATING SHUKLA'S STABILITY VARIANCE AND KANG'S
                      YIELD - STABILITY (YSi) STATISTICS

Genotype
Precipitation  - covariate
```

Analysis of Variance

Source	d.f.	Sum of Squares	Mean Squares	F	p.value
TOTAL	203	2964.1716			
GENOTYPES	16	186.9082	11.6818	4.17	<0.001
ENVIRONMENTS	11	2284.0116	207.6374	115.35	<0.001
INTERACTION	176	493.2518	2.8026	1.56	<0.001
HETEROGENEITY	16	4.3577	0.2724	0.09	1
RESIDUAL	160	488.8940	3.0556	1.70	<0.001
POOLED ERROR	576		1.8000		

Stability statistics

Genotype	MEANS	Sigma-square	s-square	Ecovalence
1	7.383333	2.474081 ns	2.714197 ns	25.826563
2	6.783333	1.600869 ns	1.758725 ns	17.351269
3	7.250000	0.567657 ns	0.624604 ns	7.323033
4	6.783333	2.611778 ns	2.852117 ns	27.163033
5	7.066667	1.862364 ns	2.030613 ns	19.889308
6	6.916667	3.575818 *	3.934039 *	36.519896
7	7.808333	3.580929 *	3.933173 *	36.569504
8	7.908333	2.723717 ns	2.960728 ns	28.249504
9	7.275000	4.248566 **	4.656603 **	43.049504
10	7.083333	2.273838 ns	2.473687 ns	23.883033
11	6.433333	2.560384 ns	2.791349 ns	26.664210
12	6.891667	1.558061 ns	1.713683 ns	16.935779
13	6.791667	3.483879 *	3.741592 *	35.627543
14	7.491667	5.164848 **	5.683086 **	51.942837
15	7.658333	2.380202 ns	2.611730 ns	24.915386
16	6.425000	3.445414 *	3.687730 *	35.254210
17	6.158333	3.531232 *	3.777338 *	36.087151

Signif. codes: 0 '**' 0.01 '*' 0.05 'ns' 1

Simultaneous selection for yield and stability (++)

	Genotype	Yield	Rank	Adj.rank	Adjusted	Stab.var	Stab.rating	YSi	...
1	A	7.383333	13	1	14	2.474081	0	14	+
2	B	6.783333	4	-1	3	1.600869	0	3	
3	C	7.250000	11	1	12	0.567657	0	12	+
4	D	6.783333	4	-1	3	2.611778	0	3	
5	E	7.066667	9	1	10	1.862364	0	10	+
6	F	6.916667	8	-1	7	3.575818	-4	3	
7	G	7.808333	16	2	18	3.580929	-4	14	+
8	H	7.908333	17	2	19	2.723717	0	19	+
9	I	7.275000	12	1	13	4.248566	-8	5	
10	J	7.083333	10	1	11	2.273838	0	11	+
11	K	6.433333	3	-2	1	2.560384	0	1	
12	L	6.891667	7	-1	6	1.558061	0	6	
13	M	6.791667	6	-1	5	3.483879	-4	1	
14	N	7.491667	14	1	15	5.164848	-8	7	+
15	O	7.658333	15	2	17	2.380202	0	17	+
16	P	6.425000	2	-2	0	3.445414	-4	-4	
17	Q	6.158333	1	-3	-2	3.531232	-4	-6	

Yield Mean: 7.065196
YS Mean: 6.82353
LSD (0.05): 0.4511874

\+ selected genotype
++ Reference: Kang, M. S. 1993. Simultaneous selection for yield and stability: Consequences for growers. Agron. J. 85:754-757.

Este procedimiento determina el genotipo con mejor rendimiento sobre el promedio del conjunto, siendo a su vez estables.

stability.nonpar() Método de análisis de estabilidad mediante un modelo no paramétrico para casos en que la variable observada no cumple los supuestos para un análisis paramétrico.

Parámetros. (data, variable = NULL, ranking = FALSE)

data: Tabla de datos, donde la primera columna corresponde al genotipo y las otras columnas a la respuesta en cada ambiente.

variable: nombre de la variable de estudio.

ranking: TRUE para indicar el orden de las respuestas y FALSE, para otros casos.

Aplicación. Con los datos de "CIC" se elabora una tabla de promedios, donde la primera columna corresponde a los genotipos y las otras, al ambiente. El procedimiento es el siguiente:

```
> data(CIC)
> data1<-CIC$comas[,c(1,6,7,17,18)]
> data2<-CIC$oxapampa[,c(1,6,7,19,20)]
> cic <- rbind(data1,data2)
> means <- by(cic[,5], cic[,c(2,1)], function(x) mean(x, na.rm
= TRUE))
> means <-as.data.frame(means[,])
> cic.mean<-data.frame(genotype=row.names(means),means)
> cic.mean<-delete.na(cic.mean,"greater")
> stability.nonpar(cic.mean)
```

Resultado:

```
Nonparametric Method for Stability Analysis
-------------------------------------------

Estimation and test of nonparametric measures
Variable:

Statistics...
             Mean Rank    s1   Z1      s2    Z2
762616.258   0.21   49 100.0 6.14 5000.00 11.72
762619.251   0.09    3  58.0 0.71 1682.00  0.35
Atzimba      0.27   70 106.0 7.35 5618.00 15.61
CC2.1        0.23   57  18.0 0.51  162.00  0.50
...
Yungay       0.51  107  98.0 5.76 4802.00 10.59

------------------------
Sum of Z1:  217.9763
Sum of Z2:  326.9589
------------------------

Test...
The Z-statistics are measures of stability. The test for the
significance of the sum of Z1 or Z2 are compared to a Chi-
Square value of chi.sum. individual Z1 or Z2 are compared to a
Chi-square value of chi.ind.
```

```
      MEAN       es1 es2       vs1      vs2  chi.ind  chi.sum
1 0.2425566 36.33028 990 660.1667 1372041 12.276 134.3688
---
expectation and variance: es1, es2, vs1, vs2.
```

index.bio(). Índices de biodiversidad.

Parámetros. (data, method = c("Margalef", "Simpson.Dom", "Simpson.Div", "Berger.Parker", "McIntosh", "Shannon"), level = 95, nboot = 500, console = TRUE)

Aplicación: Se aplica a los datos "paracsho" de la base agricolae.

```
> data(paracsho)
> # date 22-06-05 and treatment CON = application with
insecticide
> specimens <- paracsho[1:10,6]
> output <-
index.bio(specimens,method="Simpson.Div",level=95,nboot=200)
```

Resultado:

```
Method: Simpson.Div
The index: 0.82
95 percent confidence interval:
 0.8 ; 0.892562
```

El índice estimado de biodiversidad mediante el método Simpson.Div es de 0.82, con un intervalo de confianza al 95%. El verdadero valor del índice se encuentra entre 0.8 y 0.89.

Consenso en dendrogramas. La función utiliza los métodos de las funciones dist() y hclus() de R. La función dist() determina las distancias de disimilaridad entre individuos y la función hclust() el agrupamiento y presenta el dendrograma. Ambos métodos se aplican en la función "consensus()"y mediante el procedimiento de bootstrap se estima las coincidencias entre las ramas. Al resultado final se le llama consenso de los grupos; una función complementaria es la función hcut() de agricolae que realiza los cortes de estos diagramas.

consensus(). Consenso del dendrograma.

Parámetros. (data, distance = c("binary", "euclidean", "maximum", "manhattan", "canberra", "minkowski"), method = c("complete", "ward", "single", "average", "mcquitty", "median", "centroid"), nboot=500, duplicate=TRUE, cex.text=1, col.text = "red", ...)

"data": Corresponde a la matriz de datos formado por "n" individuos que conforman las filas y "k" columnas que son las variables.

"distance": Método para el calculo de la distancia.

"method": Método de agrupamiento.

"nboot": Número de bootstrap a realizar.

"duplicate": TRUE si debe detectar duplicados y retirarlos, continuando con individuos diferentes.

"cex.text": Tamaño del texto, 1=normal. Utiliza los criterios de R.

"col.text": color de los valores que representa el consenso.

Aplicación. Con datos de marcadores moleculares se utiliza "pamCIP" de la base agricolae.

```
> data(pamCIP)
> # only code
> rownames(pamCIP)<-substr(rownames(pamCIP),1,6)
> par(cex=0.8)
> output <-consensus( pamCIP,distance="binary",
method="complete",nboot=500)
```

Resultado: Un resumen del análisis en ASCII y el gráfico del dendrograma, con los porcentajes del consenso, tal y como se aprecia en la figura 4.4.

```
Duplicates: 18
New data  : 25 Records

Consensus hclust

Method distance: binary
Method cluster : complete
rows and cols  : 25 107
n-bootstrap     : 500
Run time        : 12.86 secs
```

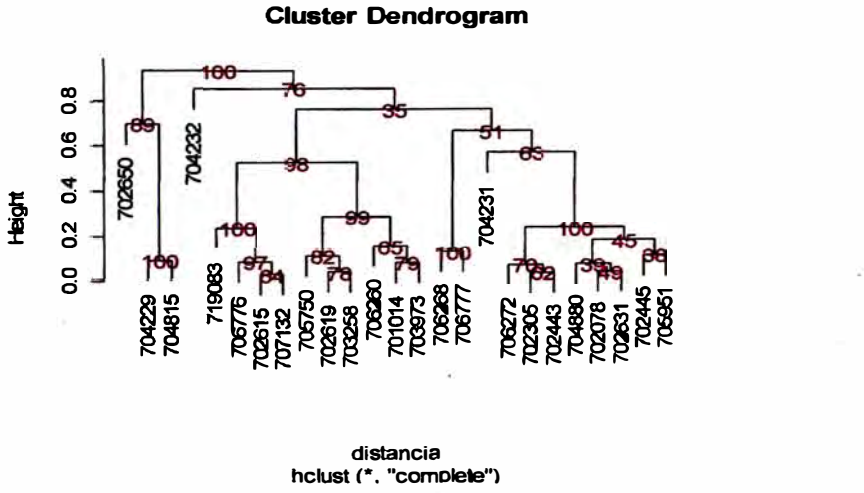


Figura 4.4. Consenso del dendrograma

Con la función hcut() de agricolae se puede extraer parte del dendrograma y mostrar sólo lo extraido:

```
> output<-consensus(pamCIP,nboot=100)
> hcut(output,h=0.4,group=5,main="Group 5")
> hcut(output,h=0.4,group=8,type="t",edgePar = list(lty=1:2,
col=2:1),main="group 8" ,col.text="blue",cex.text=1)
```

El resultado se muestra en la figura 4.5

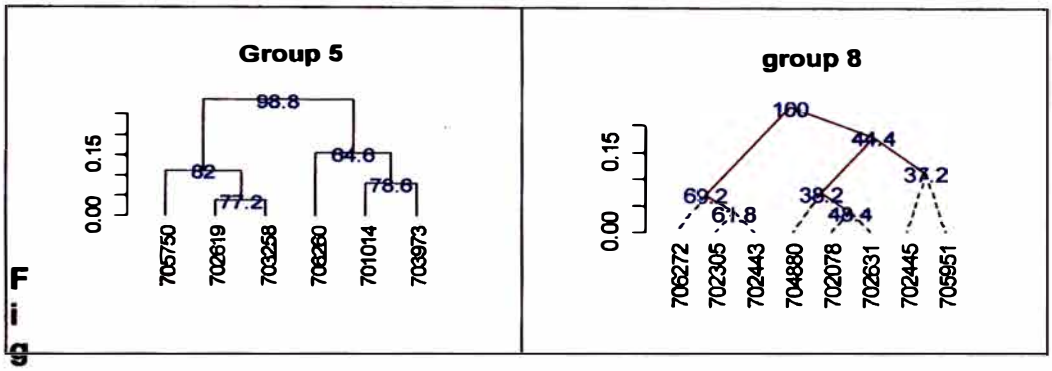


Figura. 4.5. Parte del dendrograma modo rectangular y triangular

Simulación y re-muestreo. Las funciones de simulación incluídas en agricolae son para generar datos simulados de una distribución desconocida mediante el método de Montecarlo y para validar los modelos en función de normalidad del error. La técnica de re-muestreo sirve para determinar las probabilidades del análisis de variancia de un modelo lineal.

montecarlo(). Método de simulación de montecarlo.

parámetros. (data, k, ...)

data: vector u objeto producido por hist() o graph.freq().

k: número de datos por generar

...: parámetros de la función density() sólo cuando corresponden a un vector.

Aplicación. Para esta aplicación se utilizará los datos de invernadero. Corresponde al peso del tubérculo de la variedad costanera bajo el procedimiento de aeroponia. Los datos están en la base de agricolae.

```
> data(greenhouse)
> g<-greenhouse$greenhouse1
> selecto<-subset(g,(g$variety=="Costanera" &
g$method=="aeroponic"))
> x <- montecarlo(selecto$weight,1000)
> par(mar=c(1,1,1,1))
> plot(density(selecto$weight),main="",lwd=2.5, col="red")
> lines(density(x),lty=8,lwd=2.5,col="blue")
> legend("topright",c("Obs","Sim"),lty=c(1,8),lwd=2.5,
col=c("red","blue"))
>
```

El resultado se muestra en la figura 4.6

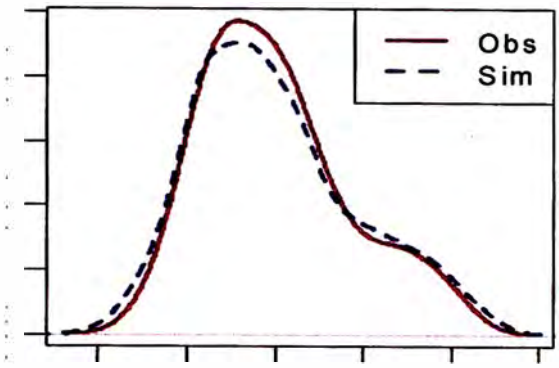


Figura 4.6. Densidad de los datos originales, simulados por montecalos.

Se puede observarr una buena coincidencia de los datos simulados por la función montercarlo() respecto a los observados.

Resultados. Son 1000 datos simulados almacenados en el objeto "x". Las estadísticas de ambos grupos son:

```
> summary(selecto$weight)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  12.90   27.95   37.00   40.15   49.23   78.60
> summary(x)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
26.020  37.290  39.570  49.870  93.460
```

Las medidas importantes del cuantil, el promedio y la desviación estándar son coincidentes.

```
> sd(selecto$weight)
[1] 16.55248
> sd(x)
[1] 17.55802
```

simulation.model(). La función permite reproducir el análisis de variancia del modelo lineal muchas veces mediante la generación normal de los residuos, permitiendo la verificación de la normalidad. Contabiliza el número de coincidencias estadísticamente. El no-cumplimiento de estas coincidencias dan la sospecha que los datos no se ajustan a la normalidad.

Parámetros. (k, file, model, categorical = NULL)

K: número de simulaciones.

file: datos.

model: modelo lineal bajo las reglas de R.

categorical: posición de las columnas de variables categóricas.

Aplicación. Se aplica a los datos del experimento de virus en camote, de la librería agricolae.

```
> data(sweetpotato)
> model<-"yield~virus"
> simulation.model(1000,sweetpotato,model,categorical=1)
```

Resultado:.

```
Simulation of experiments
Under the normality assumption
- - - - - - - - - - - - - - - - - -
Proposed model: yield~virus
Analysis of Variance Table

Response: yield
          Df  Sum Sq Mean Sq F value    Pr(>F)
virus      3 1170.21  390.07  17.345 0.0007334 ***
Residuals  8  179.91   22.49
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
---
Validation of the analysis of variancia for the proposed model
Simulations: 1000

      Df  F value % Acceptance % Rejection  Criterion
virus  3 17.34478         60.2        39.8 acceptable
```

De 1000 simulaciones el 60.2% de los experimentos simulados dió resultados similares al análisis de variancia.

resampling.model(). La función permite hallar las probabilidades de cada una de las fuentes de variación mediante distribuciones empíricas. El método supone que las variables y factores incluidos en el modelo no tienen ningún efecto en la variable respuesta; es decir, los parámetros son ceros. Mediante este análisis se puede dar la significación de cada fuente de variación sin necesidad del cumplimiento de normalidad.

Parámetros. (k, data, model).

k: número de re-muestreos.

data: datos por procesar.

model; modelo lineal según las reglas de R.

Aplicación. Se aplica a los datos del experimento de virus en camote, de la librería agricolae.

```
> data(sweetpotato)
> model<-"yield~virus"
> estudio<-resampling.model(1000,sweetpotato,model)
```

Resultado.

```
Resampling of the experiments
- - - - - - - - - - - - - - - - -
Proposed model: yield~virus
```

```
---
Resampling of the analysis of variancia for the proposed model
Determination of the P-Value by Resampling
Samples: 1000

           Df    Sum Sq   Mean Sq  F value Pr(>F) Resampling
virus       3 1170.2092 390.06972 17.34478 0.0007      0.003
Residuals   8  179.9133  22.48917
```

El análisis indica que el efecto del virus es altamente significativo. El valor de la probabilidad por re-muestreo 0.003 da confianza al resultado obtenido mediante el análisis de variancia bajo el supuesto de normalidad.

4.1.3 DOCUMENTACIÓN DE LAS FUNCIONES

Cada función tiene un documento prototipo generado por R y es un archivo de extensión ".Rd", el cual es editado en ASCII. Todos los archivos para editar se encuentran en el fólder "man". Asi se editó por ejemplo la función "design.graeco":

```
\name{design.graeco}
\alias{design.graeco}
%- design.graeco.
\title{ Graeco - latin square design}
\description{
A graeco - latin square is a KxK pattern that permits the study of k
treatments simultaneously with three different blocking variables, each  at k
levels.

The function is only for squares of the odd numbers and even numbers (4, 8,
10 and 12)
}
\usage{
design.graeco(trt1, trt2, number = 1, seed = 0, kinds = "Super-Duper")
}

\arguments{
  \item{trt1}{ Treatments }
  \item{trt2}{ Treatments }
  \item{number}{ number of first plot }
  \item{seed}{ seed }
  \item{kinds}{ method for to randomize }
}
\details{
  kinds <- c("Wichmann-Hill", "Marsaglia-Multicarry", "Super-Duper",
  "Mersenne-Twister", "Knuth-TAOCP", "user-supplied",  "Knuth-TAOCP-
2002",
```

```
  "default" )
}
\value{
  \item{trt1 }{vector, name of the treatments}
  \item{trt2 }{vector, name of the treatments}
  \item{number }{Numeric}
  \item{seed }{Numeric}
}
\references{
1. Statistics for Experimenters Design, Innovation, and Discovery
Second Edition. George E. P. Box. Wiley-Interscience. 2005.

2. Experimental design. Cochran and Cox. Second edition.
Wiley Classics Library Edition published 1992.
}
\author{ Felipe de Mendiburu }
\seealso{\code{\link{design.crd}}, \code{\link{design.lsd}},
\code{\link{random.ab}},
\code{\link{fact.nk} } }

\examples{
library(agricolae)
T1<-c("a","b","c","d")
T2<-c("v","w","x","y")
graeco <-  design.graeco(T1,T2,number=101)
plots <-as.numeric(graeco[,1])
trt <- paste(graeco[,4],graeco[,5])
dim(plots)<-c(4,4)
dim(trt) <-c(4,4)
print(t(plots))
print(t(trt))
# 10 x 10
T1 <- letters[1:10]
T2 <- 1:10
graeco <-  design.graeco(T1,T2)
trt <- paste(graeco[,4],graeco[,5])
dim(trt) <-c(10,10)
print(t(trt))
}
\keyword{ design }
```

Este procedimiento se realizó para cada una de las funciones de agricolae.

4.1.4 DOCUMENTACIÓN DE LOS DATOS

Cada dato tiene un documento prototipo generado por R, el cual fue editado. Los archivos se encuentran en el fólder "man" y tienen la extensión ".Rd". La edición es en ASCII. Así por ejemplo, los datos "sweetpotato":

```
\name{sweetpotato}
\alias{sweetpotato}
\docType{data}
\title{ Data of sweetpotato yield }
\description{
 The data correspond to an experiment with costanero
sweetpotato made at the locality
 of the Tacna department, southern Peru. The effect of two
viruses (Spfmv and Spcsv)
 was studied. The treatments were the following: CC (Spcsv) =
Sweetpotato chlorotic
 dwarf, FF (Spfmv) = Feathery mottle, FC (Spfmv y Spcsv) =
Viral complex and OO
 (witness) healthy plants.
 In each plot, 50 sweetpotato plants were sown and 12 plots
were employed.
 Each treatment was made with 3 repetitions and at the end of
the experiment the total
 weight in kilograms was evaluated. The virus transmission was
made in the cuttings
 and these were sown in the field.
}
\usage{data(sweetpotato)}
\format{
  A data frame with 12 observations on the following 2
variables.
  \describe{
    \item{\code{virus}}{a factor with levels \code{cc}
\code{fc} \code{ff} \code{oo}}
    \item{\code{yield}}{a numeric vector}
  }
}
\source{
  Experimental field.
}
\references{
  International Potato Center. CIP - Lima Peru
}
\examples{
library(agricolae)
data(sweetpotato)
str(sweetpotato)
}
\keyword{datasets}
```

Este procedimiento se realizó para cada uno de los datos de agricolae.

4.1.5 EMPAQUETADO DE LAS FUNCIONES

Para este proceso se dispuso las funciones y los datos en la consola de R. La primera vez se hizo la lectura de medios externos, de archivos en ASCII y Excel; posteriormente se realizó la lectura de la misma librería en su versión anterior. La preparación se realiza en el ambiente R. En la consola se dispone de todos los objetos que se va a juntar; en este paso se puede agregar más datos y funciones o actualizar los mismos.

Para generar la nueva versión, se utilizó la versión 1.0-6. Los datos ya están disponibles y estos se cargan de la librería mediante las siguientes instrucciones:

```
library(agricolae)
# )

data(Chz2006)
data(CIC)
data(ComasOxapampa)
data(Glycoalkaloids)
data(LxT)
data(RioChillon)
          ...
data(yacon)
data(wilt)
# las fuentes de la libreria estan en una carpeta en H:
setwd("H:/Agricolae/R")
# cargar las fuentes
source("AMMI.contour.R")
source("AMMI.R")
source("audpc.R")
source("bar.err.R")
source("bar.group.R")
source("BIB.test.R")
...
source("graph.freq.R")
source("summary.graph.freq.R")
source("plot.graph.freq.R")
...
source("waller.test.R")
source("wxyz.R")
# Empaquetar
package.skeleton(name="agricolae",list=ls(), names=TRUE,
path="d:/", force=T)
```

En la nueva versión, se incluyó funciones genéricas para la función graph.freq, tales como summary.graph.freq() y plot.graph.freq(), y así, es posible utilizar las funciones summary() y plot():

```
> h <- graph.freq(x)
> summary(h)
> plot(h)
```

La función package.skeleton() de R crea la estructura del paquete en la dirección "d" en la carpeta "agricolae" con 3 sub-carpetas. La estructura en la figura 4.7.

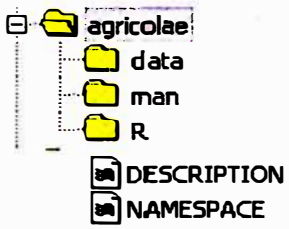


Figura 4.7. Orden de las carpetas en el explorador de Windows

"DESCRIPTION" en un archivo ASCII sin extensión. Este archivo es creado automáticamente y contiene el prototipo de la descripción del paquete.

El contenido del prototipo es el siguiente:

```
Package: agricolae
Type: Package
Title: What the package does (short line)
Version: 1.0
Date: 2009-03-11
Author: Who wrote it
Maintainer: Who to complain to <yourfault@somewhere.net>
Description: More about what it does (maybe more than one line)
License: What license is it under?
LazyLoad: yes
```

"NAMESPACE" es el archivo sin extensión, creado para el uso de funciones genéricas como plot() y summary(); su contenido fue:

```
exportPattern("^[[:alpha:]]+")
S3method(summary,graph.freq)
S3method(plot,graph.freq)
```

- La carpeta "data" contiene todos los archivos correspondientes a los datos comprimidos con su nombre original de creación, siendo un archivo para cada objeto con la extensión "rda' en forma codificada. Estos archivos no se pueden modificar.
- el manual de referencia en PDF. La carpeta "man" contiene todos los archivos de extensión ".Rd", siendo un archivo por cada objeto creado (dato o programa). Cada archivo contiene un prototipo para editar y documentar. Los archivos están en modo ASCII. Según el formato, fue llenado exactamente como se indica en

los manuales de documentación de R, dado que estos documentos son parte del sistema de ayuda de la librería en HTML y

- La carpeta "R" contiene todos los programas fuentes, todos los archivos tienen la extensión ".R"; su contenido es el mismo creado por el autor.

Requerimientos para el empaquetado.

En Windows se instaló los programas libres:

- ActivePerl. Se utilizó el archivo ActivePerl-5.8.8.817-MSWin32-x86-257965.zip. El archivo se descomprime en C:\ y la carpeta interna, con el nombre "perl", se copia a la raíz principal: C:\perl
- TOOLS de la web http://www.murdoch-sutherland.com/Rtools/ El archivo que se utilizó fue tools.zip; el nombre en Internet es muy similar ya que está identificado con la versión. Se renombró a "tools.zip" y se descomprimió en C:\, quedando como C:\tools.
- MikTex de la web http://www.miktex.org/setup.html.
- Se agregó las rutas de acceso directo y variable de ambiente:
  PATH=c:\tools\bin;c:\Perl\bin;
  TMPDIR c:\windows\temp

Ambiente DOS.

En la carpeta BIN de R se copió la carpeta agricolae con todos sus directorios. En la figura 4.8 se muestra su ubicación.

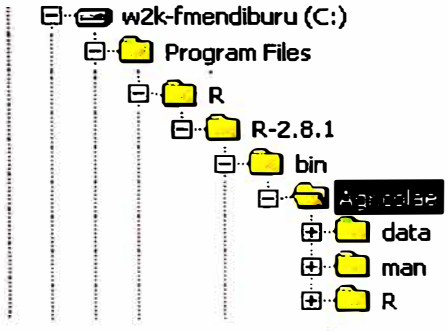


Fig. 4.8. Orden de las carpetas de agricolae en BIN de R

Se ejecutó las siguientes órdenes en línea de comandos de DOS.

Chequeo de funciones y datos antes de enviar al proyecto R

RCMD check agricolae

4.2 RESULTADOS

Según el modelo McCall, la librería fue sometida al proceso de chequeo de su operatividad y aceptación por el proyecto R. Su calidad fue evaluada mediante la encuesta de satisfacción y confrontación con otros programas estadísticos.

4.2.1 CHEQUEO DE LA LIBRERÍA CON R

Para este proceso se utilizó la versión de R más reciente 2.9.0 disponible desde el 17 de abril del 2009.

En la ventana de comandos de DOS, en la carpeta: C:\Program Files\R\R-2.9.0\bin, se ejecutó la siguiente orden:

RCMD check agricolae

siendo el resultado el siguiente:

```
(C) Copyright 1985-2001 Microsoft Corp.
C:\Documents and Settings\Felipe>cd\progra~1\r\R-2.9.0\bin
C:\PROGRA~1\R\R-2.9.0\bin>RCMD check agricolae
* checking for working pdflatex ... OK
* using log directory 'C:/PROGRA~1/R/R-2.9.0/bin/agricolae.Rcheck'
* using R version 2.9.0 (2009-04-17)
* using session charset: ISO8859-1
* checking for file 'agricolae/DESCRIPTION' ... OK
* checking extension type ... Package
* this is package 'agricolae' version '1.0-7'
* checking package name space information ... OK
* checking package dependencies ... OK
* checking if this is a source package ... OK
* checking for .dll and .exe files ... OK
* checking whether package 'agricolae' can be installed ... OK
* checking package directory ... OK
* checking for portable file names ... OK
* checking DESCRIPTION meta-information ... OK
* checking top-level files ... OK
* checking index information ... OK
* checking package subdirectories ... OK
* checking R files for non-ASCII characters ... OK
* checking R files for syntax errors ... OK
* checking whether the package can be loaded ... OK
* checking whether the package can be loaded with stated dependencies ...
OK
* checking whether the name space can be loaded with stated dependencies
... OK
* checking for unstated dependencies in R code ... OK
* checking S3 generic/method consistency ... OK
* checking replacement functions ... OK
```

```
* checking foreign function calls ... OK
* checking R code for possible problems ... OK
* checking Rd files ... OK
* checking Rd files against version 2 parser ... OK
* checking Rd cross-references ... OK
* checking for missing documentation entries ... OK
* checking for code/documentation mismatches ... OK
* checking Rd \usage sections ... OK
* checking data for non-ASCII characters ... OK
* checking examples ... OK
* checking PDF version of manual ... OK
```

Este proceso genera una carpeta con toda la documentación del proceso de chequeo. La ubicación de las carpetas y los archivos se presenta en la figura 4.9.

En la carpeta "agricolae.Rcheck" se crea los siguientes archivos:

"00check.log". Es ASCII y contiene la información del chequeo.

"00install.out". Es ASCII y contiene las llamadas de atención respecto a la instalación. Por ejemplo compatibilidad con el HTML.

"agricolae-Ex.ps". Documento generado en PostScrip para el tratamiento de los ejemplos en la documentación de ayudas y el manual.

"agricolae-Ex.R". Procedimiento para ejecutar los ejemplos de agricolae.

"agricolae-Ex.Rout". Contiene los resultados de los ejemplos en ASCII.

"agricolae-manual.log". Chequeo en la construcción del manual.

"agricolae-manual.pdf". Manual creado en PDF.

"R.css".- Informa sobre los parámetros incorporados en el sistema de ayuda. Puede ser visto con FrontPage.

Según los archivos con información del chequeo, agricolae paso todas las pruebas.

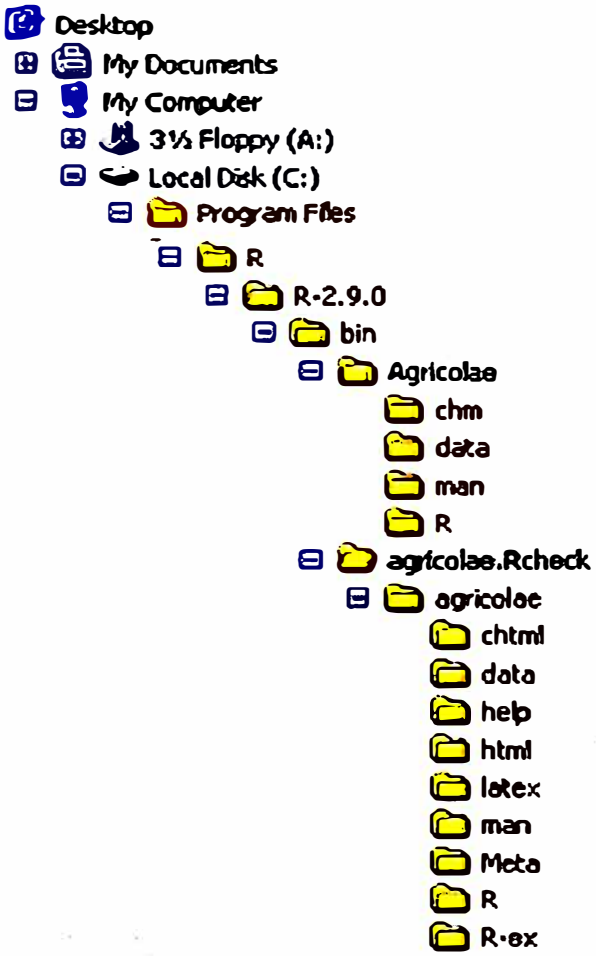


Figura 4.9. Distribución de las carpetas después de ser ejecutado el check de agricolae

4.2.2 "AGRICOLAE" AL REPOSITORIO DE R

Como agricolae paso todas las pruebas de chequeo, se preparó el archivo empaquetado para enviarlo al proyecto R.

```
RCMD build agricolae

C:\PROGRA~1\R\R-2.9.0\bin>RCMD build agricolae

* checking for file 'agricolae/DESCRIPTION' ... OK
* preparing 'agricolae':
* checking DESCRIPTION meta-information ... OK
* removing junk files
* checking for LF line-endings in source and make files
* checking for empty or unneeded directories
* building 'agricolae_1.0-7.tar.gz '
```

C:\PROGRA~1\R\R-2.9.0\bin>

Archivo generado: 'agricolae_1.0-7.tar.gz'.

Se copió a la ruta de R en INTERNET:

ftp://cran.r-project.org/incoming, como se observa la figura 4.10.

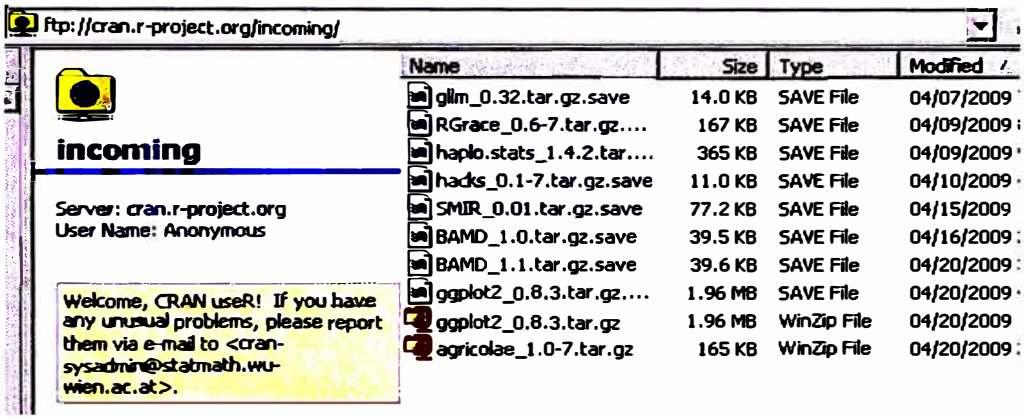


Figura 4.10. Ubicación del archivio agricolae_1.0-7.tar.gz para el chequeo en el Proyecto R

Paralelamente se envio un correo al administrador del sistema R, a la dirección: cran@r-project.org. El administrador de R realizó un chequeo final para su incorporación en las plataformas: Debian GNU/Linux, Fedora, MacOS X y Windows.

4.2.3 PREPARACIÓN DE LA LIBRERÍA Y MANUAL EN WINDOWS

RCMD build –binary agricolae

RCMD Rd2dvi –pdf agricolae

4.2.4 INSTALACIÓN Y REQUERIMIENTOS

Los requerimientos mínimos se limitan a la instalación de la base de R, que es la configuración mínima de librerías y datos para su funcionamiento. R es un programa base que se ejecuta fácilmente en la plataforma de los sistemas de cómputo más utilizados como son Windows (XP, Vista), Linux y MAC. Agricolae requiere que estén instaladas las librerías: akima, corpcor, SuppDists y klaR. Estas librerías aportan algunas funciones necesarias para agricolae; así, "klaR" para el uso del triplot en AMMI(); "akima", para la función grid3d() en interpolación; "SuppDists", para el uso de las funciones no paramétricas kruskal() y friedman(), para el cálculo de la probabilidad; y "corpcor", para la inversa generalizada utilizada en la función PBIB.test() (Joshi, 1987).

La instalación puede realizarce en el proceso de empaquetado. Se ejecuta en la línea de comando del DOS, en la carpeta de BIN la orden:

RCMD INSTALL agricolae

La instalación fue correcta como se visualiza en la ayuda de R, en la sección "Help", "Html Help" y la opción "packages". El paquete agricolae está correctamente instalado como se aprecia en la figura 4.11.

Figura. 4.11. Presentación del listado de librerias instaladas en la PC personal

De esta forma la librería está lista para su uso. Con el proceso de empaquetado se produjo el archivo agricolae_1.0-7.tar.gz. El proyecto R recibió el archivo y realizó el proceso de chequeo. No encontrando ningún error, procedió a generar las diferentes versiones para su instalación. Estos archivos pueden ser descargados de INTERNET a través de la dirección:

http://cran.at.r-project.org/web/packages/agricolae/index.html

Las versiones publicadas son:

Linux: agricolae_1.0-7.tar.gz

MacOS: agricolae_1.0-7.tgz

Windows: agricolae_1.0-7.zip

Manual de referencia: agricolae.pdf

En Windows, descargar el archivo agricolae_1.0-7.zip a una dirección, por ejemplo "C:\", y en la plataforma de R dar el comando:

```
> install.packages("c:/agricolae_1.0-7.zip")
```

o utilizar el menú, y seleccionar **Packages, Install package(s) from local zip** y seleccionar el archivo.

Otra manera es instalar directamente desde el repositorio; para ello desde el menú **Packages**, seleccionar **Install package(s)...,** luego un repositorio **Austria** y finalmente la librería **agricolae.**

Instalada la librería en R, en cualquier versión, la documentación es mostrada al invocar la ayuda en la librería.

```
> library(agricolae)
> help(design.graeco)
```

La información mostrada es:

design.graeco(agricolae) R Documentation

Graeco - latin square design

Description

A graeco - latin square is a KxK pattern that permits the study of k treatments simultaneously with three different blocking variables, each at k levels. The function is only for squares of the odd numbers and even numbers (4, 8, 10 and 12)

Usage

```
design.graeco(trt1, trt2, number = 1, seed = 0, kinds = "Super-Duper")
```

Arguments

trt1	Treatments
trt2	Treatments
number	number of first plot
Seed	seed
kinds	method for to randomize

Details

kinds <- c("Wichmann-Hill", "Marsaglia-Multicarry", "Super-Duper", "Mersenne-Twister", "Knuth-TAOCP", "user-supplied", "Knuth-TAOCP-2002", "default")

Value

trt1	vector, name of the treatments
trt2	vector, name of the treatments
number	Numeric
seed	Numeric

Author(s)

Felipe de Mendiburu

References

1. Statistics for Experimenters Design, Innovation, and Discovery Second Edition. George E. P. Box. Wiley-Interscience. 2005.
2. Experimental design. Cochran and Cox. Second edition. Wiley Classics Library Edition published 1992.

See Also

design.crd, design.lsd, random.ab, fact.nk

Examples

```
library(agricolae)
T1<-c("a","b","c","d")
T2<-c("v","w","x","y")
graeco <-  design.graeco(T1,T2,number=101)
```

De forma similar, para los datos "sweetpotato", siendo el resultado el siguiente:

> help(sweetpotato)

sweetpotato(agricolae) R Documentation

Data of sweetpotato yield

Description

The data correspond to an experiment with costanero sweetpotato made at the locality of the Tacna department, southern Peru. The effect of two viruses (Spfmv and Spcsv) was studied. The treatments were the following: CC (Spcsv) = Sweetpotato chlorotic dwarf, FF (Spfmv) = Feathery mottle, FC (Spfmv y Spcsv) = Viral complex and OO (witness) healthy plants. In each plot, 50 sweetpotato plants were sown and 12 plots were employed. Each treatment was made with 3 repetitions and at the end of the experiment the total weight in kilograms was evaluated. The virus transmission was made in the cuttings and these were sown in the field.

Usage

data(sweetpotato)

Format

A data frame with 12 observations on the following 2 variables.

virus : a factor with levels cc fc ff oo
yield : a numeric vector

Source

Experimental field. Locality of the Tacna department, southern Peru.

References

International Potato Center. CIP - Lima Peru

Examples

```
library(agricolae)
data(sweetpotato)
str(sweetpotato)
```

4.2.5 PUESTA EN MARCHA DE LA LIBRERÍA EN R

Ya instalada correctamente, para cada sesión de R se debe cargar la librería a la memoria del sistema. Así:

```
> library(agricolae)
```

Si se requiere de otras librerías, por ejemplo la librería SuppDists, se debe ejecutar:

```
> library(SuppDists)
```

Para mostrar la bases de datos de agricolae:

```
> data(package= "agricolae")
```

Para consultar a una función específica, por ejemplo PBIB.test():

```
> help(PBIB.test) o ? PBIB.test
```

Para ejecutar ejemplos de la función AMMI:

```
> example(AMMI)
```

Para cargar una base de datos por ejemplo "sweetpotato":

```
> data(sweetpotato)
```

Para la lista de funciones de diseños experimentales:

```
> apropos("design")
 [1] "design.ab"      "design.alpha"   "design.bib"     "design.crd"
 [5] "design.cyclic"  "design.graeco"  "design.lattice" "design.lsd"
 [9] "design.rcbd"    "plot.design"
```

Los siguientes son ejemplos de agricolae:

Ejemplo 1. Diseño latice triple: : función design.lattice()

```
> library(agricolae)
> design.lattice(3,type="triple")

Lattice design,  triple   3 x 3
$square1
     [,1] [,2] [,3]
[1,]    7    6    5
[2,]    9    1    8
[3,]    3    2    4

> design.lattice(3,type="triple")

Lattice design,  triple   3 x 3
$square1
     [,1] [,2] [,3]
[1,]    1    4    3
[2,]    8    6    2
[3,]    9    5    7
```

Cada vez, la función genera cuadros distintos; esto significa un nuevo diseño.

Ejemplo 2. Diseño bloques incompletos balanceado: función design.bib()

```
> library(agricolae)
> # 3 tratamientos y 2 tratamientos por bloque
> trt<-c("A","B","C")
> bib <-design.bib(trt,k=2)
> field <-as.character(bib[,3])
> t(matrix(field,c(2,3)))
```

En una corrida aleatoria puede resultar:

```
     [,1] [,2]
[1,] "C"  "A"
[2,] "B"  "A"
[3,] "C"  "B"
```

En otra podria ser:

```
     [,1] [,2]
[1,] "C"  "A"
[2,] "B"  "C"
[3,] "B"  "A"
```

En cada corrida es diferente la aleatorización.
Los bloques, son las filas, hay dos repeticiones y 3 bloques, siendo cada bloque de tamaño 2. En ambos casos, la eficiencia es la misma. Así:

```
Parameters BIB
==============
Lambda        : 1
treatmeans    : 3
Block size    : 2
Blocks        : 3
Replication   : 2
```

```
Efficiency factor 0.75
```

Ejemplo 3. Prueba de diferencia límite de significación (LSD) con ajuste de Bonferroni: función LSD.test()

Continuando el ejemplo anterior.

```
> attach(sweetpotato)
> compara <- LSD.test(yield, virus, GLe, CMe, p.adj="bon",
main="Rendimiento de camote con presencia de virus")
```

El resultado es presentado en la consola, así:

```
Study: Rendimiento de camote con presencia de virus

LSD t Test for yield
P value adjustment method: bonferroni
                              ......
Alpha                        0.050000
Error Degrees of Freedom     8.000000
Error Mean Square           22.489167
Critical Value of t          3.478879

Treatment Means
  virus    yield  std.err replication
1    cc 24.40000 2.084067           3
2    fc 12.86667 1.246774           3
3    ff 36.33333 4.233727           3
4    oo 36.90000 2.482606           3

Least Significant Difference 13.47040
Means with the same letter are not significantly different.

Groups, Treatments and means
a         oo       36.9
a         ff       36.33333
ab        cc       24.4
 b        fc       12.86667
```

Utilizando la opción de ajuste por Bonferroni.

```
> compara <- LSD.test(yield, virus, GLe, CMe, group=FALSE, p.adj="bon",
main="Rendimiento de camote con presencia de virus")
> detach(sweetpotato)
```

El resultado es presentado en la consola, así:

```
Study: Rendimiento de camote con presencia de virus
LSD t Test for yield
P value adjustment method: bonferroni
                              ......
Alpha                        0.050000
Error Degrees of Freedom     8.000000
Error Mean Square           22.489167
Critical Value of t          3.478879

Treatment Means
  virus    yield  std.err replication
1    cc 24.40000 2.084067           3
2    fc 12.86667 1.246774           3
3    ff 36.33333 4.233727           3
4    oo 36.90000 2.482606           3
```

```
Comparison between treatments means

  tr.i tr.j       diff pvalue
1    1    2 11.5333333 0.1058
2    1    3 11.9333333 0.0904
3    1    4 12.5000000 0.0725
4    2    3 23.4666667 0.0018
5    2    4 24.0333333 0.0015
6    3    4  0.5666667 1.0000
```

Comparar 1 y 2 es comparar "cc" y "fc"; la diferencia es 11.53 y su probabilidad de la diferencia, 0.1058.

Ejemplo 4. Prueba no paramétrica con la función kruskal()

Con los datos de maíz (corn), publicados por Connover.

```
> library(agricolae)
> library(SuppDists)
> data(corn)
> attach(corn)
```

R dispone de la función kruskal.test().

```
> kruskal.test(observation ~ method)
Kruskal-Wallis rank sum test
data:  observation by method
Kruskal-Wallis chi-squared = 25.6288, df = 3, p-value = 1.141e-
05
```

Es similar a los resultados de SPSS y MINITAB.

Con agricolae

```
> compara<-kruskal(observation,method,group=TRUE, main="corn")
```

El resultado es más amplio y tiene las comparaciones específicas entre los tratamientos.

```
Study: corn
Kruskal-Wallis test's

Value: 25.62884
degrees of freedom: 3
Pvalue chisq  : 1.140573e-05
pKruskalWallis: 1.279183e-09

Mean of the ranks
  method observation replication
1      1    21.83333           9
2      2    15.30000          10
3      3    29.57143           7
4      4     4.81250           8

t-Student: 2.042272
Alpha     : 0.05
LSD       : 4.9175
```

```
Harmonic Mean of Cell Sizes  8.351284
Means with the same letter are not significantly different

Groups, Treatments and mean of the ranks
a         3        29.57143
 b        1        21.83333
  c       2        15.3
   d      4        4.8125
```

También permite mostrar gráficos de los objetos creados. Las funciones disponibles de agricolae son bar.group() y bar.err(). Adicionalmente se puede encontrar la probabilidad de comparación con sólo indicar el parámetro group=FALSE de la función kruskal().

```
> compara<-kruskal(observation,method,group=FALSE, main="corn")

Comparison between treatments mean of the ranks
  tr.i tr.j      diff pvalue signf  LSD
1    1    2  6.533333  0.007     * 4.62
2    1    3  7.738095  0.004     * 5.06
3    1    4 17.020833  0.000     * 4.88
4    2    3 14.271429  0.000     * 4.95
5    2    4 10.487500  0.000     * 4.77
6    3    4 24.758929  0.000     * 5.20
```

Ejemplo 5. Análisis de datos de bloques incompletos parcialmente balanceados. Función PBIB.test() para el análisis de un latice 5x5 con 2 repeticiones; datos "soy" de SAS.

En SAS, la numeración de los bloques es del 1 al 5 por repetición. En agricolae, la función PBIB.test() numera a los bloques en forma continua, siendo la primera repetición del 1 al 5; la segunda, del 6 al 10; y así sucesivamente. Con estos cambios se procede al análisis.

```
> library(agricolae)
> library(corpcor)
> soy <- read.table("soy.txt",header=T)
> attach(soy)
> model <-PBIB.test(Block,Treatmnt,Group,Yield,k=5)
```

Resultados de la función PBIB.test()

```
ANALYSIS PBIB: Yield
Class level information
Blocks: 10
Trts : 25
Number of observations: 50

Analysis of Variance Table
Response: y
                      Df Sum Sq Mean Sq  F value  Pr(>F)
replication            1 212.1  212.18  15.53  0.001 **
trt.unadj             24 559.2   23.30   1.70  0.135
replication:block.adj  8 501.8   62.73   4.59  0.004 **
Residuals             16 218.4   13.65
```

```
---
Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
coefficient of variation: 27.1 %

Treatments

Parameters PBIB
treatmeans : 25
Block size : 5
Blocks/rep : 5
Replication: 2

Efficiency factor 0.75
Comparison between treatments means
<<< to see the objects: comparison and means >>>>
> model$means
trt means mean.adj N std.err
1 1 15.0 19.068484 2 2.994033
2 2 11.5 16.973123 2 2.994033
3 3 20.5 14.646724 2 2.994033
...
25 25 11.5 15.404639 2 2.994033

> model$comparison

    tr.i tr.j     diff     stderr  pvalue
1      1     2 2.095249482 3.973854 0.6052
2      1     3 4.421767894 3.973854 0.2822
3      1     4 4.299338435 3.973854 0.2954
4      1     5 6.221106329 3.973854 0.1370
...
299   23    25 3.200661565 3.973854 0.4324
300   24    25 1.921767894 3.973854 0.6352
```

4.2.6 COMPARACIÓN CON SAS, SPSS Y MINITAB

Caso 1: Agricolae y SAS en el planeamiento de 4 bloques en un factorial 2A3B.

Agricolae:

```
> A<-1:2
> B<-1:3
> plan <- design.ab(A, B, r=4, seed= 7913)
> trt<-paste(plan$A,plan$B,sep="-")
> t(matrix(trt,c(6,4)))
     [,1]  [,2]  [,3]  [,4]  [,5]  [,6]
[1,] "2-1" "1-2" "2-3" "2-2" "1-1" "1-3"
[2,] "1-1" "2-3" "1-2" "1-3" "2-2" "2-1"
[3,] "2-3" "2-2" "2-1" "1-1" "1-3" "1-2"
[4,] "1-1" "2-3" "1-3" "2-2" "1-2" "2-1"
```

SAS:

```
data sorgo1 (drop=N V);
do bloques=1 to 4;
do N=1 to 2;
```

```
do V=1 to 3;
NV=10*N+V;
columna=3*(N-1)+V;
output;
end;
end;
end;
run;
proc plan seed=7913;
factors bloques=4 ordered columna=6;
output data=sorgo1 out=sorgo2;
quit;
proc tabulate;
CLASS bloques columna;
var NV;
TABLE bloques, columna*(NV*f=5.0)/rts=6;
keylabel sum = ' ';
run;
```

Resultado:

Diseño 4 bloques con factorial 2x3

	columna					
	1	2	3	4	5	6
	NV	NV	NV	NV	NV	NV
bloques						
1	23	22	12	11	21	13
2	23	13	12	11	22	21
3	13	22	21	23	12	11
4	11	13	12	21	23	22

Corresponde al diseño de bloques con diferente aleatorización.

Caso 2: SAS con los datos de camote en comparación al ejemplo de waller.test(), en la sección 4.1.2, y Bonferroni, en la sección 4.2.5, ejemplo 3

```
proc anova data=sweetpotato;
class virus;
model yield = virus;
means virus / waller BON ;
run;
```

Resultado:

```
Waller-Duncan K-ratio t Test for yield

NOTE: This test minimizes the Bayes risk under additive loss
and certain other
assumptions.

Kratio                                 100
```

```
Error Degrees of Freedom                 8
Error Mean Square                 22.48917
F Value                              17.34
Critical Value of t                2.23603
Minimum Significant Difference       8.658

Means with the same letter are not significantly different.

g           Mean       N    virus
A         36.900       3    oo
A         36.333       3    ff
B         24.400       3    cc
C         12.867       3    fc

Bonferroni (Dunn) t Tests for yield
```

NOTE: This test controls the Type I experimentwise error rate, but it generally has a higher Type II error rate than REGWQ.

```
Alpha                                0.05
Error Degrees of Freedom                8
Error Mean Square                 22.48917
Critical Value of t                3.47888
Minimum Significant Difference       13.47

Means with the same letter are not significantly different.
 Bon
Groupi
  ng              Mean       N    virus

     A         36.900       3    oo
     A         36.333       3    ff
B    A         24.400       3    cc
B              12.867       3    fc
```

Los resultados coinciden en el valor critico de t, la mínima diferencia significativa y la formación de grupos. SAS no presenta las probabilidades de la comparación, solo los intervalos de confianza con la opción CLDIFF.

```
means virus /bon cldiff;

                Difference
  virus            Between      Simultaneous 95%
Comparison           Means    Confidence Limits

 oo - ff             0.567     -12.904    14.037
 oo - cc            12.500      -0.970    25.970

...
```

Caso 3: Con SPSS para el ejemplo 3

En SPSS, los factores deben ser numéricos. Para el análisis se asignó números a los tratamientos: 1=cc, 2=fc, 3=ff, 4=oo. Los resultados se muestran para todas las comparaciones en la tabla 4.1.

Tabla 4.1 Resultado de la comparación de virus con SPSS.

Variable dependiente: yield

	(I) virus	(J) virus	Diferencia de medias (I-J)	Error típico	Sig.
Bonferroni	1	2	11.5333	3.8721	.106
		3	-11.9333	3.8721	.090
		4	-12.5000	3.8721	.073
	2	3	-23.4667(*)	3.8721	.002
		4	-24.0333(*)	3.8721	.002
	3	4	-.5667	3.8721	1.000

* La diferencia de medias es significativa al nivel .05.

Los resultados obtenidos en el ejemplo 3 con agricolae son iguales a los producidos por SPSS. El error típico puede ser calculado con los datos de agricolae como sqrt(2*22.489167/3) = 3.872051.

Caso 4. Con SPSS, ejemplo de waller.test() sección 4.1

Para SPSS, los tratamientos deben ser números, la prueba de Waller con SPSS produce la tabla 4.2. Tratamientos : 1=cc, 2=fc, 3=ff, 4=oo

Tabla 4.2. Waller-Duncan con SPSS

			Subconjunto para alfa = .05		
	virus	N	2	3	1
Waller-Duncan(a,b)	2	3	12.8667		
	1	3		24.4000	
	3	3			36.3333
	4	3			36.9000

La forma es un poco distinta, pero los grupos son coincidentes con agricolae.

Caso 5. Con SAS para el ejemplo 4

```
proc rank data=corn out=a;
var observation;
run;
proc anova data=a;
class method;
model observation = method;
means method/lsd lines;
run;
quit;
```

```
Alpha                          0.05
Error Degrees of Freedom         30
Error Mean Square           24.20943
```

```
Critical Value of t            2.04227
Least Significant Difference    4.9175
Harmonic Mean of Cell Sizes   8.351284

g          Mean       N     method
A        29.571       7     3
B        21.833       9     1
C        15.300      10     2
D         4.813       8     4
```

El resultado es el mismo; adicionalmente, agricolae halla las probabilidades de las comparaciones.

Caso 6 SPSS y MINITAB para el ejemplo 4.

SPSS:

	observation
Chi-cuadrado	25.629
Gl	3
Sig. asintót.	.000

a Prueba de Kruskal-Wallis
b Variable de agrupación: method

Rangos

	method	N	Rango promedio
observation	1.00	9	21.83
	2.00	10	15.30
	3.00	7	29.57
	4.00	8	4.81
	Total	34	

MINITAB:

```
Kruskal-Wallis Test: observation versus method
Kruskal-Wallis Test on observation

method     N  Median  Ave Rank      Z
1          9   91.00      21.8   1.52
2         10   86.00      15.3  -0.83
3          7   95.00      29.6   3.60
4          8   80.50       4.8  -4.12
Overall   34              17.5
H = 25.46  DF = 3  P = 0.000
H = 25.63  DF = 3  P = 0.000  (adjusted for ties)
```

Los resultados son iguales en medias de rangos, estadístico de la prueba (25.62884), y probabilidad de la prueba con chi-cuadrado (1.140573e-05), que es 0.000. Adicionalmente, agricolae calcula las probabilidades de la comparación y también forma los grupos de significación.

Caso 7 Con SAS para el ejemplo 5.

Los datos de "soy" están propuestos en la función PROC LATICE de SAS, cuyo procedimiento se muestra exclusivamente.

```
data Soy;
     do Group = 1 to 2;
       do Block = 1 to 5;
         do Plot = 1 to 5;
            input Treatmnt Yield @@;
            output;
         end;
       end;
     end;
     drop Plot;
     datalines;
   1   6   2   7   3   5   4   8   5   6
   6   16  7   12  8   12  9   13  10  8
   11  17  12  7   13  7   14  9   15  14
   16  18  17  16  18  13  19  13  20  14
   21  14  22  15  23  11  24  14  25  14
   1   24  6   13  11  24  16  11  21  8
   2   21  7   11  12  14  17  11  22  23
   3   16  8   4   13  12  18  12  23  12
   4   17  9   10  14  30  19  9   24  23
   5   15  10  15  15  22  20  16  25  19
   ;
proc lattice data=Soy; run;
```

Resultado:

```
The Lattice Procedure
Analysis of Variance for Yield
```

Source	DF	Sum of Squares	Mean Square
Replications	1	212.18	212.18
Blocks within Replications (Adj.)	8	501.84	62.7300
Treatments (Unadj.)	24	559.28	23.3033
Intra Block Error	16	218.48	13.6550
Total	49	1491.78	30.4445

```
Additional Statistics for Yield
Variance of Means in Same Block 15.7915
Variance of Means in Different Bloc 17.9280
Average of Variance 17.2159
LSD at .01 Level 12.1189
LSD at .05 Level 8.7959
Efficiency Relative to RCBD 174.34
Adjusted Treatment
Means for Yield
Treatment Mean

1 19.0681
...
25 15.4048
1 vs 2, Variance of Means in Same Block 15.7915
diff = abs(19.0681 - 16.9728) = 2.0953
t.stat = 2.0953 / sqrt(15.7915) =  0.5272718
```

```
Df = 16
pvalue (two sided) = 2*(1-pt(0.5272718,16)) = 0.6052396
```

Con el procedimiento de SAS PROC MIXED se tiene los mismos resultados.

```
proc mixed data=soy;
class Group Block Treatmnt;
model yield = Treatmnt;
random Group block(group);
lsmeans Treatmnt/pdiff;
run;
quit;
Type 3 Tests of Fixed Effects
Num Den
Effect      DF DF F Value  Pr > F
Treatmnt    24 16 1.97     0.0824
Differences of Least Squares Means
                                        Standard
Effect Treatmnt _Treatmnt Estimate Error DF  t Value Pr> |t|
Treatmnt      1          2   2.09 3.97   16     0.53   0.6052
...
Treatmnt     24         25   1.92 3.97   16     0.48   0.6352
```

La función PBIB.test() de agricolae produce resultados iguales en las probabilidades del análisis de variancia de SAS. También son equivalentes en la comparación de medias ajustadas; así, en la comparación de 1 vs. 2, la probabilidad es de 0.6052.

En las siguientes tablas: 4.5, 4.6, 4.7 y 4.8 se presenta un resumen comparativo.

Tabla 4.5. Comparación en Diseños experimentales

Procedimiento	SAS	R-agricolae
Aleatorio, bloque, latino	Generación	Generación y libro de campo
Greco-latino		Generación y libro de campo
Bloques incompletos balanceados		Generación y libro de campo
Latice	Generación	Generación y libro de campo
Alfa		Generación y libro de campo

Tabla 4.6. Comparación en funciones de medias de tratamientos.

Procedimiento	SAS	SPSS	MINITAB	R-agricolae
LSD- student	grupos, Probabilidad intervalos	Intervalos	Intervalos	Grupos, probabilidad
HSD- tukey	Grupos, Probabilidad intervalos	Intervalos	Intervalos	Grupos, probabilidad
Kruskal, Friedman	Un valor y una probabilidad	Un valor y una probabilidad	Un valor y una probabilidad	Adicionalmente Comparaciones grupos
Waller-Duncan	Grupos	Grupos		Grupos, probabilidad
Bonferroni	Grupos, intervalos	Probabilidad		Grupos, probabilidad

Tabla 4.7 Comparación en estabilidad genética

Procedimiento	SAS	R-agricolae
Paramétrica y no paramétrica		Análisis
AMMI	Análisis y gráfico mediante macros	Análisis y gráficos

Tabla 4.8. Comparación en otras funciones de agricolae

Procedimiento	SAS	R-agricolae
Análisis bloques incompletos	Medias ajustadas	Análisis completo
Línea x Probador	Modelo parcial	Análisis completo
Diseños carolina	Modelo	Análisis
Consenso dendrogramas		Gráfico general y parcial

4.2.7 GENERACIÓN DE LA TABLA DE WALLER-DUNCAN

Agricolae puede producir valores tabulares de Waller-Duncan como lo publicado en; Steel (1997) en pag. 652 como muestra la tabla 4.4. Con waller(), se utilizó los siguientes prametros: K=100, F=4, 6, 10, 25 y 150, y se generó la tabla 4.3.

```
library(agricolae)
K<-100
Fc<-4
q<-c(2,4,6,10,20,100)
f<-c(seq(4,20,2),24,30,40,60,120)
n<-length(q)
m<-length(f)
W.D <-rep(0,n*m)
dim(W.D)<-c(n,m)
for (i in 1:n) {
for (j in 1:m) {
W.D[i,j]<-waller(K, q[i], f[j], Fc)
}}
W.D<-round(W.D,2)
dimnames(W.D)<-list(q,f)
print(W.D)
```

Para generar la segunda parte de la tabla 4.3, se cambió el valor de Fc y los valores de q, según se indica en la tabla 4.3. Los resultados son iguales a los de la tabla 4.4. Para valores infinitos no es posible, generalmente se utiliza los valores máximos de la tabla, tal como se muestra para q=100.

Para valores particulares que no se muestran en la tabla, la función waller() puede ser aplicada. Así, para K=500 (equivalente al valor 0.01 en la significación de la prueba); 15 grados de libertad, para el error; 5 grados de libertad, para tratamientos; y 3.5, para el valor de F calculado de tratamientos. El valor tabular es 4.349:

```
> waller(K=500, q=15, f=5, Fc= 3.5); 4.349
```

Tabla 4.3. Valores criticos de Waller-Duncan. F= 4, 6, 10, 25,150

F = 4

	4	6	8	10	12	14	16	18	20	24	30	40	60	120
2	2.85	2.58	2.44	2.35	2.29	2.25	2.22	2.20	2.18	2.15	2.12	2.09	2.06	2.03
4	2.85	2.63	2.50	2.41	2.35	2.30	2.27	2.24	2.22	2.18	2.15	2.12	2.08	2.04
6	2.85	2.65	2.52	2.44	2.37	2.32	2.29	2.25	2.23	2.19	2.16	2.12	2.08	2.04
10	2.85	2.67	2.55	2.46	2.39	2.34	2.30	2.26	2.24	2.20	2.16	2.12	2.08	2.04
20	2.85	2.69	2.57	2.47	2.40	2.35	2.30	2.27	2.24	2.20	2.15	2.11	2.07	2.03
100	2.85	2.70	2.58	2.49	2.41	2.35	2.31	2.27	2.24	2.19	2.15	2.11	2.06	2.02

F = 6

	4	6	8	10	12	14	16	18	20	24	30	40	60	120
2	2.85	2.53	2.37	2.27	2.21	2.16	2.13	2.10	2.08	2.05	2.02	1.99	1.96	1.93
4	2.85	2.56	2.40	2.30	2.23	2.18	2.14	2.12	2.09	2.06	2.02	1.99	1.96	1.93
6	2.85	2.58	2.42	2.31	2.24	2.19	2.15	2.12	2.09	2.06	2.02	1.99	1.96	1.92
10	2.85	2.59	2.43	2.32	2.24	2.19	2.15	2.12	2.09	2.06	2.02	1.99	1.95	1.92
20	2.85	2.60	2.44	2.33	2.25	2.19	2.15	2.12	2.09	2.05	2.02	1.98	1.95	1.92
100	2.85	2.61	2.44	2.33	2.25	2.19	2.15	2.12	2.09	2.05	2.02	1.98	1.95	1.92

F = 10

	4	6	8	10	12	14	16	18	20	24	30	40	60	120
2	2.85	2.48	2.29	2.19	2.12	2.07	2.04	2.01	1.99	1.96	1.93	1.9	1.87	1.85
4	2.85	2.49	2.31	2.20	2.13	2.08	2.04	2.01	1.99	1.96	1.93	1.9	1.87	1.84
6	2.85	2.50	2.31	2.20	2.13	2.08	2.04	2.01	1.99	1.96	1.93	1.9	1.87	1.84
10	2.85	2.51	2.32	2.20	2.13	2.08	2.04	2.01	1.99	1.96	1.93	1.9	1.87	1.84

F = 25

	4	6	8	10	12	14	16	18	20	24	30	40	60	120
2	2.85	2.40	2.21	2.1	2.03	1.99	1.95	1.93	1.91	1.88	1.86	1.83	1.8	1.78
10	2.84	2.41	2.21	2.1	2.03	1.99	1.95	1.93	1.91	1.88	1.86	1.83	1.8	1.78

F = 150

	4	6	8	10	12	14	16	18	20	24	30	40	60	120
2	2.85	2.34	2.14	2.04	1.98	1.94	1.91	1.89	1.87	1.84	1.82	1.8	1.77	1.75

TABLE A.20 (CONTINUED)
Duncan-Waller minimum-average-risk t values (k = 100)

	f														
q	4	6	8	10	12	14	16	18	20	24	30	40	60	120	∞
							$^{\dagger}F$ = 4.0 (a = .500, b = 1.155)								
2	*	2.58	2.44	2.35	2.29	2.25	2.22	2.20	2.18	2.15	2.12	2.09	2.06	2.03	2.00
4	2.85	2.63	2.50	2.41	2.35	2.30	2.27	2.24	2.22	2.18	2.15	2.12	2.08	2.05	2.01
6	2.85	2.65	2.52	2.43	2.37	2.32	2.28	2.25	2.23	2.19	2.16	2.12	2.08	2.04	2.01
10	2.85	2.67	2.55	2.46	2.39	2.34	2.30	2.26	2.24	2.20	2.16	2.12	2.08	2.04	2.00
20	2.85	2.69	2.57	2.47	2.40	2.35	2.30	2.27	2.24	2.20	2.15	2.11	2.07	2.03	1.99
∞	2.85	2.71	2.59	2.49	2.42	2.36	2.31	2.27	2.24	2.19	2.15	2.11	2.06	2.02	1.99
							$^{\dagger}F$ = 6.0 (a = .408, b = 1.095)								
2	2.85	2.53	2.37	2.27	2.21	2.16	2.13	2.10	2.08	2.04	2.02	1.99	1.96	1.93	1.90
4	2.85	2.56	2.40	2.30	2.23	2.18	2.14	2.12	2.09	2.06	2.02	1.99	1.96	1.93	1.90
6	2.85	2.58	2.42	2.31	2.24	2.19	2.15	2.12	2.09	2.06	2.02	1.99	1.95	1.92	1.89
10	2.85	2.59	2.43	2.32	2.24	2.19	2.15	2.12	2.09	2.06	2.02	1.99	1.95	1.92	1.89
20	2.85	2.60	2.44	2.32	2.25	2.19	2.15	2.12	2.09	2.05	2.02	1.98	1.95	1.92	1.89
∞	2.85	2.61	2.44	2.33	2.25	2.19	2.15	2.12	2.09	2.05	2.02	1.98	1.95	1.92	1.89
							$^{\dagger}F$ = 10.0 (a = .316, b = 1.054)								
2	2.85	2.48	2.30	2.19	2.12	2.07	2.04	2.01	1.99	1.96	1.93	1.90	1.87	1.85	1.82
4	2.85	2.49	2.31	2.20	2.13	2.08	2.04	2.01	1.99	1.96	1.93	1.90	1.87	1.84	1.82
6	2.85	2.50	2.31	2.20	2.13	2.08	2.04	2.01	1.99	1.96	1.93	1.90	1.87	1.84	1.82
10–∞	2.85	2.51	2.32	2.20	2.13	2.08	2.04	2.01	1.99	1.96	1.93	1.90	1.87	1.84	1.82
							$^{\dagger}F$ = 25.0 (a = .200, b = 1.021)								
2–4	2.85	2.40	2.20	2.10	2.03	1.99	1.95	1.93	1.91	1.88	1.86	1.83	1.80	1.78	1.76
10–∞	2.85	2.41	2.21	2.10	2.03	1.99	1.95	1.93	1.91	1.88	1.86	1.83	1.80	1.78	1.76
							$^{\dagger}F$ = ∞ (a = 0, b = 1)								
2–∞	2.85	2.33	2.13	2.03	1.97	1.93	1.90	1.88	1.86	1.84	1.81	1.79	1.76	1.74	1.72

*All differences not significant.
†See boundary conditions on page 649.

Figura 4.12. Waller-Duncan (Steel, 1997) pag. 652

4.2.8 RESULTADOS DE LA ENCUESTA

La encuesta estuvo disponible en INTERNET desde diciembre del 2008 hasta junio del 2009. Se recibió 48 encuestas vía correo electrónico, provenientes del Perú (35) y otros países (13), las cuales se analizaron mediante el programa R y las librerías agricolae y RODBC:

Lectura de la encuesta:

```
> library(agricolae)
> library(RODBC)
> setwd("E:/Encuestas agricolae")
> canal <- odbcConnectExcel("datos para el análisis.xls")
> T1<- sqlFetch(canal,"tema1")
> T2<- sqlFetch(canal,"tema2")
> T3<- sqlFetch(canal,"tema3")
> T4<- sqlFetch(canal,"tema4")
> odbcCloseAll()
> # T1:  P1, P11, P12, P15, P18, P19, P20, P21
> # P18 Genero
> total<-table(T1[,6])
> P18<- round(total*100/sum(total),0)
> P18

Mujer Varón
   17    83
> # P20 Grados
> total<-table(T1[,8])
> P20<- round(total*100/sum(total),0)
> # P21 Ocupación
> total<-table(T1[,9])
> P21<- round(total*100/sum(total),0)
> # P19 Edad
> P19<-na.omit(T1[,7])
> clases<-c(20,35,50,65)
> h<- graph.freq(P19,breaks=clases,plot=F)
> table.freq(h)
> # P17, tema 4 Otros programas estadísticos, FIGURA 4.2
> x<-sort(table(T4[,2]),decreasing = TRUE)
> P17<-round(c(x[1:5],OTROS=sum(x[-(1:5)]))*100/42,1)
> par(mar=c(4,4,2,0),cex=0.7)
> x<-barplot(P17,density=c(30,24,16,8,4,0), col=colors()[81], lwd=2,
ylim=c(0,60), ylab="%")
> text(4,55,"Usuarios de agricolae que utilizan\n otros programas
estadisticos",col="blue")
> # Especialidad
> total<-table(T1[,10])
> especialidad<- round(total*100/sum(total),0)
```

Tipo de usuarios más frecuentes

Según la especialidad, grado académico, ocupación, edad, tipos de programa que utilizan y la institución donde trabajan, presentaron la siguiente distribución de frecuencia:

Por especialidad

Forestales	Estadísticos	Agrónomos	Biólogos	Genetistas	Otros
12(25%)	11(23%)	9(19%)	4(8%)	(2%)	11(23%)

Los usuarios más frecuentes pertenecen al area forestal, con 25%; en segundo lugar los estadísticos, con 23%, y los agrónomos, en tercer lugar, con 19%.

Por grado académico

PhD	Maestría	Graduado
11(23%)	30(62%)	7(15%)

Los usuarios más frecuentes tienen grado de maestria, con 62%; en segundo lugar, los de grado de doctor, con 23%.

Por ocupación:

Investigador	Docente	Estudiante
32(67%)	7(15%)	9(19%)

Los investigadores son los más frecuentes, con un 67%.

Por tipo de programa estadístico

Según la encuesta, los software más utilizados son SPSS, SAS y MINITAB, como se muestra en la figura 4.2. Más del 30% de usuarios utiliza al menos uno de estos programas.

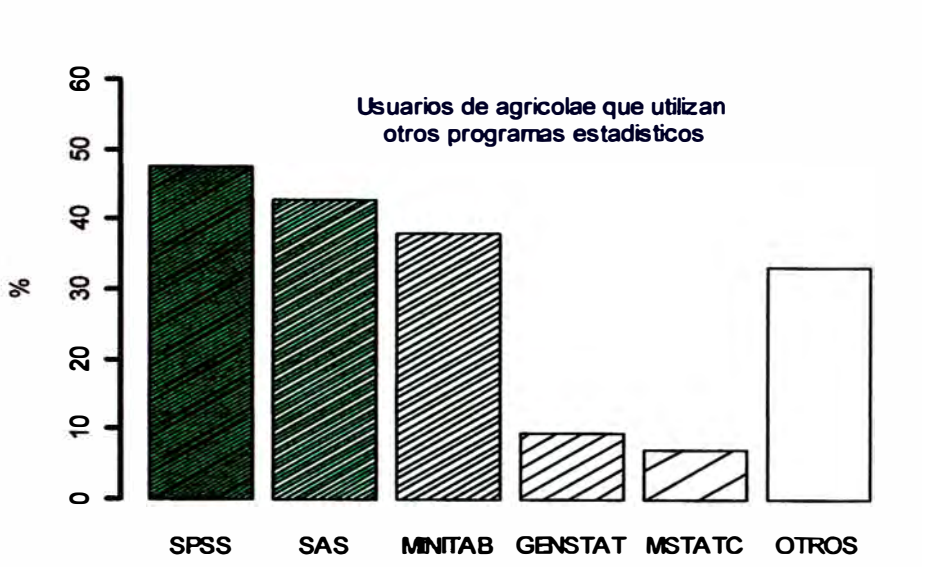


Figura 4.2. Uso de otros programas

Por edad.

El 41.6% de los encuestados tiene entre 20 y 35 años de edad, y el 33% representa a los usuarios mayores de 50 años.

Tabla de frecuencias de las edades:

Inf	Sup	MC	fi	fri	Fi	Fri
20	35	27.5	20	0.4166667	20	0.4166667
35	50	42.5	12	0.2500000	32	0.6666667
50	65	57.5	16	0.3333333	48	1.0000000

Instituciones con mayor número de encuestas enviadas:

- Universidad Nacional Agraria La-Molina (19)
- International Potato Center (8)
- INIA- Peru (5)

Otras instituciones con menor número de encuestas (16):

- University of Copenhagen - Dinamarca
- Warsaw University of Life Sciences - Polonia
- ARVALIS - Institut du Végétal – Francia
- ITB – Institut technique de la betterave – Francia
- Department Earth and Environmental Sciences, K.U.Leuven- Belgica
- Chinese Academy of Science the Institute of Botany - China
- Government of Canada - Canada
- ICRISAT - India
- Ind Inst of Hort Research, Bangalore - India
- IRRI – Filipinas

- Universidade do Estado de Santa Catarina - Brasil
- Edumetrics Cía. Ltda - Ecuador
- MINAG - Perú
- Universidad Nacional San Antonio Abad-Cusco - Perú
- Universidad Nacional Altiplano – Perú

Antigüedad del uso de agricolae por el usuario

P1.- ¿Cuánto tiempo hace que utiliza la librería AGRICOLAE?

```
> total<-table(T1[,2])
> P1<- round(total*100/sum(total),0)
> par(mar=c(0,0,0,0),cex=0.6)
> pie(P1,density=c(4,8,16,0), col=c(1,2,3,4))
> legend("topright",title="Meses  %",paste(names(P1)," :
",as.numeric(P1),sep=""))
```

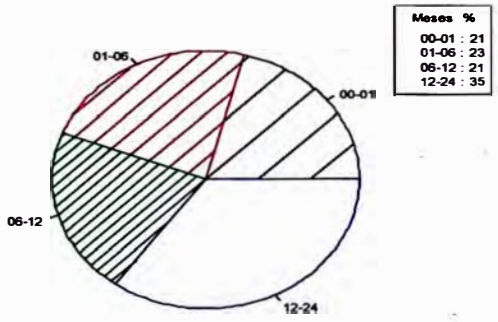


Figura 4.3. Distribución de los usuarios según la antigüedad del uso de agricolae

Según la figura 4.3, el 44 % de los encuestados usa la librería por más de 1 mes y menos de 1 año, y el 35% la usa por más de un año. De acuerdo a este resultado se puede afirmar que el 79% de los participantes en la encuesta conoce las funciones de agricolae.

Análisis de satisfacción.

Para relacionar la apuntación de la tabla 3.4 con la escala 0-1 (Sadana, 2006), se utilizo la siguiente relación:

Valor = -0.25 + 0.25*puntaje;

Siendo su equivalencia de 1, 2, 3 ,4 y 5 a: 0, 0.25, 0.5, 0.75 y 1.

Análisis de frecuencia por tipo de función de agricolae.

Se aplicó el siguiente procedimiento en R para obtener los resultados por tipo de función.

```
> # Tema 2, P2 y P3
> # Escala de 0 a 1
> E3 <- -0.25+0.25*T2[,3]
> P3 <- tapply.stat(E3,T2[,2], mean)
> V3 <- tapply.stat(E3,T2[,2], var)
> N3 <- tapply.stat(E3,T2[,2],length)
> tabla<- cbind(round(table(T2[,2])*100/45,1),round(P3[,2],2))
> tabla<-data.frame(tabla,n= N3[,2],var=V3[,2])
> tabla <- tabla[-5,]
> # ordenando
> tabla <- tabla[order(tabla[,1]),]
> colnames(tabla)<-c("%", "Escala", "n", "var")
> par(mar=c(4,12,2,2),cex=0.8)
> x<-barplot(tabla[,1],las=2,horiz=T,density=6,xlab="% de
uso",xlim=c(0,100), names.arg = rownames(tabla),col= "blue")
> text(cbind(tabla[,1]+8,x),paste(as.numeric(tabla[,2])))
> title(main="Agricolae: % uso y satistacción de las funciones\nEscala 0-1")
```

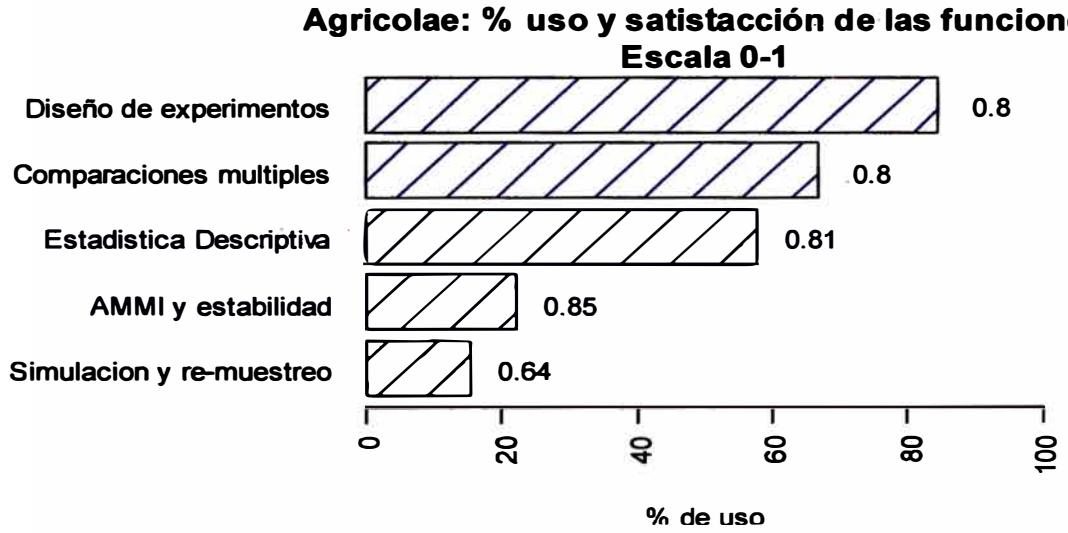


Figura 4.4. Satisfacción del uso de funciones de agricolae

El objeto creado "tabla" proporcionan la figura 4.4 y la tabla 4.10.

Tabla 4.10 Satisfacción del uso de funciones y sus estadisticas.

	%	Escala	n	var
Simulacion y re-muestreo	15.6	0.64	7	0.01785714
AMMI y estabilidad	22.2	0.85	10	0.03055556
Estadistica Descriptiva	57.8	0.81	26	0.03653846
Comparaciones multiples	66.7	0.80	30	0.04482759
Diseño de experimentos	84.4	0.80	38	0.02080370

Según la figura 4.4 y la tabla 4.10, los tipos de funciones más utilizados fueron: "Diseño Experimental", con 84.4%, y "comparaciones múltiples" con 66.7%. Respecto a la calificación en la escala de 0 – 1, las funciones de estabilidad alcanzan el nivel 0.85, siendo el más alto según la escala 0-1 (Sadana, 2006). Otras funciones presentan puntajes mayores a 0.75, siendo catalogadas como muy buenas, y el tipo "simulación y re-muestreo", como bueno.

Según la prueba de t-student para la media de la escala en "comparaciones múltiples" bajo la hipótesis alterna mayor de 0.5 resulto. así:

T calc. = (0.8-0.5) /sqrt(0.04482/30) = 7.76, p.valor = 7.37e-09

De igual forma para las medias de los otros tipos según el número de respuestas se encontró p.valores como:

Diseño de experimentos: 1.17 e-15

Estadística descriptiva: 2.43 e-08

Estadísticamente los niveles de escala son superiores a 0.5, dando una satisfacción muy buena para los tipos de función.

Análisis de las preguntas de satisfacción del usuario dado por las preguntas de la encuesta P4, P5, P6, P12 y P15, según la escala de Likert.

```
> # Tema 3:  likert P 4-5-6-12-15
> A<-T3[,c(2:20,26:27)]
> rownames(A)<-T3[,1]
```

La matriz original tiene 48 filas y 21 columnas.

Promedios por ítem, para puntajes entre 1 y 5:

```
> round(mean(A,na.rm=T),1)

P4_1 P4_2 P4_3 P4_4 P4_5 P4_6 P4_7 P5_1 P5_2 P5_3 P5_4
 4.2  4.3  4.2  4.4  4.0  3.8  4.0  4.4  4.2  4.5  4.1

P5_5 P5_6 P5_7 P6_1 P6_2 P6_3 P6_4 P6_5  P12  P15
 3.7  4.4  4.3  4.4  4.2  3.9  3.7  4.0  4.8  4.7
```

El puntaje más bajo fue 3.7 para P5_5 "Conocimiento de agricultura" y P6_4 "Agricolae es fácil de usar", y el más alto, 4.8, para P12 "Usará agricolae en el futuro".

Puntaje total por ítems:

```
> sum(A,na.rm=T)
[1] 4076
```

El puntaje total máximo esperado: (48)(21)(5) = 5040

El factor de calidad es: 4076/5040 = **0.8087**

Evaluación de los ítems.

Para la evaluación de los items se elimina las filas con celdas vacías, resultando 39 filas (encuestas):

```
> A<-na.omit(A)
```

La nueva matriz completa es de 21 variables: "P4_1" "P4_2" "P4_3" "P4_4" "P4_5" "P4_6" "P4_7", "P5_1" "P5_2" "P5_3" "P5_4" "P5_5" , "P5_6" "P5_7" , P6: "P6_1" "P6_2" "P6_3" "P6_4" "P6_5", "P12" y "P15"

Con la matriz se calcula el puntaje total, sus límites de confianza, la correlación de spearman de cada variable con el total, y el coeficiente de Cronbach.

Estimación del límite de confianza del puntaje total de satisfacción

Según el número de ítems, se espera un puntaje promedio de (21+105)/2 = 63.

Según la encuesta:

```
> unos <- rep(1,ncol(A))
> suma<- as.matrix(A)%*%unos
> n<- nrow(A)
> media <- mean(suma)
> pos1<-1*21
> pos5<-5*21
> esperado <- (pos1+pos5)/2
> sdprom <- sd(suma)/sqrt(n)
> Linf <- media - 1.96*sdprom
> Lsup <- media + 1.96*sdprom
```

El promedio es 87.53 y su desviación estándar, 1.3759; entonces sus límites de confianza son < 84.8 ; 90.2 >

El intervalo de confianza está por encima del valor esperado (63),
por lo tanto es satisfactorio a favor de la calidad.

Cálculo de la correlación de spearman y la determinación de las variables más importantes para la satisfacción de los diferentes tipos de función

```
> correlacion <- cor(A,suma,method= "spearman")
```

Con los puntajes totales se separa dos grupos: el 25% superior y el 25% inferior de las encuestas, y se calcula las diferencias entre los promedios para cada ítem (pregunta de la encuesta). Para esto, se utiliza los cuantiles del tipo 6 que son los cuantiles estándar para los programas SAS, SPSS y MINITAB.

```
> maximos<-A[suma>quantile(suma,0.75,type=6),]
> minimos<-A[suma<quantile(suma,0.25,type=6),]
```

```
> diferencia <-  mean(maximos)-mean(minimos)
> puntajes<- -0.25+0.25*mean(A)
```

Con las diferencias, las correlaciones y los puntajes se forma una nueva tabla para escoger los ítems de máxima diferencia y correlación alta.

```
> B<- data.frame(diferencia, correlacion, puntajes)
> B<-round(B,4)
```

La tabla del objeto B tiene el siguiente contenido

```
> B
```

	diferencia	correlacion	puntajes
P4_1	1.1528	0.7422	0.7949
P4_2	0.9167	0.5999	0.8013
P4_3	1.3056	0.6478	0.7949
P4_4	1.1528	0.5778	0.8526
P4_5	1.1528	0.4606	0.7628
P4_6	0.6250	0.3715	0.6987
P4_7	1.3194	0.5895	0.7500
P5_1	0.6389	0.3164	0.8397
P5_2	0.9306	0.3365	0.7885
P5_3	0.5417	0.2830	0.8590
P5_4	0.8333	0.4475	0.7756
P5_5	1.1667	0.4208	0.6731
P5_6	0.7778	0.5118	0.8397
P5_7	0.8056	0.4139	0.8013
P6_1	1.0139	0.6068	0.8526
P6_2	1.3889	0.7627	0.7756
P6_3	1.5694	0.5247	0.7179
P6_4	2.4167	0.7559	0.6667
P6_5	1.5278	0.7511	0.7308
P12	0.6250	0.5153	0.9423
P15	0.7500	0.5366	0.9167

En la tabla B se debe seleccionar el conjunto de variables que tienen las máximas diferencias y las más altas correlaciones. El criterio es:

Las diferencias mayores que sean superiores a la mediana de las diferencias.

Las correlaciones mayores a 0.60.

Con estos criterios se tiene dos conjuntos selectos:

```
> selecto1<-B[B[,1]> quantile(B[,1],0.5,type=6),]
> selecto2<- B[B[,2]>0.6,]
> selecto1
```

	diferencia	correlacion	puntajes
P4_1	1.1528	0.7338	0.7949
P4_3	1.3056	0.6983	0.7949
P4_4	1.1528	0.6825	0.8526
P4_5	1.1528	0.4314	0.7628
P4_7	1.3194	0.6897	0.7500
P5_5	1.1667	0.3927	0.6731
P6_2	1.3889	0.7145	0.7756
P6_3	1.5694	0.6123	0.7179
P6_4	2.4167	0.7876	0.6667
P6_5	1.5278	0.7358	0.7308

```
> selecto2
      diferencia correlacion puntajes
P4_1      1.1528      0.7422   0.7949
P4_3      1.3056      0.6478   0.7949
P6_1      1.0139      0.6068   0.8526
P6_2      1.3889      0.7627   0.7756
P6_4      2.4167      0.7559   0.6667
P6_5      1.5278      0.7511   0.7308
```

La intersección de ambos conjuntos conforma los ítems que mejor miden el nivel de satisfacción del usuario según los criterios de la escala de Likert:

```
> selecto <- selecto1[rownames(selecto1)%in%rownames(selecto2),]
> selecto[order(selecto[,3],decreasing=T),]
      diferencia correlacion puntajes
P4_1      1.1528      0.7422   0.7949
P4_3      1.3056      0.6478   0.7949
P6_2      1.3889      0.7627   0.7756
P6_5      1.5278      0.7511   0.7308
P6_4      2.4167      0.7559   0.6667
```

Estas variables en orden de importancia corresponden a:

P4_1 Satisfecho con la calidad

P4_3 Integración con R

P6_2 Agricolae hace lo que dice

P6_5 Agricolae es competitivo

P6_4 Agricolae es fácil de usar

```
mean(selecto)
diferencia correlacion    puntajes
   1.55836     0.73194     0.75258
```

El promedio 0.752 es superior a 0.75 según la escala de 0-1 (Sadana, 2006); siendo muy buena la satisfacción del usuario respecto del software.

Estimación del índice alfa de Cronbach (α)

```
> V<-diag(var(A))
> S<-var(suma)
> alpha <- (21/20)*(1 - sum(V)/S)
> alpha
```

El valor estimado de α es 0.88. Como es superior a 0.80, el instrumento para medir la satisfacción de la librería agricolae es fiable.

Análisis de las preguntas de satisfacción del usuario respecto a otros programas estadísticos (preguntas 11, 12 y 15 de la encuesta)

P11.- Piense en productos similares ofrecidos por otros programas estadísticos, ¿cómo comparar los procedimientos ofrecidos por agricolae dentro del programa R?

```
> par(mar=c(6,4,2,2))
> total<-table(T1[,3])
> P11<- round(total*100/sum(total),0)
> barplot(c(P11[4],P11[-4]),density=c(8,16,24,32,0), col=2:5, ylim=c(0,50),
ylab="%",
main="Funciones de agricolae\nversus otros programas similares",las=2)
```

Según la figura 4.5, más del 40% considera mejor a los productos ofrecidos por agricolae dentro de R.

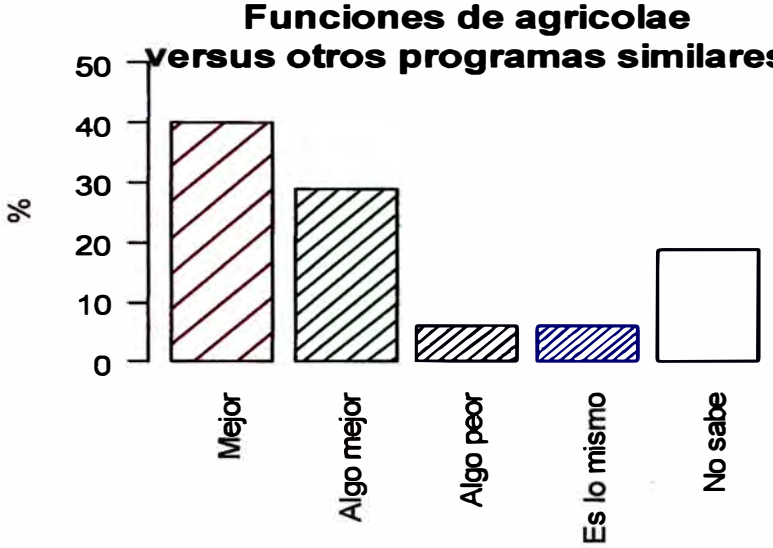


Figura 4.5. Comparación con otros software para las funciones de agricolae

P12.-¿Usaría agricolae en el futuro?

```
> total<-table(T1[,4])
> P12<- sort(round(total*100/sum(total),0) , decreasing=T)
> par(mar=c(2,4,1,0),cex=0.85)
> barplot(P12,col=colors()[c(91,21,1)], ylim=c(0,100), ylab="%")
> text(2,95,"¿Usaria agricolae en el futuro?")
```

Según la opinión de los encuestados, el 80% definitivamente seguirá usando la librería agricolae y sólo un 4% no está seguro. Ver la figura 4.6.

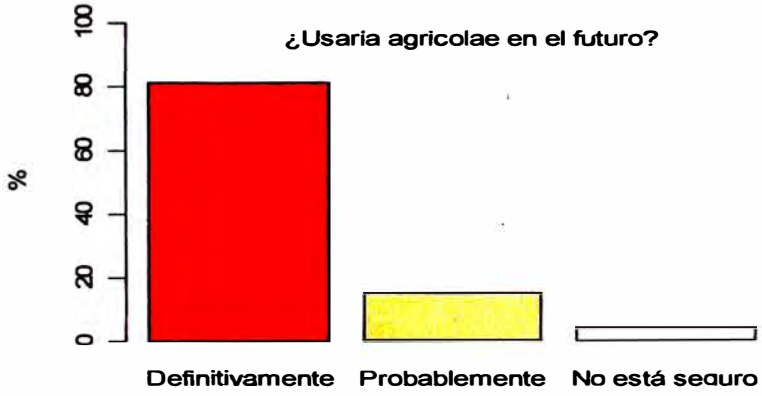


Figura 4.6. Potencial de usuarios de agricolae

P15.- ¿Recomendaría usar agricolae a los colegas o contactos de su institución?

```
> total<-table(T1[,5])
> P15<- round(total*100/sum(total),0)
> par(mar=c(2,4,2,2),cex=0.9)
> barplot(P15, col=colors()[c(81,16)], ylim=c(0,100), ylab="%")
> text(1.5,90,"¿Recomendaría usar agricolae\na colegas o contactos de su institución?")
```

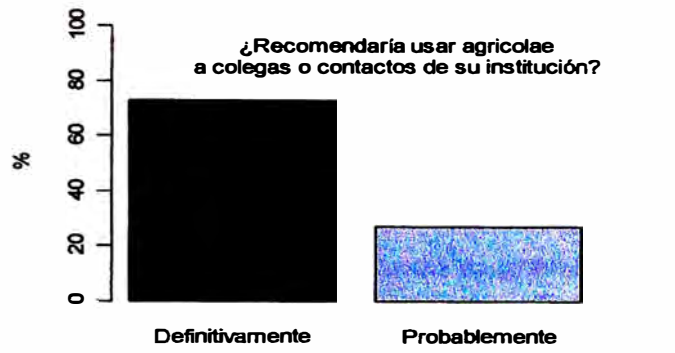


Fig. 4.7. Porcentaje de encuestados que recomiendan el uso

Según la figura 4.7, más del 60% de usuarios de agricolae recomendaría el programa a otros.

4.2.9 OPINIÓN DE ENCUESTADOS

P8: ¿Qué le gusta de la librería agricolae?

Es funcional, amigable, versátil, bastante útil, rapida en el proceso, novedosa, y provee soluciones estadísticas que otros paquetes no disponen. Es exclusiva para la agricultura, ademas libre y con practicidad y contenido.

P9.-¿Qué le disgusta de la librería agricolae?

Proporciona poca información sobre los datos de ejemplo; el inicio es dificultoso, algunas veces; requiere de muchos comandos, y la forma de los datos para usar las funciones, así como la forma de presentar sus resultados son complejas.

P10.- ¿Qué funciones le gustaría que incorpore agricolae en el futuro?

Estadística espacial, climatodiagrama, biodiversidad, estabilidad fenotípica (Eberhart and Russell), econométricas, redes neuronales.

P13.- ¿Por qué no usaría agricolae en el futuro?

Porque existen otros software más simples; a agricolae le falta funciones para microarrays, por lo complicado, además de ser compleja en su uso. Si deja de ser libre, requiere de mucha dedicación.

P14.- ¿Por qué usaría agricolae en el futuro?

Porque se actualiza, es gratuito, tiene elementos más amigables y versátiles que otros sofware. Tiene funciones que yo necesito, es competitivo, rápido, y de aplicación práctica, en sus resultados.

P16.- ¿Por qué no recomendar agricolae?

Por conocimiento del R; por la dedicación, requiere de mucho tiempo para familiarizarse. No hay razon por no recomendarlo.

4.2.10 OPINIÓN DE EXPERTOS

En el Anexo 4 se describe a los encuestados con el grado de PhD, especialísta en áreas agrícolas, y conocedores de la librería agricolae, y su opinión completa en inglés.

P8 - ¿Qué le gusta de la librería agricolae?

Es un paquete de conceptos para la agricultura, con diseños experimentales, tratamiento de Genotipo*ambiente, facilidad de uso, e importante para la mejora de la investigación de cultivos en los países en desarrollo sin acceso a costosos programas. Ahora se puede añadir los experimentos de campos al open office y R; presenta una comparación múltiple paramétrica así como no paramétrica; existe un ahorro de tiempo al hacer el análisis de los datos; hay buenos resultados; y el apoyo a través de la respuesta es excelente.

P9 - ¿Qué le disgusta de la librería agricolae?

El redondeo de decimales; la estimación de parámetros debe darse en un número suficiente de decimales sobre todo si estos valores se pueden utilizar en otros cálculos; no se puede guardar los resultados en un objeto; un diseño alfa bueno resulta dificil de analizar; la idea de presentar los diseños, por ejemplo, dado por Cochran y Cox es buena, pero también, se debe mostrar la forma de analizarlos. Ladescripción de los datos es mala; las funciones deben ser mejor explicadas. El usuario sin conocimientos profundos sobre los temas correspondientes puede tener problemas en la comprensión de las funciones. La sintaxis en la descripción es compleja. Según mi uso agricolae es bueno, Es muy importante tener tan encuenta el análisis dialélico es tan importante como LxT para un mejorador.

P10 - ¿Qué funciones le gustaría que incorpore agricolae en el futuro?

Análisis de modelos mixtos, análisis espacial, de medidas repetidas, comparación de medias y accesorios con diferentes estructuras de varianza-covarianza, diseños desequilibrados (multi-entorno, ensayos para comparar diversas variedades), medias ajustadas, prueba de Newman-Keuls, mezcla de modelos especialmente para la serie de experimentos y parcelas divididas, utilizar algunos procedimientos del paquete SASmixed para analizar los datos, experimentos dialelicos, cartografía genética y fuerte análisis de QTL (MI, CIM y MIM).

P13 - ¿Por qué no usaría agricolae en el futuro?

Porque no todo lo que necesito se encuentra en el paquete. No se incluye algunos análisis.

P14 - ¿Por qué usaria agricolae en el futuro?

Por lo fácil de usar; hay muchas funciones en el mismo paquete; existe una posible mejora en cada versión; hay énfasis en la experimentación agrícola. Esto es muy importante para mí. Es libre uso.

P16 - ¿Por qué no recomendaría el uso de Agricolae?

Porque no todo el mundo está familiarizado con R.

4.2.11 PUNTAJE DE LAS RESPUESTAS DE EXPERTOS Y OTROS SEGÚN LAS VARIABLES IDENTIFICADAS PARA MEDIR LA CALIDAD

Expertos:

```
> filas<-c(39,32,43,42,31,33,45,41,46,47,48)
> columnas<-c(19,20,2,17,4)
> T3[filas, columnas]
```

	P6_4	P6_5	P4_1	P6_2	P4_3
39	3	4	4	5	5
32	3	3	3	3	3
43	5	4	5	4	5
42	5	4	4	5	4
31	4	4	4	5	4
33	4	4	4	4	5
45	3	5	5	5	5
41	4	3	5	5	5
46	4	4	5	4	5
47	-	4	5	5	5
48	5	5	5	5	5

Escala 1-5 (promedio)

	P6_4	P6_5	P4_1	P6_2	P4_3	Promedio
Experto	4.0	4.0	4.45	4.54	4.63	4.32
Otros	3.56	4	4.14	4.08	4.09	3.97

Escala 0-1 (promedio)

	P6_4	P6_5	P4_1	P6_2	P4_3	Promedio
Experto	0.75	0.75	0.86	0.88	0.9	0.83
Otros	0.64	0.75	0.79	0.77	0.77	0.74

Según la escala 0-1 (Sadana, 2006), la librería agricolae está categorizada como un software de calidad **MUY BUENA,** sin considerar la opinión subjetiva presentada que también es muy favorable.

4.2.12 CHEQUEO DE AGRICOLAE POR EL PROYECTO R

Se envió la nueva versión corregida al proyecto R el 20 de abril del 2009. La evaluación esta publicada en la pagina del proyecto R en la dirección de Internet siguiente:

http://cran.at.r-project.org/web/checks/check_results_agricolae.html

como se muestra en la tabla 4.11, con la descripción siguiente:

"CRAN Package Check Results for Package agricolae
Last updated on 2009-04-25 23:51:08"

Tabla 4.11. Resultado del chequeo por el Proyecto-R

Flavor	Version	$T_{install}$	T_{check}	T_{total}	Status
r-devel-linux-ix86	1.0-7	5.94	63.30	69.24	OK
r-devel-linux-x86_64-gcc	1.0-7	8.04	85.73	93.77	OK
r-devel-linux-x86_64-sun	1.0-7				OK
r-patched-linux-ix86	1.0-7	5.88	65.19	71.07	OK
r-patched-linux-x86_64	1.0-7	8.02	86.00	94.02	OK
r-release-linux-ix86	1.0-7	5.60	63.50	69.10	OK
r-release-windows-ix86	1.0-7	20.00	77.00	97.00	OK

4.2.13 PUBLICACIÓN DE AGRICOLAE EN EL PROYECTO R

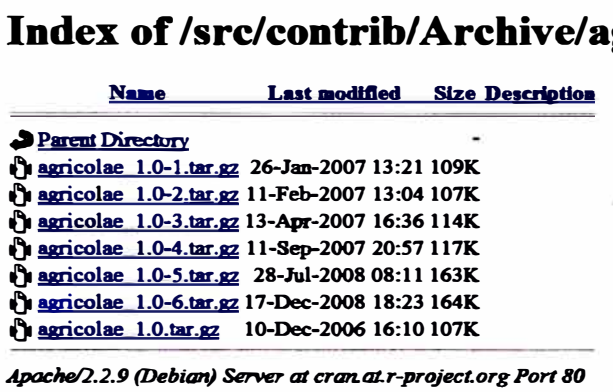
Aceptada la librería, el Proyecto R hace la difusión creando los instaladores en Windows y Mac. El lugar para descargar está en el rubro Packages de la página de R-Project. La ubicación exacta es:

http://cran.at.r-project.org/web/packages/agricolae/index.html

Historia de la librería agricolae en el proyecto R

Desde su primera versión 1.0, del 10 de diciembre del 2006, el proyecto R lleva el registro de todas sus versiones como puede verse en la figura 4.7 y en la dirección por INTERNET:

http://cran.at.r-project.org/src/contrib/Archive/agricolae



Index of /src/contrib/Archive/agricolae

Name	Last modified	Size	Description
Parent Directory		-	
agricolae_1.0-1.tar.gz	26-Jan-2007 13:21	109K	
agricolae_1.0-2.tar.gz	11-Feb-2007 13:04	107K	
agricolae_1.0-3.tar.gz	13-Apr-2007 16:36	114K	
agricolae_1.0-4.tar.gz	11-Sep-2007 20:57	117K	
agricolae_1.0-5.tar.gz	28-Jul-2008 08:11	163K	
agricolae_1.0-6.tar.gz	17-Dec-2008 18:23	164K	
agricolae_1.0.tar.gz	10-Dec-2006 16:10	107K	

Apache/2.2.9 (Debian) Server at cran.at.r-project.org Port 80

Figura 4.7. Historia de los programas fuente de agricolae

4.2.14 RECONOCIMIENTO EN LAS LIBRERIAS DE DISEÑO DE EXPERIMENTOS

En la sección de "ExperimentalDesign" de la página del proyecto R se hace mención a la librería agricolae. La persona responsable es la Dra. Ulrike Groemping, quien describe a la librería de esta forma:

"Experimental designs for agricultural and plant breeding experiments

agricolae offers extensive functionality on experimental design especially for agricultural and plant breeding experiments, which can also be useful for other purposes. It supports planning of lattice designs, factorial designs, randomized complete block designs, completely randomized designs, (Graeco-)Latin square designs, balanced incomplete block designs and alpha designs. There are also various analysis facilities for experimental data, e.g. treatment comparison procedures and several non-parametric tests, but also some quite specialized possibilities for specific types of experiments."

4.2.15 AGRICOLAE EN EL CORE

En la sección de "ExperimentalDesign", la librería está calificada dentro del "core" como se indica a continuación:

- agricolae (core)
- AlgDesign (core)
- BHH2 (core)
- blockTools
- BsMD
- conf.design
- crossdes
- desirability
- experiment
- FrF2 (core)
- granova
- ldDesign
- lhs
- qtlDesign
- rsm (core)
- SensoMineR

4.3 DISCUSIÓN DE RESULTADOS

El proyecto R, con el lenguaje de programación de fuente abierta y la documentación de todo el sistema, facilitó la creación, chequeo y puesta en marcha de la librería agricolae con funciones estadísticas aplicadas a la investigación agrícola. La administración del proyecto lleva el registro de todos los programas fuente desde su primera versión y se mantiene al día con los cambios, lo que da confianza suficiente para la continuidad en el uso del sistema.

La encuesta de satisfacción fue aceptada y comprendida por los usuarios. El retorno de la encuesta tardó 6 meses en alcanzar el mínimo aceptable (> 44). Las preguntas hechas permitieron obtener medidas sobre las siguientes variables:

- Conocimiento del usuario,
- Satisfacción de la librería,
- Importancia en el uso,
- Experiencia del usuario, y
- Uso futuro.

Los ítems correspondientes a estas variables lograron medir la calidad del software. Según el análisis de Likert se detectó los ítems que permitieron calificar la satisfacción de la librería; éstos son: la calidad, la integración con R, la credibilidad, la competencia, y el uso. El puntaje de 0.752 en la escala de 0-1 permitieron calificar el software como MUY BUENO, y 0.88 para el índice de Cronbach, lo cual permitió dar fiabilidad al conjunto de items para medir la satisfacción. La descripción del software por los usuarios permitió evaluar la calidad del software en el uso de las funciones específicas a través de sus comentarios, incluidos la de los expertos, que

fueron muy favorables y con sugerencias para mejorar la librería en el futuro. Sobre las comparaciones entre las funciones de agricolae y los programas como SAS, SPSS y MINITAB, se obtuvo resultados similares. El reconocimiento de agricolae en el proyecto R es una garantía sobre su contenido y uso en R.

CAPITULO V

CONCLUSIONES Y RECOMENDACIONES

5.1 CONCLUSIONES

1. La librería agricolae es de libre uso, bajo la licencia GPL como lo está el programa R y está diseñada para el uso dentro de R. La librería está publicada en el proyecto R y en su descripción especifica "License: GPL".
2. La librería agricolae es un conjunto de funciones estadísticas aplicadas para la investigación agrícola, disponible en INTERNET http://cran.at.r-project.org/web/packages/agricolae/index.html, en versiones para Windows, Linux y MAC, y satisface los requerimientos de calidad propuestos por el modelo McCall.
3. Según la hipótesis general, se acepta que la librería cumple con los términos de calidad de software planteados por el modelo de McCall a través de su operatividad, revisión y transición del software, probado mediante la encuesta de satisfación con la escala de Likert, el índice de Cronbach, y el chequeo de software realizado por el programa CHECK de R, además del registro histórico de las fuentes del programa en el proyecto R.
4. Según las hipótesis específicas, agricolae es funcional en el programa R y es libre bajo la licencia GPL como se manifiesta en su publicación en INTERNET y como lo es el programa R donde se instala para su uso. Sobre los resultados comparativos de funciones específicas, se encontró similitud en los resultados de agricolae con los programas SAS, SPSS y MINITAB.
5. La librería es funcional y está probada con los ejemplos de datos experimentales proporcionados por el Centro Internacional de la Papa.

6. Los tipos de función más utilizados fueron "Diseño Experimental", con 80%, y "comparaciones múltiples", con un 66.6%. Respecto a la calificación en la escala de 0-1, las funciones de estabilidad alcanzaron el valor más alto con 0.85 y las otras funciones, valores superiores a 0.75. Según la prueba de t-student para la media de la escala mayor a 0.5 para los tipos de función, la prueba resultó altamente significativa (p.valor < 0.01), lo que satisface su uso.
7. Según 48 encuestas y 21 ítems de Likert, con un puntaje de 1 a 5, la encuesta alcanzó 4076 de un máximo de 5040, dando un factor de calidad de 0.808 (Muy Bueno).
8. Según la escala de Likert, los ítems de la encuesta que mejor calificaron para el estudio de satisfacción fueron:

   P4_1 Satisfecho con la calidad (0.79)

   P4_3 Integración con R (0.79)

   P6_2 Agricolae hace lo que dice (0.77)

   P6_5 Agricolae es competitivo (0.73)

   P6_4 Agricolae es fácil de usar (0.66)

   En la escala de 0 a 1, los puntajes obtenidos satisfacen la calidad del software; el puntaje promedio de estos ítems dado por los expertos, alcanzó el valor de 0.83 y en cuanto al puntaje de otros usuarios 0.74.
9. Según el índice de alfa de Cronbach de 0.88, los ítems considerados en la encuesta es fiable, siendo el minimo aceptable 0.80.
10. La librería agricolae satisface al proyecto R como se puede ver en la página del proyecto R:

    http://cran.at.r-project.org/web/packages/agricolae/index.html.

    La librería fue sometida al control de calidad del proyecto R y pasó todas las pruebas, fue seleccionada en el CORE, y esta nombrada en el rubro de librerías para diseño de experimentos:

    http://cran.at.r-project.org/web/views/ExperimentalDesign.html

5.2 RECOMENDACIONES

1. Como la librería agricolae se encuentra en un repositorio público y es una herramienta de servicio a la comunidad, el mantenimiento debe ser continuo y se debe asumir con responsabilidad frente a los cambios futuros y sin costo alguno como se estipula en la licencia GPL.
2. Con respecto a mejoras, según la opinión de los evaluadores, la librería agricolae necesita mejorar en la documentación de los datos, el incremento de nuevos métodos, la simplificación de sus procedimientos y la optimización de los recursos de programación.
3. Sobre la metodología para el análisis de calidad del software, es una buena alternativa el uso de una encuesta de satisfacción y es recomendable disponer de evaluadores expertos sobre el uso.
4. Según la experiencia en esta investigación, el medio más adecuado para solicitar el llenado de la encuesta es vía correo electrónico a un gran número de posibles usuarios, y en INTERNET en un lugar accesible.
5. Si se va a desarrollar funciones estadísticas aplicadas a la investigación en general, el proyecto R es un buen medio y sin costo alguno; permite la participación de una comunidad en su proyecto y así, mejorar el software.
6. La librería agricolae tiene la licencia GPL; esto significa que no puede ser utilizada con fines de lucro.

GLOSARIO DE TÉRMINOS

Aditividad.- Linealidad de los parámetros en el modelo.

agricolae.- Del latín, agricultor.

Aleatorización.- Principio básico del diseño, correspondiente a la asignación del tratamiento en la unidad experimental.

Alfa de Cronbach.- Indicador de la fiabilidad del instrumento, utilizado en análisis de encuestas.

AMMI.- Modelo de efectos aditivos e interacción multiplicativa.

AMMI.contour.- Función de agricolae que permite construir líneas de contorno en el biplot.

audpc.- Área del progreso de la enfermedad en un tiempo de evaluación. Función de agricolae.

Balance, Balanceado.- Referido al diseño. Los tratamientos alcanzan una determinada equidad en la comparación, la cual puede ser parcial o completa.

bar.err.- Función de agricolae para el diagrama de barras y su desviación o error estándar.

bar.group.- Función de agricolae para el diagrama de barras y su semejanza o diferencia entre tratamientos mediante letras.

BIB.test.- Función de agricolae para el análisis de tratamientos en un diseño de bloques incompletos balanceado.

Biodiversidad.- Llamada *biodiversidad biológica* referida a la amplia variedad de seres vivos.

index.bio.- Función de agricolae que halla los límites de confianza de los índices de biodiversidad.

Biplot.- Gráfico en un plano de dos componentes principales, en el que se expresa dos efectos de un modelo. Por ejemplo ambientes y genotipos.

Bloque.- Grupo de unidades experimentales homogéneas.

Clúster o conglomerado.- Grupo de unidades homogéneas.

Componentes principales.- Variables independientes formadas por la combinación de todas las variables de estudio. Se manifiestan en orden de importancia.

consensus.- Función de agricolae para el estudio del consenso de un dendrograma de individuos o unidades multivariables que se expresan en forma semejante con cierto grado.

correlation.- Función de agricolae para el cálculo y significación de las correlaciones entre variables para tres métodos (Pearson, Spearman y Kendall)

Cuadrados medios(CM).- Medida de variación promedio de un efecto en un modelo lineal.

cv.model.- Función de agricolae, que calcula el coeficiente de variación del experimento.

Dendrograma.- Diagrama jerárquico de semejanza entre individuos.

design.ab.- Función de agricolae para hallar el diseño de bloques de un factorial pq.

design.alpha.- Función de agricolae para hallar el diseño alfa.

design.bib.- Función de agricolae para hallar el diseño de bloques incompletos balanceado.

design.crd.- Función de agricolae para hallar el diseño completamente al azar.

design.cyclic.- Función de agricolae para hallar el diseño cíclico.

design.graeco.- Función de agricolae para hallar el diseño greco latino.

design.lattice.- Función de agricolae para hallar el diseño latice.

design.lsd.- Función de agricolae para hallar el diseño cuadrado latino.

design.rcbd.- Función de agricolae para hallar el diseño de bloques completos al azar.

Diseño alfa.- Diseño de bloques incompletos parcialmente balanceado similar al latice pero rectangular. El diseño alfa es un diseño resoluble.

Diseño bloques completos e incompletos.- Diseño en donde los tratamientos están aleatoriamente distribuidos, completos o una parte en cada bloque.

Diseño Carolina.- Diseño genético de los modelos I, II y III.

Diseño cíclico.- Diseño bloques incompletos con una sustitución cíclica de tratamientos a partir de un generador.

Diseño cuadrado latino.- Diseño que controla el material experimental por doble bloqueo y asigna aleatoriamente los tratamientos en ambos bloqueos una sola vez.

Diseño genético.- Diseño en el que los tratamientos se forman en base a progenies y progenitores.

Diseño greco.- Diseño que controla el material experimental por doble bloqueo y asigna aleatoriamente de 2 tipos de tratamientos en ambos bloqueos una sola vez. Todos los factores son ortogonales.

Diseño latice.- Diseño resoluble, puede ser "simple" si presenta dos repeticiones o "triple", si presenta tres repeticiones.

Diseño resoluble.- Grupos de pequeños bloques que conforman un bloque completo de tratamientos. Cada bloque completo es una repetición.

durbin.test.- Función de agricolae para la prueba de Durbin.

Durbin.- Prueba no-paramétrica para los diseños de bloques incompletos balanceado.

Escala de Likert.- Función utilizada para medir aptitudes en una escala de menos a más. La escala es numérica (1 – 5), donde 5 representa al más favorable.

Estabilidad.- Capacidad de los cultivos de mantenerse estables en diferentes ambientes.

fact.nk.- Función de agricolae para construir un diseño de bloques de un factorial pxq.

Fenotipo.- Union de las palabras "ambiente" y "genotipo".

friedman.- Función de agricolae para la prueba no paramétrica en la comparación de tratamientos agrupados en grupos diferentes. Puede ser utilizada para datos provenientes de un diseño de bloques completo al azar.

Generadores.- Arreglos básicos para la generación de diseños.

Genotipo.- Contenido genético de un individuo en forma de ADN.

GLP.- Del inglés "General Public License".

GNU.- Licencia pública para usar libremente el software y protegerse del intento de apropiación.

Grado de libertad.- Parámetro de variables aleatorias relacionadas a la variancia. Libertad de escoger un número de datos para hallar su variancia dado su promedio.

graph.freq.- Función de agricolae para gráfico de frecuencia.

grid3p.- Función de agricolae para la interpolación de una matriz cuantitativa de información. Útil para completar datos faltantes.

hcut.- Función de agricolae para cortar un dendrograma construido con la función consensus().

Heredabilidad.- Proporción de la variación fenotípica de una población. Atribuible a la variación genotípica entre individuos.

Heterosis.- Capacidad de transmitir genes de padres a hijos para cierta característica deseada (resistencia, rendimientos, etc.).

hgroups.- Función de agricolae, para extraer los individuos que conforman grupos del dendrograma, el cual es producido por la función concensus().

HSD.test.- Función de agricolae para la prueba de Tukey. Permite la formación de grupos de tratamientos similares.

index.bio.- Función de agricolae para el cálculo de los índices de biodiversidad y sus límites de confianza.

kruskal.- Función de agricolae para la prueba no paramétrica en la comparación de muestras de diferentes grupos. Puede ser usado para datos provenientes de un diseño completo al azar.

Librería.- Conjunto de funciones integradas para un propósito en un lenguaje de programación.

LineXtester.- Análisis completo del cruzamiento genético línea por probador.

LSD.test.- Función de agricolae para la prueba de diferencia minima de significación. Permite la formación de grupos de tratamientos similares, el ajuste por bonferroni y otros.

Mínimos cuadrados ponderados.- Método estadístico para la estimación de modelos lineales generalizados donde existe heterogeneidad de varianza y se puede corregir por una ponderación de la variable respuesta en una función de las predictoras.

Modelo estadístico.- Expresión de relación funcional de predictoras a una respuesta aleatoria.

Modelo McCall.- Formulación de condiciones para el estudio de la calidad de software, en base a tres ejes: operación, revisión y transición del producto.

Montecarlo.- Método de simulación a partir de la distribución acumulativa. Útil para generar números aleatorios de distribuciones no conocidas de la variable.

nonadditivity.- Función de agricolae para la prueba de no aditividad del modelo.

normal.freq.- Función de agricolae para la gráfica de la normalidad de datos agrupados.

Normalidad.- Condición de los datos sobre la distribución normal de probabilidades.

ojiva.freq.- Función de agricolae para graficar la ojiva de datos agrupados.

Ortogonalidad.- Similar al concepto matemático sobre el producto escalar igual a cero.

Parcialmente balanceado.- Expresión utilizada cuando los tratamientos están distribuidos de tal forma que no tienen igual eficiencia en sus comparaciones, a pesar de tener igual repetición.

PBIB.test.- Función de agricolae para la prueba de bloques incompletos balanceados.

plot.graph.freq.- Función genérica de agricolae para reproducir el histograma con solo utilizar el término plot.

polygon.freq.- Función de agricolae para construir el polígono de frecuencia.

Producto de Kronecker.- Producto matricial denotado por $\otimes$. Da como resultado una matriz bloque. Cada elemento de una matriz se multiplica por toda la matriz del segundo. Si A y B son matrices de orden 2, $A \otimes B$, es de orden 4.

P-valor.- Representa la probabilidad de rechazo de la hipótesis planteada.

Regresión.- Procedimiento estadístico para predecir una respuesta aleatoria en función de predictores medidos.

Re-muestreo.- Procedimiento repetido de muestreo para el estudio del comportamiento del mismo con fines estadísticos. Por ejemplo, estimación y prueba sobre parámetros de un modelo.

Repetición.- Realización del experimento por segunda vez con el fin de observar la variabilidad de la respuesta. Utilizada para minimizar el error experimental y para predecir su variancia de error.

Reproducibilidad.- Proceso de repetición.

resampling.model.- Función de agricolae para realizar prueba de un modelo lineal mediante re-muestreos.

Robusto.- Método de estimación en el cual no se ve afectado por valores extremos.

simulation.model.- Función de agricolae para simular el comportamiento de un modelo bajo el supuesto de normalidad del error.

Spearman.- Es un método de correlación no paramétrica. Utiliza el valor del orden de los datos.

Split plot.- Diseño experimental en donde la unidad experimental es dividida en sub unidades para la aplicación de un factor a pequeña escala.

stability.nonpar.- Función de agricolae para el estudio de la estabilidad de cultivos para datos seudo cuantitativos (cualitativos jerárquicos).

stability.par.- Función de agricolae para el estudio de la estabilidad de cultivos para datos cuantitativos.

stat.freq.- Función de agricolae para el cálculo de estadísticas de una tabla de frecuencia.

strip.plot.- Función de agricolae para el análisis del diseño bloques divididos.

sturges.freq.- Función de agricolae para encontrar intervalos de frecuencia según el método de Sturges.

summary.graph.freq.- Función genérica de agricolae para la función graph.freq() para presentar la tabla de frecuencia con el uso de summary().

table.freq.- Función de agricolae para presentar la tabla de frecuencia de un objeto construido con graph.freq() o hist().

tapply.stat.- Función de agricolae para construir estadísticas de datos agrupados.

Triplot.- Gráfico en un plano cartesiano con tres componentes principales.

waerden.test.- Función de agricolae para la prueba de Warden cuando los valores obtenidos seudo-cuantitativa tienen un comportamiento asimétrico como una normal.

waller.- Función de agricolae para calcular el valor tabular de Waller-Duncan.

waller.test.- Función de agricolae para la prueba comparativa de tratamientos mediante la prueba bayesiana de Waller-Duncan.

wxyz.- Función de agricolae para completar los valores perdidos mediante un modelo por efecto de dos variables (x,y). Z=f(x,y) un modelo lineal.

BIBLIOGRAFIA

1. Box, George E. P. (2005) Statistics for Experimenters Design, Innovation, and Discovery Second Edition.. Wiley-Interscience.
2. Calzada B. (1960). Métodos estadísticos para la investigación. Lima-Peru.
3. Cochran and Cox.(1992). Experimental design. Second edition. Wiley Classics Library Edition published. John Wiley & Sons, INC.
4. Cochran, W. (1972) Tecnicas de Muestreo, C.E.C.S.A. Mexico
5. Conover, W.J. (1999) Practical Nonparametrics Statistics. John Wiley & Sons, INC, New York.
6. Efron, B., Tibshirani, R. (1993) An Introduction to the Bootstrap. Chapman and Hall/CRC
7. Kwanchai, A. (1984) Statistical procedures for agricultural research.
8. Good, Phillip (2001). Resampling Methods, second edition, Birkhauser, Boston.
9. Hayes, Bob E. (2008) Measuring Customer Satisfaction and Loyalty,: Survey Design, Use, and Statistical Analysis Methods. Third Edition
10. Haynes K G, Lambert D H, Christ B J, Weingartner D P, Douches D S, Backlund J E, Fry W and Stevenson W. (1998). Phenotypic stability of resistance to late blight in potato clones evaluated at eight sites in the United States American Journal Potato Research 75, pag 211-217
11. Joshi, - D.D. (1987) Linear Estimation and Design of Experiments.. WILEY EASTERN LIMITED, New Delhi, India
12. Kang, M. S. (1993). Simultaneous selection for yield and stability: Consequences for growers. Agron. J. 85:754-757
13. Kuehl, Robert (2000) Design of Experiments. 2nd ed., Duxbury
14. LeClerg, Erwin (1962). Field Plot Technique, Burgess Publishing Company

15. Magurran, A.E. (1988). Ecological diversity and its measurement. Princeton University Press.
16. Patterson, H.D. and Williams, E.R. Biometrika (1976) A new class of resolvable incomplete block designs. printed in Great Britain.
17. R Development Core Team (2008). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org
18. R Development Core Team (2008). Writing R Extensions. Vienna, Austria. ISBN 3-900051-11-9
19. Sadana, Vishal (2006). "A Survey Based Software Quality Model". Department of Computer Science, University of Missouri-Rolla. Rolla, MO 65401: http://www.secc.org.eg/SEPG%202006/Ingredients/Presentations/215/215_1.html
20. Steel & Torry & Dickey (1997) Principles and procedures of statistics a biometrical approach. Third Edition
21. Singh R. K., Chaudhary, B.D. (1979). Biometrical Methods in Quantitative Genetic Analysis. Kalyani Pubishers New Delhi-Ludhiana.
22. Waller, R. A. and Duncan, D. B. (1969). A Bayes Rule for the Symmetric Multiple Comparison Problem, Journal of the American Statistical Association 64, pages 1484-1504.
23. Waller, R. A. and Kemp, K. E. (1976) Computations of Bayesian t-Values for Multiple Comparisons, Journal of Statistical Computation and Simulation, 75, pages 169-172.
24. Weikai Yan and Manjit S. Kang (2003) GGE Biplot Analysis: A graphical tool for breeder, geneticists, and agronomists.. www.crepress.com.
25. Williams, E.R. (1977). Iterative Analysis of Generalizad Lattice Designs. Austral J. Statistics 19(1) 39-42.

ANEXOS

ANEXO 1. UBICACIÓN Y DESCRIPCIÓN DE LA LIBRERÍA AGRICOLAE EN INTERNET

Agricolae esta localizada en la página del proyecto R: http://www.r-project.org

Acceso directo: http://cran.at.r-project.org/web/packages/agricolae/index.html

La versión 1.0-7 fue publicada en internet el 20 de abril del 2007 con la siguiente descripción:

Suggest: librerías necesarias para la funcionalidad completa de agricolae.

Licencia: GPL

In Views: área de aplicación de la librería, Diseños de experimentos.

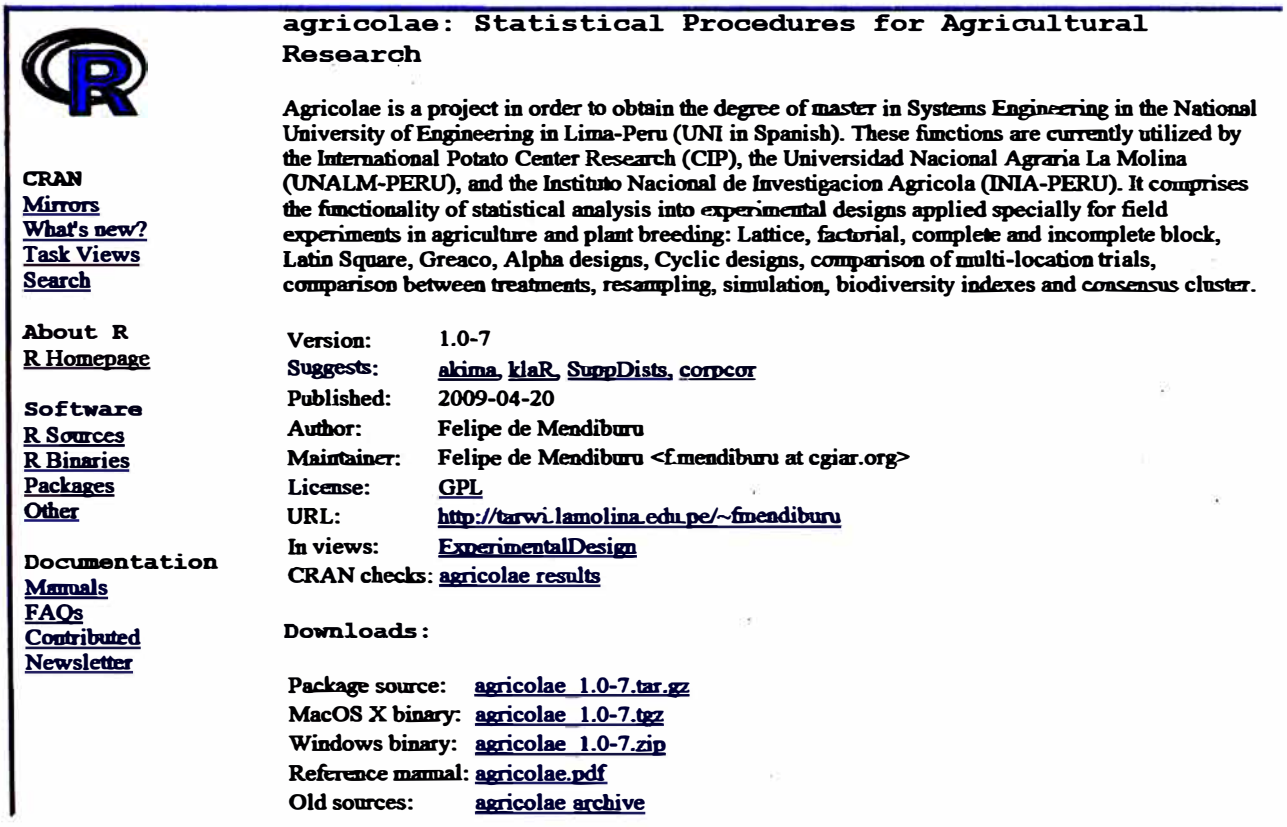
CRAN Checks: Información del chequeo de la librería en las diferentes plataformas. La librería paso todas las pruebas>

Package Source: Las fuentes disponibles para un uso mas avanzado de las funciones, el archivo esta disponible para la inhalación en Linux.

MacOX X Binary: Para instalar en Mac.

Windows Binary: Para instalar en Windows.

Reference Manual: el manual de la descripción de las funciones y datos y uso de las funciones con ejemplos.



CRAN
Mirrors
What's new?
Task Views
Search

About R
R Homepage

Software
R Sources
R Binaries
Packages
Other

Documentation
Manuals
FAQs
Contributed
Newsletter

agricolae: Statistical Procedures for Agricultural Research

Agricolae is a project in order to obtain the degree of master in Systems Engineering in the National University of Engineering in Lima-Peru (UNI in Spanish). These functions are currently utilized by the International Potato Center Research (CIP), the Universidad Nacional Agraria La Molina (UNALM-PERU), and the Instituto Nacional de Investigacion Agricola (INIA-PERU). It comprises the functionality of statistical analysis into experimental designs applied specially for field experiments in agriculture and plant breeding: Lattice, factorial, complete and incomplete block, Latin Square, Greaco, Alpha designs, Cyclic designs, comparison of multi-location trials, comparison between treatments, resampling, simulation, biodiversity indexes and consensus cluster.

Version:	1.0-7
Suggests:	akima, klaR, SuppDists, corpcor
Published:	2009-04-20
Author:	Felipe de Mendiburu
Maintainer:	Felipe de Mendiburu <f.mendiburu at cgiar.org>
License:	GPL
URL:	http://tarwi.lamolina.edu.pe/~fmendiburu
In views:	ExperimentalDesign
CRAN checks:	agricolae results

Downloads:

Package source:	agricolae_1.0-7.tar.gz
MacOS X binary:	agricolae_1.0-7.tgz
Windows binary:	agricolae_1.0-7.zip
Reference manual:	agricolae.pdf
Old sources:	agricolae archive

ANEXO 2. ENCUESTA DE SATISFACCIÓN

Por favor, tómese unos minutos para completar la encuesta de satisfacción de la librería agricolae en R. Su opinión es muy importante. Sus respuestas ayudarán a comprender la importancia y problemas que pueda tener, así también orientar las funciones y la base de datos que contiene. Sus respuestas se mantendrán confidenciales y solo se utilizara para fines de este estudio.

Esta encuesta le tomará aproximadamente 10 minutos para completar

P1.- ¿Cuánto tiempo hace que utiliza la librería AGRICOLAE?

Menos de 1 mes :
de 1 a 6 meses :
Mas de 6 meses a 1 año :
Mas de 1 año :

P2. En el último año, ¿cuál de las siguientes areas de funciones estadisticas ha utilizado? Por favor, marque todas las que correspondan

Diseño de experimentos :
Comparaciones multiples :
AMMI y estabilidad :
Estadistica Descriptiva :
Simulacion y re-muestreo :
Otras :

P3. En general, ¿cómo es su satisfacción respecto a las siguientes areas de funciones?

5 =Muy satisfecho, 4 Satisfecho, 3=Neutral, 2= Insatisfecho, 1= Muy insatisfecho

Diseño de experimentos :
Comparaciones multiples :
AMMI y estabilidad :
Estadistica Descriptiva :
Simulacion y re-muestreo :

P4. ¿Qué tan satisfecho está usted con las siguientes características de AGRICOLAE?

5 =Muy satisfecho, 4 Satisfecho, 3=Neutral, 2= Insatisfecho, 1= Muy insatisfecho

Calidad :
Tiempo de respuesta :
Integracion con R :
Instalacion de la librería :
Uso de ejemplos :
El manual :
Procesar con sus datos :

P5. ¿Qué tan importantes son las siguientes características en el uso de la libreria?

5 =Muy importante, 4=Importante, 3=Neutral, 2=Algo importante, 1= Nada Importante

Calidad :
Presentacion de resultados :
Sin costo :
Instalacion por primera vez :
Conocimientos de agricultura :
Conocimientos de estadistica:
Reproducir con sus datos :

P6. Al pensar de su más reciente experiencia con la libreria, ¿qué tanto está de acuerdo con las siguientes afirmaciones?

5 =Totalmente de acuerdo, 4 = De acuerdo, 3=Neutral, 2=En desacuerdo, 1 Totalmente en desacuedo

Agricolae es funcional :
Agricolae hace lo que dice :
Agricolae hace lo que yo necesito:
Agricolae es facil de usar :
Agricolae es competitivo :

P7. ¿Qué tan importantes son cada uno de los siguientes grupos de funciones para usted?

5 =Muy importante, 4=Importante, 3=Neutral, 2=Algo importante, 1= Nada Importante

Diseño de experimentos :
Comparaciones multiples :
AMMI y estabilidad :
Estadistica Descriptiva :
Simulacion y re-muestreo :

P8.- ¿Qué le gusta de la librería agricolae?

P9.- ¿Qué le disgusta de la librería agricolae?

P10.- ¿Qué funciones le gustaría que incorpore agricolae en el futuro?

P11.-Piense en productos similares ofrecidos por otros programas estadisticos, ¿cómo comparar los procedimientos ofrecidos por agricolae dentro del programa R?

Mejor :
Algo mejor :
Es lo mismo :

Algo peor :
Peor :
No sabe :

P12.-¿Usaria agricolae en el futuro?

Definitivamente :
Probablemente :
No está seguro :
Probablemente no :
Definitivamente no :

P13.-¿Por qué no usaría agricolae en el futuro?

P14.-¿Por qué usaría agricolae en el futuro?

P15.- ¿Recomendaría usar agricolae a los colegas o contactos de su institución?

Definitivamente :
Probablemente :
No está seguro :
Probablemente no :
Definitivamente no :

P16.-¿Por qué no recomendaría el uso de agricolae?

P17.-Software Estadistico que usa aparte del R:

SAS :
GENSTAT :
SPLUS :
MSTAT :
MINITAB :
SPSS :
STATA :
Otros :

Esta encuesta es casi completa. Estas últimas preguntas nos ayudaran analizar las respuestas.

P18.-Genero : (Varón/ Mujer)
P19.-Edad :
P20.-Educación : (Pre Grado/Maestría/ Doctorado/Otros)
P21.-Ocupación : (Estudiante/Profesor/Investigador)

Gracias por su tiempo y sus valiosos comentarios.
Por favor enviar a la dirección: f.mendiburu@cgiar.org

ANEXO 3. DATOS DE LA ENCUESTA

P11: Mejor (5), Algo Mejor (4), Es lo mismo (3), Algo peor (2)
P12, P15: Definitivamente (5), Probablemente (4), No esta seguro (3)
P18: Mujer (1), Varon (2), P1: Intervalo en meses del uso de agricolae

	P1	P11	P12	P15	P18	P19	P20	P21	Especialidad
1	12-24	4	5	5	2	56	Maestría	Investigador	Statistics
2	12-24	5	5	5	2	60	Maestría	Investigador	Agronomist
3	12-24	5	5	5	2	51	Maestría	Investigador	Agronomist
4	12-24	4	5	4	1	50	Maestría	Investigador	Entomologic
5	12-24	5	5	5	2	56	Maestría	Estudiante	Forestry
6	12-24	4	5	5	2	27	Superior	Investigador	Statistics
7	00-01	2	5	5	2	55	Maestría	Investigador	Agronomist
8	01-06	NA	5	5	2	28	Maestría	Investigador	Biologist
9	06-12	5	5	5	2	40	Maestría	Investigador	Agronomist
10	01-06	5	5	5	2	50	Maestría	Docente	Agronomist
11	06-12	2	5	5	1	29	Maestría	Investigador	Biologist
12	01-06	5	5	5	2	28	Maestría	Estudiante	Forestry
13	06-12	NA	5	5	2	36	Maestría	Estudiante	Genetics
14	12-24	4	5	5	2	30	Maestría	Docente	Statistics
15	01-06	5	5	5	2	28	Superior	Estudiante	Statistics
16	01-06	5	4	4	1	34	Maestría	Investigador	Forestry
17	01-06	5	5	5	2	29	Maestría	Estudiante	Forestry
18	12-24	NA	3	4	2	24	Superior	Estudiante	Statistics
19	12-24	4	4	4	2	54	Maestría	Investigador	Agronomist
20	12-24	3	4	4	2	52	Maestría	Investigador	Agronomist
21	00-01	4	5	5	1	24	Maestría	Investigador	Biologist
22	12-24	5	5	5	2	33	Maestría	Investigador	Environmental Engineer
23	00-01	5	5	5	2	52	Maestría	Investigador	Forestry
24	01-06	NA	3	4	2	37	Maestría	Investigador	Forestry
25	00-01	4	4	4	1	25	Maestría	Investigador	Forestry
26	01-06	5	5	5	2	32	Maestría	Docente	Forestry
27	00-01	2	4	4	1	24	Superior	Investigador	Forestry
28	01-06	3	5	5	1	53	Maestría	Investigador	Statistics
29	12-24	4	5	5	2	23	Superior	Estudiante	Statistics
30	01-06	5	5	5	2	38	Maestría	Estudiante	Geographer
31	06-12	5	5	5	2	46	PhD	Investigador	Statistics
32	12-24	4	5	5	2	64	PhD	Docente	Weed Science
33	12-24	4	5	4	2	56	PhD	Investigador	Canola Pathology
34	00-01	5	4	4	2	57	Superior	Docente	Physicist
35	00-01	NA	5	4	2	20	Superior	Estudiante	Statistics
36	00-01	NA	5	5	2	40	Maestría	Investigador	Forestry
37	00-01	NA	5	4	2	41	Maestría	Investigador	Forestry
38	06-12	5	5	5	2	32	Maestría	Investigador	Forestry
39	06-12	NA	5	5	2	32	PhD	Docente	Experimental Design and Bioinformatics
40	06-12	5	4	4	2	54	Maestría	Investigador	Agronomist
41	12-24	4	5	5	2	43	PhD	Investigador	Statistics
42	06-12	4	5	5	1	30	PhD	Investigador	Plant Ecology
43	06-12	5	5	5	2	32	PhD	Investigador	Corn Breeding
44	06-12	5	5	5	2	48	Maestría	Investigador	Statistics
45	00-01	4	5	5	2	39	PhD	Investigador	Biologist
46	01-06	NA	5	5	2	52	PhD	Docente	Earth and Environmental Sciences
47	12-24	5	5	5	2	42	PhD	Investigador	Virologist
48	12-24	3	5	5	2	45	PhD	Investigador	Agronomist

Especialidad: Información en el correo, enviado por cada usuario.

P2: Diseños Experimentales (1), Comparaciones multiples (2), AMMI estabilidad (3), Estadistica descriptiva (4) Simulación Re-muestreo (5), otros (6)
P3, P4: Muy satisfecho (5),.., Muy insatisfecho (1)
P5: Muy importante (5),.., Nada importante (1)
P6: Totalmente de acuerdo (5),.., Totalmente en desacuerdo (1)

	P2	P3
1	1	4
1	2	4
1	4	4
2	1	4
2	2	5
3	1	4
3	2	4
3	4	4
4	2	4
4	4	4
5	1	5
5	4	5
6	1	5
6	2	5
7	1	4
7	4	4
8	1	4
8	4	3
8	5	3
9	6	3
10	1	5
10	2	5
10	3	5
11	1	5
11	2	4
11	4	5
12	1	5
12	4	4

	P2	P3
13	2	3
13	4	5
14	1	4
14	2	5
14	4	4
14	5	3
15	1	5
15	2	5
16	1	4
16	2	4
16	3	4
16	4	4
16	5	4
17	1	4
17	2	4
17	4	5
18	1	4
19	1	4
19	2	4
20	1	3
20	3	3
21	1	4
22	1	3
23	1	4
23	2	4
24	2	5
25	1	4
25	4	4

	P2	P3
26	1	4
26	4	4
27	1	4
27	4	4
28	1	4
28	2	4
29	1	4
29	2	4
29	3	4
29	4	3
29	5	4
30	1	4
30	4	5
30	5	4
31	1	4
31	2	4
31	4	4
32	1	3
32	2	1
33	1	4
33	2	4
33	3	4
33	5	4
34	4	2
35	2	4
35	4	4
35	5	3
36	6	5

	P2	P3
37	1	4
38	1	5
38	2	5
38	4	5
39	2	4
39	3	4
40	1	4
40	2	4
41	1	5
41	2	4
41	3	5
41	4	5
42	1	5
42	2	5
43	3	5
44	1	4
44	2	5
44	4	4
45	1	5
45	2	5
45	4	5
46	1	4
46	2	3
46	4	5
47	1	5
47	2	5
47	3	5
47	4	5
48	3	5

Variables de satisfaction (P4 items 1-7, P5 items 1-6)

	P4_1	P4_2	P4_3	P4_4	P4_5	P4_6	P4_7	P5_1	P5_2	P5_3	P5_4	P5_5	P5_6
1	4	4	5	5	4	4	5	4	4	4	4	4	4
2	4	5	4	3	4	3	4	NA	5	5	3	4	4
3	4	5	5	NA	4	3	NA	4	3	5	3	3	4
4	4	4	4	4	4	3	4	4	4	5	4	5	5
5	5	4	5	5	5	4	5	5	5	4	4	5	5
6	4	4	5	5	3	4	4	4	4	5	5	3	5
7	4	3	3	4	4	4	3	4	4	3	4	4	4
8	5	5	3	5	5	4	4	5	4	5	5	5	5
9	4	5	NA	5	4	3	3	5	5	5	5	5	5
10	5	5	NA	NA	NA	5	5	NA	NA	5	5	NA	5
11	5	4	4	5	4	4	5	5	5	5	4	3	4
12	5	5	NA	5	4	NA	5	5	5	5	5	5	5
13	3	4	4	5	3	3	5	4	4	5	4	2	5
14	4	4	4	5	5	3	4	4	4	5	3	1	4
15	5	5	5	5	5	4	4	5	4	5	5	5	5
16	4	4	4	4	4	4	4	4	4	4	4	4	4
17	5	5	5	5	3	3	5	4	4	5	4	3	5
18	4	5	3	5	3	4	4	5	4	3	3	2	5
19	4	3	4	4	4	4	3	5	4	5	4	4	4
20	3	4	4	4	4	4	3	3	3	5	4	4	4
21	4	5	5	4	3	3	4	5	5	4	5	2	4
22	NA	NA	NA	NA	NA	4	NA	NA	NA	NA	NA	NA	4
23	4	4	3	3	4	2	4	4	4	4	3	4	4
24	3	4	2	2	4	3	2	3	2	5	2	3	4
25	4	4	5	4	4	4	4	5	5	4	5	4	4
26	4	4	4	4	4	4	4	3	4	4	4	4	4
27	3	4	4	4	3	4	4	5	5	3	4	5	4
28	4	4	4	4	4	4	4	4	4	5	4	3	4
29	4	4	4	4	4	4	4	5	5	5	4	5	5
30	4	4	4	5	5	5	4	4	4	5	5	2	3
31	4	5	4	5	4	5	4	4	3	4	4	5	5
32	3	3	3	3	3	3	3	5	3	5	5	4	4
33	4	4	5	5	5	4	4	4	4	4	4	4	4
34	4	4	3	5	5	5	4	4	4	3	4	2	5
35	4	4	4	4	3	3	3	5	5	3	3	3	3
36	5	5	5	5	5	5	5	5	5	5	5	5	5
37	4	4	4	4	4	4	4	4	4	4	4	4	4
38	5	5	5	4	4	5	4	4	5	5	4	4	5
39	4	5	5	5	4	2	NA	5	4	5	5	4	3
40	4	4	4	NA	NA	NA	NA	4	4	5	NA	NA	NA
41	5	4	5	4	3	4	4	5	4	5	4	3	5
42	4	4	4	5	5	1	4	5	5	5	3	5	5
43	5	5	5	5	5	3	3	4	4	5	5	4	5
44	4	4	4	4	4	4	4	4	4	5	4	4	4
45	5	5	5	5	5	4	5	5	5	3	5	5	5
46	5	4	5	5	3	4	4	4	5	5	5	3	4
47	5	5	5	5	4	5	5	5	5	3	2	3	5
48	5	5	5	5	5	5	5	5	3	5	4	3	3

NA: Sin respuesta

Variables de satisfaction (P5 items 7, P6 y P7 items 1-5, P12, P15)

	P5_7	P6_1	P6_2	P6_3	P6_4	P6_5	P7_1	P7_2	P7_3	P7_4	P7_5	P12	P15
1	4	5	5	4	4	4	5	4	4	5	4	5	5
2	4	4	4	4	3	4	5	5	4	3	2	5	5
3	4	4	5	4	4	5	5	5	4	5	NA	5	5
4	4	4	4	4	3	4	3	4	3	4	3	5	4
5	5	5	5	5	5	5	5	4	3	5	4	5	5
6	4	5	5	5	3	4	5	5	3	4	4	5	5
7	3	4	3	3	2	3	4	3	3	5	3	5	5
8	5	5	5	4	5	5	4	3	3	5	5	5	5
9	5	4	4	3	2	4	1	5	1	1	1	5	5
10	5	5	5	5	5	5	NA	5	NA	NA	NA	5	5
11	4	4	3	4	4	4	5	4	3	5	3	5	5
12	5	5	5	4	4	5	5	5	5	5	5	5	5
13	4	5	3	5	4	3	4	4	NA	5	NA	5	5
14	3	4	4	3	4	3	5	5	1	5	4	5	5
15	4	5	4	4	5	4	5	5	NA	NA	NA	5	5
16	4	4	4	4	4	4	4	4	4	4	4	4	4
17	5	4	4	4	3	5	4	5	NA	5	NA	5	5
18	5	5	3	3	1	3	5	4	3	4	3	3	4
19	3	4	4	3	3	3	4	4	NA	NA	NA	4	4
20	4	4	4	4	3	4	5	5	5	5	5	4	4
21	4	5	4	4	4	3	4	3	3	3	3	5	5
22	NA	4	NA	NA	NA	NA	4	NA	NA	NA	NA	5	5
23	4	4	4	3	2	4	4	4	NA	4	NA	5	5
24	3	2	3	2	1	3	NA	5	NA	4	4	4	3
25	5	4	3	4	3	3	4	4	3	4	4	4	4
26	4	4	4	4	3	4	4	4	4	4	4	5	5
27	5	4	3	4	2	3	4	4	4	4	4	4	4
28	4	4	4	4	4	4	4	5	NA	NA	NA	5	5
29	5	4	5	4	5	5	5	5	4	4	3	5	5
30	5	4	4	4	5	4	5	5	5	3	5	5	5
31	5	5	5	3	4	4	5	5	4	2	3	5	5
32	4	4	3	1	3	3	4	4	NA	NA	3	5	5
33	4	4	4	4	4	4	4	4	4	4	4	5	4
34	5	5	3	4	3	4	3	5	5	5	1	4	4
35	4	4	4	4	3	3	4	5	3	5	3	5	4
36	5	5	5	5	5	5	5	5	5	5	5	5	5
37	4	5	5	5	5	5	4	4	4	4	4	5	4
38	4	5	5	5	4	5	5	4	3	4	5	5	5
39	4	4	5	4	3	4	4	3	4	2	4	5	5
40	NA	4	4	4	4	4	4	4	NA	NA	NA	4	4
41	5	4	5	5	4	3	5	5	4	4	NA	5	5
42	5	5	5	5	5	4	5	5	NA	1	4	5	5
43	4	5	4	3	5	4	NA	NA	4	NA	NA	5	5
44	4	4	4	4	4	4	5	5	NA	5	5	5	5
45	5	5	5	3	3	5	5	5	NA	5	5	5	5
46	3	5	4	3	4	4	4	4	NA	4	NA	5	5
47	5	5	5	3	NA	4	3	5	5	3	4	5	5
48	3	5	5	5	5	5	1	1	5	1	1	5	5

NA: Sin respuesta.

P17: Software Estadistico que usa aparte del R:

Encuesta	P17
1	SAS
1	MINITAB
1	SPSS
2	SAS
3	MSTAT
4	SAS
4	Otros
5	SAS
6	SPLUS
6	MINITAB
6	SPSS
7	MSTATC
7	SPSS
8	PAST
9	NTSYS
10	SAS
11	MINITAB
11	SPSS
12	PAST
13	SPSS
14	MINITAB
14	SPSS
15	MINITAB
15	SPSS
16	SPSS
18	MINITAB
18	SPSS
19	SAS
19	MSTATC
20	SAS
20	MSTATC
20	MINITAB
20	SPSS
21	SAS
24	MINITAB
25	SPSS
26	SAS
26	SPSS

Encuesta	P17
27	MINITAB
27	SPSS
28	SAS
28	GENSTAT
28	NTSYS
28	CropStat
29	SAS
29	MINITAB
29	SPSS
29	STATA
30	MINITAB
30	SPSS
31	SAS
31	MINITAB
31	STATBOX
32	SAS
33	SAS
34	MINITAB
35	SPSS
35	MINITAB
38	SAS
38	MINITAB
40	SAS
40	MSTAT
41	GENSTAT
41	SPSS
41	STATA
42	SPSS
43	Otros
44	SAS
44	SPSS
44	STATA
47	GENSTAT
47	MINITAB
47	SPSS
48	SAS
48	GENSTAT

ANEXO 4. OPINIÓN DE EXPERTOS

Relacion de expertos (PhD)

Nombre	Especialidad	Ocupacion	Instituto	Pais
Banerjee, Partha P.	Corn Breeding	Researcher	ICRISAT	India
Ding, Jane	Plant Ecology	Researcher	Chinese Academy of Science the Institute of Botany	China
Duleep Samuel	Virologist	Researcher	Ind Inst of Hort Research, Bangalore	India
Duyme, Florent	Statistics	Researcher	ARVALIS - Institut du Vegetal	Francia
Escriou, Hervé	Genetics	Researcher	ITB – Institut technique de la betterave	Francia
Klein-Gebbinck, Henry	Canola Pathology	Researcher	Government of Canada	Canada
Kozak, Marcin	Experimental Design and Bioinformatics	Professor	Warsaw University of Life Sciences	Polonia
Marcelo Luiz de Laia	Biologist	Researcher	Universidade do Estado de Santa Catarina	Brasil
Salgado Enríquez, Gustavo	Statistics	Researcher	Edumetrics Cía. Ltda.	Ecuador
Streibig, Jens Carl	Weed Science	Professor	University of Copenhagen	Dinamarca
Wyseure, Guido	Earth and Environmental Sciences	Professor	Department Earth and Environmental Sciences, K.U.Leuven	Belgica

Email de los expertos:

Banerjee, Partha P.: parthabanerjee@aol.in

Ding, Jane: janeding1@hotmail.com

Duleep Samuel: dksamuel@gmail.com

Duyme, Florent: F.DUYME@arvalisinstitutduvegetal.fr

Escriou, Hervé: herve.escriou@itbfr.org

Klein-Gebbinck, Henry: Henry.Klein-Gebbinck@AGR.GC.CA

Kozak, Marcin: nyggus@gmail.com

Marcelo Luiz de Laia: marcelolaia@gmail.com

Salgado Enríquez, Gustavo: eduardosalgado1@hotmail.com

Streibig, Jens Carl: jcs@life.ku.dk

Wyseure, Guido: Guido.Wyseure@ees.kuleuven.be

Según la encuesta (preguntas P8, P9, P10, P13, P14, P16) la opinión de 8 expertos fue:

Q8 - What do you like about the Agricolae Library?

- At least one package for agricultural concepts !, Agricultural designs, Genotype * environment treatments, Ease of use.
- I do not need to invent the wheel once more. Important for the upgrading of crop research in developing countries with no access to expensive programmer.
- Together with Open Office and R with your add on package, then statistical analysis of field experiment can now be done properly.
- The focus on agricultural experimentation. This is extremely important for me because I work with agricultural statistics. This is the only free of charge piece of software that does that, so it's a great thing for me.
- The parametric or non-parametric multiple comparison is very convenient to use. It's time saving when doing data analysis, Give good result. Support through response is excellent. I am grateful to him.

Q9 - What do you dislike about the Agricolae Library?

- The rounding of decimal places. Estimated parameters should be given in sufficient number of decimal places especially if these values are still to be used in other computations.
- You cannot save results in an object.
- You have a design.alpha which is all right but you have as far as I can see no examples of how to analyses it properly. The idea of presenting the designs give by for example by Cochran and Cox is good but you must also show how to analyze them.
- Design of some experiments is implemented but no analysis.
- The poor description of data sets. Functions might be better explained as well user without deep knowledge on the corresponding issues may have problems with understanding the functions. The description of syntax.
- As per my use it's nice. Please consider upon diallel analysis. It's as important as LXT. For a breeder Diallel is very important.

Q10 - Which functions would you like us to incorporate to Agricolae in the future?

- Analysis of mixed models...eg spatial analysis, repeated measures, etc. with BLUPS and mean comparison and fitting different variance-covariance structures.
- Use of mixed models to analyze alpha-designs. Power
- Difficulties in analyzing data coming from unbalanced designs (multi-environment trials for comparing several varieties; not all varieties present in all sites)
- Adjusted means
- Spatial analysis
- Traditional newman-keuls test !.
- Mixed models particularly for series of experiments and split-plots. You could for example use some from the SASmixed package.
- Will be more effective if diallel experiment is included. A separate section should help molecular breeder if, strong genetic mapping and QTL analysis (IM, CIM and MIM) are included.

Q13 - Why wouldn't you choose Agricolae in the future?

- All that I need is not in the package. Some analyses not included.

Q14 - Why would you choose Agricolae in the future?

- It is easy to use and we do not have to develop similar programs.
- Many functions in the same package.
- Improving with each version. Ease of use. Agricultural.
- The focus on agricultural experimentation. This is extremely important for me because I work with agricultural statistics. This is the only free of charge piece of software that does that, so it's a great thing for me.
- It's time saving when carry out a multicomp.
- To do my analysis. But I have lot of expectation as mentioned earlier.

Q16 - Why wouldn't you recommend the use of Agricolae?

- Not everybody is familiar with R. The principles of good crop science statistics can easily be taught by means of your package

ANEXO 5. PROGRAMA PARA ACTUALIZAR AGRICOLAE

```
library(agricolae)
setwd("H:/Agricolae/R")
data(Chz2006)
data(CIC)
data(ComasOxapampa)
data(Glycoalkaloids)
data(LxT)
data(RioChillon)
data(DC)
data(clay)
data(corn)
data(cotton)
data(disease)
data(frijol)
data(genxenv)
data(grass)
data(growth)
data(greenhouse)
data(haynes)
data(Hco2006)
data(heterosis)
data(huasahuasi)
data(plrv)
data(markers)
data(melon)
data(natives)
data(pamCIP)
data(paracsho)
data(plots)
data(potato)
data(ralstonia)
data(rice)
data(sinRepAmmi)
data(soil)
data(sweetpotato)
data(yacon)
data(wilt)
source("AMMI.contour.R")
source("AMMI.R")
source("audpc.R")
source("bar.err.R")
source("bar.group.R")
source("BIB.test.R")
source("carolina.R")
source("consensus.R")
source("correl.R")
source("correlation.R")
source("cv.model.R")
source("cv.similarity.R")
source("delete.na.R")
source("design.ab.R")
source("design.alpha.R")
source("design.bib.R")
source("design.crd.R")
source("design.cyclic.R")
```

```
source("design.graeco.R")
source("design.lattice.R")
source("design.lsd.R")
source("design.rcbd.R")
source("durbin.test.R")
source("fact.nk.R")
source("friedman.R")
source("graph.freq.R")
source("summary.graph.freq.R")
source("plot.graph.freq.R")
source("grid3p.R")
source("hcut.R")
source("hgroups.R")
source("HSD.test.R")
source("index.bio.R")
source("index.smith.R")
source("intervals.freq.R")
source("join.freq.R")
source("kendall.R")
source("kruskal.R")
source("kurtosis.R")
source("lastC.R")
source("lineXtester.R")
source("LSD.test.R")
source("montecarlo.R")
source("nonadditivity.R")
source("normal.freq.R")
source("ojiva.freq.R")
source("order.group.R")
source("order.stat.R")
source("path.analysis.R")
source("PBIB.test.R")
source("polygon.freq.R")
source("random.ab.R")
source("reg.homog.R")
source("resampling.cv.R")
source("resampling.model.R")
source("similarity.R")
source("simulation.model.R")
source("skewness.R")
source("stability.nonpar.R")
source("stability.par.R")
source("stat.freq.R")
source("strip.plot.R")
source("sturges.freq.R")
source("table.freq.R")
source("tapply.stat.R")
source("vark.R")
source("waerden.test.R")
source("waller.R")
source("waller.test.R")
source("wxyz.R")

package.skeleton(name="agricolae",list=ls(), names=TRUE, path="d:/",
force=T)
```

ANEXO 6. FUNCIONES DE AGRICOLAE Y SU DESCRIPCIÓN

Función	Descripción
AMMI.contour	Grafico adicional a AMMI
AMMI	Análisis AMMI, biplot y triplot
audpc	Halla el audpc absoluto y relativo
bar.err	Grafico de barras con su error estándar para comparación de tratamientos
bar.group	Grafico de barras con grupos para comparación de tratamientos
BIB.test	Análisis del diseño bloques incompletos balanceado
carolina	Análisis de los diseños Carolina
consensus	Consenso del dendrograma
correl	Coeficiente de correlación
correlation	Análisis de correlación.
cv.model	Coeficiente de variación relativas de datos experimentales.
cv.similarity	Coeficiente de variabilidad de matriz de disimilaridad
delete.na	Elimina una fila o columna de mayor numero de datos faltantes
design.ab	Diseño factorial pxq en bloques
design.alpha	Diseño alfa
design.bib	Diseño de bloques incompleto balanceado
design.crd	Diseño completamente aleatorio
design.cyclic	Diseño cíclico
design.graeco	Diseño Greco Latino
design.lattice	Diseño Latice
design.lsd	Diseño cuadrado latino.
design.rcbd	Diseño de bloques completos al azar
durbin.test	Prueba de Durbin.
fact.nk	Diseño de n-factores para "k" niveles
friedman	Prueba comparativa de Friedman
graph.freq	Histogramas, absoluta, relativa y densidad
summary.graph.freq	Función genérica de resumen del graph.freq
plot.graph.freq	Función genérica grafica de graph.freq.
grid3p	Interpolación en 3 dimensiones.
hcut	Realiza cortes en el dendrograma de consenso
hgroups	Extrae elementos de grupos de consensus() o hclus() de R.
HSD.test	Prueba de comparación de medias por Tukey
index.bio	Intervalos de confianza para los índices de Bio-diversidad
index.smith	Índice de Smith para uniformidad del suelo experimental.
intervals.freq	Extrae los intervalos de frecuencia de un histograma
join.freq	Junta intervalos de frecuencia

Función	Descripción
kendall	Halla la correlación de Kendall
kruskal	Prueba comparativa de Kruskal-Wallis
kurtosis	Halla la curtosis similar a SAS
lastC	Determina el ultimo caracter en una comparación de tratamientos
lineXtester	Realiza el análisis línea por probador
LSD.test	Prueba de LSD de t-student y ajuste en la probabilidad.
montecarlo	Genera números aleatorios por Montecarlo
nonadditivity	realiza la prueba de no-aditividad del modelo
normal.freq	Superpone la normal a un histograma.
ojiva.freq	Construye la Ojiva de un histograma
order.group	Formar grupos en la comparación de tratamientos
order.stat	Establece el order en la comparación de tratamientos
path.analysis	Análisis de caminos con la matriz de correlación.
PBIB.test	Análisis del diseño bloques incompletos parcealmente balanceado
polygon.freq	Superpone un polígono al grafico de frecuencia
random.ab	Aleatoriza un factorial AxB
reg.homog	Análisis de la homogeneidad de la regresión
resampling.cv	Remuestreo en marcadores moleculares y el calculo del coeficiente de variación
resampling.model	remuestreo para modelos lineales
similarity	Calcula la distancia de similaridad
simulation.model	Simula experimentos bajo la normalidad del error y calcula el p.valor del ANVA.
skewness	Halla el coeficiente de asimétrica igual que SAS
stability.nonpar	Análisis de estabilidad no-parametrico
stability.par	Análisis de estabilidad parametrico
stat.freq	Estadísticas de datos agrupados en una tabla de frecuencia
strip.plot	Análisis del diseño de bloques divididos
sturges.freq	Halla los intervalos por la regla de sturges para un histograma
table.freq	Presenta la tabla de frecuencia de un histograma
tapply.stat	Halla estadísticas de datos agrupados
vark	Halla la variancia de Kendall
waerden.test	Realiza la prueba de Warden, no parametrica
waller	Halla los valores tabulares de Waller-Duncan
waller.test	Realiza la comparacion de tratamientos mediante Waller-Duncan
wxyz	Completa datos faltantes de una matriz de coordenadas mediante un modelo.

ANEXO 7. DATOS DE LA LIBRERÍA AGRICOLAE

Nombre	Descripcion	Fuente	Obs.	Variables
Chz2006	Rendimiento y evaluacion de la marchitez bacteriana en Carhuas.	CIP	1920, 48	5, 5
CIC	Un estudio de la enfermedad en las plantas de papa en las localidades de Comas y Oxapampa en el Perú.	CIP	732	7
ComasOxapampa	Medición de AUDPC en los campos de Comas y Oxapampa para Tizon	CIP	168	4
Glycoalkaloids	Una medición de la Glycoalkaloids por dos métodos: HPLC y espectrofotómetro.	CIP	25	2
LxT	Rendimiento de papa para las cruzas línea x probador	CIP	94	4
RioChillon	Estudio parcial en oca, medida de 27 de marcadores moleculares	CIP	23	27
DC	Datos para análisis de los diseños Carolna I, II y II para mejoramiento genetico.	(Sing, 1979), CIP	72, 300, 64	9, 6, 5
clay	Poblacion de Ralstonia en el suelo	CIP	69	3
corn	Rendimiento de maiz por 4 metodos.Uso con Kruskal-Wallis	(Connover , 1999)	34	3
cotton	Algodón de dos localidades de Lima y Pisco	(Calzada, 1960)	96	5
disease	Rendimiento de papa, aplicación de varios controles de la enfermendad en tres tiempos	CIP	21	7
frijol	Rendimiento de frijol con aplicación de 4 tecnologías de fertilizacion	ICA-Colombia	84	3
genxenv	Rendimiento de 50 genotipos de papa en 5 ambientes	CIP	250	3
grass	Doce viviendas se seleccionan al azar para participar en un experimento con una graminia en vivero	(Connover , 1999)	48	3
growth	Crecimiento de árboles de pijuayo en varias localidades	ICRAF	30	3

Nombre	Descripcion	Fuente	Obs.	Variables
greenhouse	Producción de semilla de papa por hidroponia	CIP	480, 48, 480	5, 5, 5
haynes	Rendimiento de papa para el análisis de la estabilidad no parametrica.	(Haynes,1 998)	16	9
Hco2006	Rendimiento y evaluacion de la marchitez bacteriana en Huanuco	CIP	1347, 27	5, 5
heterosis	Estudio de heterosis en cruzas de papa	CIP	216	11
huasahuasi	Rendimiento de papa en Huasahuasi.	CIP	45	9
plrv	Rendimiento de clones de papa de la población LTVR	CIP	504	6
markers	Estudio parcial en oca, medida de 27 de marcadores moleculares	CIP	23	27
melon	Rendimiento de melón en un cuadrado latino experi	U. Agraria	16	4
natives	Una evaluación en papa nativa, número, peso y tamaño de 24 variedades.	CIP	876	4
pamCIP	Marcadores moleculares en papa silvestre	CIP	43	107
paracsho	Biodiversidad de insectos en la localidad de Paracsho	CIP	110	6
plots	Rendimientos de papa en parcelas divididas	CIP	18	5
potato	Rendimiento de dos variedades de papa realizado en la estación experimental.	CIP	18	4
ralstonia	Marchitez bacteriana en papa, medida de AUDPC	CIP	13	6
rice	Rendimiento de arroz variedad IR8	(Kwanchai , 1984)	36	18
sinRepAmmi	Rendimientos de 50 genotipos de papa en 5 ambientes.	CIP	250	3
soil	Análisis de suelos para 13 localidades	CIP	13	23
sweetpotato	Rendimiento de camote por efecto del virus	CIP	12	2
yacon	Medida de los azucares en Yacon	CIP	432	19
wilt	Porcentaje de marchitez bacteriana, medida de AUDPC.	CIP	13	5

ANEXO 8. SINTAXIS DE LAS FUNCIONES DE AGRICOLAE

```
AMMI.contour <-
function (model, distance, shape, ...)
{
    G <- subset(model$biplot, model$biplot$type == "GEN")
    x <- G$PC1
    y <- G$PC2
    d <- sqrt(x^2 + y^2)
    r <- distance * max(d)
    x <- seq(-r, r, length = shape)
    A <- cbind(x, y = sqrt(r^2 - x^2))
    B <- cbind(x, y = -sqrt(r^2 - x^2))
    lines(A, type = "l", ...)
    lines(B, type = "l", ...)
    Gin <- d <= r
    Gout <- d > r
    GEN.in <- row.names(G)[Gin]
    GEN.out <- row.names(G)[Gout]
    cat("\nLimit, radio:", r)
    cat("\nGenotype  in:", length(GEN.in))
    cat("\nGenotype out:", length(GEN.out), "\n\n")
    distance <- data.frame(row.names = row.names(G), distance = d)
    return(list(`GENOTYPE IN` = GEN.in, `GENOTYPE OUT` = GEN.out,
        Distance = distance))
}
```

```
AMMI <-
function (ENV, GEN, REP, Y, MSE = 0, number = TRUE, graph = "biplot",
    ...)
{
    name.y <- paste(deparse(substitute(Y)))
    cat("\nANALYSIS AMMI: ", name.y, "\nClass level information\n")
    ENV <- as.factor(ENV)
    GEN <- as.factor(GEN)
    nenv <- length(unique(ENV))
    ngen <- length(unique(GEN))
    cat("\nENV: ", unique(as.character(ENV)))
    cat("\nGEN: ", unique(as.character(GEN)))
    minimo <- min(ngen, nenv)
    if (length(REP) > 1) {
        REP <- as.factor(REP)
        nrep <- length(unique(REP))
        cat("\nREP: ", unique(REP))
        cat("\n\nNumber of observations: ", length(na.omit(Y)),
            "\n\n")
        modelo <- aov(Y ~ ENV + REP %in% ENV + GEN + ENV:GEN)
        cat("model Y:", name.y, " ~ ENV + REP%in%ENV + GEN +
ENV:GEN\n")
        cat("Random effect REP%in%ENV\n\n")
        mm <- anova(modelo)
        nn <- mm[2, ]
        mm[2, ] <- mm[3, ]
        mm[3, ] <- nn
        row.names(mm)[2] <- "REP(ENV)"
        row.names(mm)[3] <- "GEN      "
        mm[1, 4] <- mm[1, 3]/mm[2, 3]
```

```
        mm[1, 5] <- 1 - pf(mm[1, 4], mm[1, 1], mm[2, 1])
        print(mm)
        DFE <- df.residual(modelo)
        MSE <- deviance(modelo)/DFE
        medy <- mean(Y, na.rm = TRUE)
        cat("\nCoeff var", "\tMean", name.y, "\n")
        cat(sqrt(MSE) * 100/medy, "\t", medy, "\n")
    }
    else {
        DFE <- nenv * (ngen - 1) * (REP - 1)
        DFEa <- nenv * (REP - 1)
        nrep <- REP
        modelo <- aov(Y ~ ENV + GEN + ENV:GEN)
        xx <- as.matrix(anova(modelo))
        xx <- rbind(xx[1, ], xx[1, ], xx[2:4, ])
        xx[2, 1] <- DFEa
        xx[2, 2:5] <- NA
        xx[, 2] <- xx[, 2] * nrep
        xx[, 3] <- xx[, 3] * nrep
        xx[5, 1] <- DFE
        xx[5, 3] <- MSE
        xx[5, 2] <- MSE * DFE
        xx[1, 4] <- NA
        xx[3, 4] <- xx[3, 3]/MSE
        xx[4, 4] <- xx[4, 3]/MSE
        xx[1, 5] <- NA
        xx[3, 5] <- 1 - pf(xx[3, 4], xx[3, 1], DFE)
        xx[4, 5] <- 1 - pf(xx[4, 4], xx[4, 1], DFE)
        row.names(xx)[1] <- "ENV     "
        row.names(xx)[2] <- "REP(ENV)"
        cat("\nREP: ", REP)
        cat("\n\nNumber of means: ", length(na.omit(Y)), "\n")
        cat("\nDependent Variable:", name.y, "\n\nAnalysis of
variance\n")
        print(xx, na.print = "")
        medy <- mean(Y, na.rm = TRUE)
        cat("\nCoeff var", "\tMean", name.y, "\n")
        cat(sqrt(MSE) * 100/medy, "\t", medy, "\n")
    }
    raw <- data.frame(ENV, GEN, Y)
    MEDIAS <- tapply(raw[, 3], raw[, c(1, 2)], mean)
    xx <- rownames(MEDIAS)
    yy <- colnames(MEDIAS)
    fila <- length(xx)
    col <- length(yy)
    total <- fila * col
    x <- character(length = total)
    y <- character(length = total)
    z <- numeric(length = total)
    k <- 0
    for (i in 1:fila) {
        for (j in 1:col) {
            k <- k + 1
            x[k] <- xx[i]
            y[k] <- yy[j]
            z[k] <- MEDIAS[i, j]
        }
    }
```

```
MEDIAS <- data.frame(ENV = x, GEN = y, Y = z)
x <- MEDIAS[, 1]
y <- MEDIAS[, 2]
z <- MEDIAS[, 3]
modelo2 <- lm(z ~ x + y)
for (i in 1:length(z)) {
    if (is.na(z[i]))
        z[i] <- predict(modelo2, data.frame(x = MEDIAS[i,
            1], y = MEDIAS[i, 2]))
}
MEDIAS <- data.frame(ENV = x, GEN = y, Y = z)
modelo1 <- lm(Y ~ ENV + GEN, data = MEDIAS)
residual <- modelo1$residuals
MEDIAS <- data.frame(MEDIAS, RESIDUAL = residual)
mlabel <- names(MEDIAS)
names(MEDIAS) <- c(mlabel[1:2], name.y, mlabel[4])
OUTRES <- MEDIAS[order(MEDIAS[, 1], MEDIAS[, 2]), ]
OUTRES2 <- by(OUTRES[, 4], OUTRES[, c(2, 1)], function(x) sum(x))
OUTMED <- by(OUTRES[, 3], OUTRES[, c(2, 1)], function(x) sum(x))
s <- svd(OUTRES2)
U <- s$u
L <- s$d
V <- s$v
L <- L[1:minimo]
SS <- (L^2) * nrep
SUMA <- sum(SS)
percent <- round(((1/SUMA) * SS) * 100, 1)
minimo <- min(ngen, nenv)
DFAMMI <- rep(0, minimo)
acum <- DFAMMI
MSAMMI <- DFAMMI
F.AMMI <- DFAMMI
PROBF <- DFAMMI
acumula <- 0
for (i in 1:minimo) {
    DF <- (ngen - 1) + (nenv - 1) - (2 * i - 1)
    if (DF <= 0)
        break
    DFAMMI[i] <- DF
    acumula <- acumula + percent[i]
    acum[i] <- acum[i] + acumula
    MSAMMI[i] <- SS[i]/DFAMMI[i]
    F.AMMI[i] <- round(MSAMMI[i]/MSE, 2)
    PROBF[i] <- round(1 - pf(F.AMMI[i], DFAMMI[i], DFE),
        4)
}
SS <- round(SS, 6)
MSAMMI <- round(MSAMMI, 6)
SSAMMI <- data.frame(percent, acum, Df = DFAMMI, `Sum Sq` = SS,
    `Mean Sq` = MSAMMI, `F value` = F.AMMI, Pr.F = PROBF)
nssammi <- nrow(SSAMMI)
SSAMMI <- SSAMMI[SSAMMI$Df > 0, ]
nss <- nrow(SSAMMI)
row.names(SSAMMI) <- paste("PC", 1:nss, sep = "")
cat("\nAnalysis\n")
print(SSAMMI)
LL <- sqrt(diag(L))
SCOREG <- U %*% LL
```

```
    SCOREE <- V %*% LL
    SCORES <- rbind(SCOREG, SCOREE)
    colnames(SCORES) <- paste("PC", 1:nssammi, sep = "")
    MSCORES <- SCORES[1:ngen, ]
    NSCORES <- SCORES[(ngen + 1):(ngen + nenv), ]
    MGEN <- data.frame(type = "GEN", Y = apply(OUTMED, 1, mean),
        MSCORES)
    MENV <- data.frame(type = "ENV", Y = apply(OUTMED, 2, mean),
        NSCORES)
    bplot <- rbind(MGEN, MENV)
    bplot <- bplot[, 1:(nss + 2)]
    mlabel <- names(bplot)
    names(bplot) <- c(mlabel[1], name.y, mlabel[c(-1, -2)])
    maxy <- max(bplot[, 4])
    miny <- min(bplot[, 4])
    maxx <- max(bplot[, 3])
    minx <- min(bplot[, 3])
    row.names(bplot) <- c(row.names(MGEN), row.names(MENV))
    cp.name <- rownames(SSAMMI)[1:3]
    cp.per <- SSAMMI[1:3, 1]
    if (graph == "biplot") {
        if (number == TRUE) {
            plot(MGEN[, 3], MGEN[, 4], cex = 0, text(MGEN[, 3],
                MGEN[, 4], labels = as.character(1:nrow(MGEN)),
                col = "blue"), xlab = "PC 1", ylab = "PC 2",
                frame = TRUE, ...)
        }
        if (number == FALSE) {
            plot(MGEN[, 3], MGEN[, 4], cex = 0, text(MGEN[, 3],
                MGEN[, 4], labels = row.names(MGEN), col = "blue"),
                xlab = "PC 1", ylab = "PC 2", frame = TRUE, ...)
        }
        points(MENV[, 3], MENV[, 4], cex = 0, text(MENV[, 3],
            MENV[, 4], labels = row.names(MENV), col = "brown"))
        abline(h = 0, v = 0, lty = 2.5, col = "green", lwd = 2)
        s <- seq(length(MENV[, 3]))
        arrows(0, 0, 0.9 * MENV[, 3][s], 0.9 * MENV[, 4][s],
            col = "brown", lwd = 1.8, length = 0.1, code = 2)
        legend("topleft", NULL, pch = c("1", "2"), cp.per[1:2],
            , title = "PC      %", lty = 0)
    }
    if (graph == "triplot") {
        y <- bplot
        lugar <- y[y[, 1] == "ENV", 3:5]
        clones <- y[y[, 1] == "GEN", 3:5]
        maxcp <- max(y[, 3:5])
        mincp <- min(y[, 3:5])
        rango <- maxcp - mincp
        clones <- (clones - mincp)/rango
        nclon <- nrow(clones)
        lugar <- (lugar - mincp)/rango
        nlugar <- nrow(lugar)
        point1 <- cbind(clones[, 1], clones[, 2], clones[, 3])
        point2 <- cbind(lugar[, 1], lugar[, 2], lugar[, 3])
        point3 <- cbind(c(0.6, 0.6, 0), c(0, 0.6, 0.6), c(0.6,
            0, 0.6))
        suppressWarnings(warning(triplot(point1, cex = 0, grid =
TRUE,
```

```
            label = ""))
        if (number == TRUE)
            suppressWarnings(warning(text(tritrafo(point1),
as.character(1:nclon),
                adj = c(0.5, 0), col = "blue", cex = 0.8)))
        if (number == FALSE)
            suppressWarnings(warning(text(tritrafo(point1),
rownames(clones),
                adj = c(0.5, 0), col = "blue", cex = 0.8)))
        suppressWarnings(warning(text(tritrafo(point2),
rownames(lugar),
            adj = c(0.5, 0), col = "red", cex = 0.8)))
        suppressWarnings(warning(text(tritrafo(point3), cp.name,
            adj = c(0.5, 0), cex = 1)))
        legend("topleft", NULL, pch = c("1", "2", "3"), cp.per,
            , title = "PC     %", lty = 0)
        trilines(centerlines(3), lty = 2.5, col = "green", lwd = 2)
        for (i in 1:nlugar) {
            suppressWarnings(warning(trilines(c(point2[i, 1],
                1/3), c(point2[i, 2], 1/3), c(point2[i, 3], 1/3),
                col = "red", lty = 1)))
        }
    }
    return(list(genXenv = OUTRES2, analysis = SSAMMI, means = MEDIAS,
        biplot = bplot))
}
```

```
audpc <-
function (evaluation, dates, type = "absolute")
{
    n <- length(dates)
    k <- ncol(evaluation)
    if (n != k) {
        cat("Error:\nThe number of dates of evaluation \nmust agree
with the number of evaluations\n")
        return()
    }
    audpc <- 0
    area.total <- 100 * (dates[n] - dates[1])
    for (i in 1:(n - 1)) {
        audpc <- audpc + (evaluation[, i] + evaluation[, i +
            1]) * (dates[i + 1] - dates[i])/2
    }
    if (type == "relative")
        audpc <- audpc/area.total
    if (type == "absolute" | type == "relative") {
        return(audpc)
    }
    else cat("Error: type is 'absolute' or 'relative'\n\n")
}
```

```
bar.err <-
function (x, std = TRUE, horiz = FALSE, ...)
{
    y <- x[, 2]
    names(y) <- x[, 1]
    if (std) {
        nivel0 <- x[, 2] - x[, 5] * sqrt(x[, 4])
```

```
        nivel1 <- x[, 2] + x[, 5] * sqrt(x[, 4])
    }
    else {
        nivel0 <- x[, 2] - x[, 5]
        nivel1 <- x[, 2] + x[, 5]
    }
    n <- length(y)
    indice <- barplot(y, horiz = horiz, ...)
    tope <- max(nivel1)/20
    for (i in 1:n) {
        if (horiz) {
            lines(rbind(c(nivel0[i], indice[i]), c(nivel1[i],
                indice[i])), col = "red")
            text(cex = 1, nivel0[i], indice[i], "[")
            text(cex = 1, nivel1[i], indice[i], "]")
        }
        else {
            lines(rbind(c(indice[i], nivel0[i]), c(indice[i],
                nivel1[i])), col = "red")
            text(cex = 1, indice[i], nivel0[i], "---")
            text(cex = 1, indice[i], nivel1[i], "---")
        }
    }
}
```

```
bar.group <-
function (x, horiz = FALSE, ...)
{
    y <- x[, 2]
    names(y) <- x[, 1]
    nivel <- x[, 3]
    n <- length(y)
    indice <- barplot(y, horiz = horiz, ...)
    tope <- max(y)/10
    for (i in 1:n) {
        if (horiz)
            text(y[i] + tope, indice[i], nivel[i])
        else text(indice[i], y[i] + tope, nivel[i])
    }
}
```

```
BIB.test <-
function (block, trt, y, method = "lsd", alpha = 0.05, group = TRUE)
{
    block.unadj <- as.factor(block)
    trt.adj <- as.factor(trt)
    name.y <- paste(deparse(substitute(y)))
    modelo <- formula(paste(name.y, "~ block.unadj + trt.adj"))
    model <- lm(modelo)
    DFerror <- df.residual(model)
    MSerror <- deviance(model)/DFerror
    k <- unique(table(block.unadj))
    r <- unique(table(trt.adj))
    b <- nlevels(block.unadj)
    ntr <- nlevels(trt.adj)
    lambda <- r * (k - 1)/(ntr - 1)
    datos <- data.frame(block, trt, y)
    tabla <- by(datos[, 3], datos[, 1:2], function(x) mean(x,
```

```
        na.rm = TRUE))
    tabla <- as.data.frame(tabla[, ])
    AA <- !is.na(tabla)
    BB <- tapply(y, block.unadj, sum)
    B <- BB %*% AA
    Y <- tapply(y, trt.adj, sum)
    Q <- Y - as.numeric(B)/k
    SStrt.adj <- sum(Q^2) * k/(lambda * ntr)
    MStrt.adj <- SStrt.adj/(ntr - 1)
    sdtdif <- sqrt(2 * k * MSerror/(lambda * ntr))
    Fvalue <- MStrt.adj/MSerror
    mean.adj <- mean(y) + Q * k/(lambda * ntr)
    StdError.adj <- sqrt(MSerror * (1 + k * r * (ntr - 1)/(lambda *
        ntr))/(r * ntr))
    cat("\nANALYSIS BIB: ", name.y, "\nClass level information\n")
    cat("\nBlock: ", unique(as.character(block)))
    cat("\nTrt  : ", unique(as.character(trt)))
    cat("\n\nNumber of observations: ", length(y), "\n\n")
    print(anova(model))
    cat("\ncoefficient of variation:", round(cv.model(model),
        1), "%\n")
    cat(name.y, "Means:", mean(y, na.rm = TRUE), "\n")
    cat("\nTreatments\n")
    print(data.frame(row.names = 1:ntr, trt = row.names(Y), means =
Y/r,
        mean.adj, StdError.adj))
    parameter <- k/(lambda * ntr)
    if (method == "lsd") {
        Tprob <- qt(1 - alpha/2, DFerror)
        cat("\nLSD test")
        cat("\nStd.diff   :", sdtdif)
        cat("\nAlpha      :", alpha)
        cat("\nLSD        :", Tprob * sdtdif)
    }
    if (method == "tukey") {
        Tprob <- qtukey(1 - alpha, ntr, DFerror)
        cat("\nTukey")
        cat("\nAlpha      :", alpha)
        cat("\nStd.err    :", sdtdif)
        cat("\nHSD        :", Tprob * sdtdif)
        parameter <- parameter/2
    }
    if (method == "waller") {
        K <- 650 - 16000 * alpha + 1e+05 * alpha^2
        Tprob <- waller(K, ntr - 1, DFerror, Fvalue)
        cat("\nWaller-Duncan K-ratio")
        cat("\nThis test minimizes the Bayes risk under additive")
        cat("\nloss and certain other assumptions.\n")
        cat("\nk Ratio    : ", K)
        cat("\nMSD        :", Tprob * sdtdif)
    }
    E <- lambda * ntr/(r * k)
    cat("\n\nParameters BIB")
    cat("\nLambda     :", lambda)
    cat("\ntreatmeans :", ntr)
    cat("\nBlock size :", k)
    cat("\nBlocks     :", b)
    cat("\nReplication:", r, "\n")
```

```
    cat("\nEfficiency factor", E, "\n\n<<< Book >>>\n")
    if (group) {
        cat("\nMeans with the same letter are not significantly
different.")
        cat("\n\nComparison of treatments\n\nGroups, Treatments and
means\n")
        output <- order.group(names(mean.adj), as.numeric(mean.adj),
            rep(1, ntr), MSerror, Tprob, std.err = StdError.adj,
            parameter)
        output[, 4] <- r
    }
    if (!group) {
        comb <- combn(ntr, 2)
        nn <- ncol(comb)
        dif <- rep(0, nn)
        pvalue <- rep(0, nn)
        for (k in 1:nn) {
            i <- comb[1, k]
            j <- comb[2, k]
            dif[k] <- abs(mean.adj[i] - mean.adj[j])
            if (method == "lsd")
                pvalue[k] <- 2 * round(1 - pt(dif[k]/sdtdif,
                  DFerror), 4)
            if (method == "tukey")
                pvalue[k] <- round(1 - ptukey(dif[k] *
sqrt(2)/sdtdif,
                  ntr, DFerror), 4)
        }
        if (method == "waller")
            significant = dif > Tprob * sdtdif
        tr.i <- comb[1, ]
        tr.j <- comb[2, ]
        cat("\nComparison between treatments means\n")
        if (method == "waller")
            print(data.frame(row.names = NULL, tr.i, tr.j, diff =
dif,
                significant))
        else print(data.frame(row.names = NULL, tr.i, tr.j, diff =
dif,
            pvalue))
        output <- data.frame(trt = names(mean.adj), means =
as.numeric(mean.adj),
            M = "", N = r, std.err = StdError.adj)
    }
    return(output)
}
```

```
carolina <-
function (model, data)
{
    if (model == 1) {
        "set" <- as.factor(data[, 1])
        "male" <- as.factor(data[, 2])
        "female" <- as.factor(data[, 3])
        "progenie" <- as.factor(data[, 4])
        "replication" <- as.factor(data[, 5])
        y <- data[, 6]
        name.y <- names(data)[6]
```

```
        model <- lm(y ~ set + replication %in% set + male %in%
            set + female %in% male %in% set + replication %in%
            female %in% male %in% set)
        cat("Response(y): ", name.y, "\n\n")
        print(anova(model))
        cat("\nCV:", cv.model(model), "\tMean:", mean(data[,
            6]), "\n")
        m <- length(levels(model$model$male))
        f <- length(levels(model$model$female))
        s <- length(levels(model$model$set))
        r <- length(levels(model$model$replication))
        n <- length(levels(progenie))
        anva <- as.matrix(anova(model))
        anva <- anva[, 1:3]
        var.m <- (anva["set:male", "Mean Sq"] -
anva["set:male:female",
            "Mean Sq"])/(f * r * n)
        var.f <- (anva["set:male:female", "Mean Sq"] -
anva["set:replication:male:female",
            "Mean Sq"])/(n * r)
        var.A <- 4 * var.m
        var.D <- 4 * var.f - 4 * var.m
        output <- list(model, var.m = var.m, var.f = var.f, var.A =
var.A,
            var.D = var.D)
        return(output)
    }
    if (model == 2) {
        "set" <- as.factor(data[, 1])
        "male" <- as.factor(data[, 2])
        "female" <- as.factor(data[, 3])
        "replication" <- as.factor(data[, 4])
        y <- data[, 5]
        name.y <- names(data)[5]
        model <- lm(y ~ set + replication %in% set + male %in%
            set + female %in% set + male:female %in% set)
        cat("Response(y): ", name.y, "\n\n")
        print(anova(model))
        cat("\nCV:", cv.model(model), "\tMean:", mean(y), "\n")
        m <- length(levels(model$model$male))
        f <- length(levels(model$model$female))
        s <- length(levels(model$model$set))
        r <- length(levels(model$model$replication))
        anva <- as.matrix(anova(model))
        anva <- anva[, 1:3]
        var.m <- (anva["set:male", "Mean Sq"] -
anva["set:male:female",
            "Mean Sq"])/(m * r)
        var.f <- (anva["set:female", "Mean Sq"] -
anva["set:male:female",
            "Mean Sq"])/(f * r)
        var.mf <- (anva["set:male:female", "Mean Sq"] -
anva["Residuals",
            "Mean Sq"])/r
        var.Am <- 4 * var.m
        var.Af <- 4 * var.f
        var.D <- 4 * var.mf
        output <- list(model = model, var.m = var.m, var.f = var.f,
```

```
            var.mf = var.mf, var.Am = var.Am, var.Af = var.Af,
            var.D = var.D)
        return(output)
    }
    if (model == 3) {
        "set" <- as.factor(data[, 1])
        "male" <- as.factor(data[, 2])
        "female" <- as.factor(data[, 3])
        "replication" <- as.factor(data[, 4])
        y <- data[, 5]
        name.y <- names(data)[5]
        model <- lm(y ~ set + replication %in% set + female %in%
            set + male %in% set + female:male %in% set)
        cat("Response(y): ", name.y, "\n\n")
        print(anova(model))
        cat("\nCV:", cv.model(model), "\tMean:", mean(y), "\n")
        m <- length(levels(model$model$male))
        f <- length(levels(model$model$female))
        s <- length(levels(model$model$set))
        r <- length(levels(model$model$replication))
        anva <- as.matrix(anova(model))
        anva <- anva[, 1:3]
        var.mi <- (anva[5, 3] - anva["Residuals", "Mean Sq"])/r
        var.m <- (anva["set:male", "Mean Sq"] - anva["Residuals",
            "Mean Sq"])/(2 * r)
        var.A <- 4 * var.m
        var.D <- 2 * var.mi
        output <- list(model = model, var.mi = var.mi, var.m = var.m,
            var.A = var.A, var.D = var.D)
        return(output)
    }
}
```

```
consensus <-
function (data, distance = c("binary", "euclidean", "maximum",
    "manhattan", "canberra", "minkowski"), method = c("complete",
    "ward", "single", "average", "mcquitty", "median", "centroid"),
    nboot = 500, duplicate = TRUE, cex.text = 1, col.text = "red",
    ...)
{
    t0 <- Sys.time()
    distance <- match.arg(distance)
    method <- match.arg(method)
    distancia <- dist(data, method = distance)
    nc <- ncol(data)
    nr <- nrow(data)
    dend <- hclust(distancia, method = method)
    h1 <- cutree(dend, h = 0)
    h2 <- data.frame(h1)
    h3 <- unique(h2)
    dup <- duplicate
    duplicate <- NULL
    if (dup) {
        if (nrow(h3) < length(h1)) {
            nr <- nrow(h3)
            data <- data.frame(d = rownames(data), data)
            h3 <- data.frame(d = rownames(h3), h3)
            duplicate <- merge(h3, data, by = "d", all = TRUE)
```

```
            duplicate <- duplicate[is.na(duplicate[, 2]), ]
            dup0 <- duplicate[, -2]
            duplicate <- as.character(duplicate$d)
            data <- merge(h3, data, by = "d")
            dup1 <- data[, -2]
            dup0 <- cbind(dup0, unique = "")
            dup0[, 1] <- as.character(dup0[, 1])
            dup1[, 1] <- as.character(dup1[, 1])
            ncdup <- ncol(dup1)
            dup0[, ncdup + 1] <- as.character(dup0[, ncdup +
                1])
            ndup0 <- nrow(dup0)
            ndup1 <- nrow(dup1)
            ncdup <- ncol(dup1)
            for (i in 1:ndup0) {
                for (j in 1:ndup1) {
                  if (sum(dup0[i, 2:ncdup] == dup1[j, -1], na.rm =
TRUE) ==
                    ncdup - 1) {
                    dup0[i, ncdup + 1] <- dup1[j, 1]
                    break
                  }
                }
            }
            if (sum(dup0[, ncdup + 1] == "") > 0) {
                add1 <- dup0[dup0[, ncdup + 1] == "", ]
                add1 <- data.frame(d = add1[, 1], h1 = 0, add1[,
                  2:ncdup])
                data <- rbind(data, add1)
            }
            rownames(data) <- data[, 1]
            data <- data[, c(-1, -2)]
            nc <- ncol(data)
            distancia <- dist(data, method = distance)
            dend <- hclust(distancia, method = method)
            duplicate <- dup0[dup0[, ncdup + 1] != "", ][, c(1,
                ncdup + 1)]
            names(duplicate)[1] <- "duplicate"
        }
    }
    nr <- nrow(data)
    if (!is.null(duplicate)) {
        cat("\nDuplicates:", nrow(duplicate))
        cat("\nNew data  :", nr, "Records\n")
    }
    dinicio <- dend$merge
    d0 <- hgroups(dinicio)
    clases <- data.frame(d = d0, height = dend$height, sq =
1:length(d0))
    b <- nboot
    d <- NULL
    for (k in 1:b) {
        muestra <- sample(1:nc, replace = TRUE)
        boot1 <- data[, muestra]
        distancia <- dist(boot1, method = distance)
        d1 <- hclust(distancia, method = method)$merge
        d1 <- hgroups(d1)
        d <- rbind(d, d1)
```

```
    }
    td <- table(d)
    td <- data.frame(td)
    junto <- merge(clases, td, by = "d")
    junto[, 4] <- junto[, 4] * 100/b
    junto <- merge(clases, junto, by = "d", all = TRUE)
    junto[is.na(junto)] <- 0
    junto <- junto[order(junto[, 3]), ]
    tiempo <- Sys.time() - t0
    unidad <- attributes(tiempo)$units
    cat("\nConsensus hclust\n")
    cat("\nMethod distance:", distance)
    cat("\nMethod cluster :", method)
    cat("\nrows and cols  :", nr, nc)
    cat("\nn-bootstrap     :", nboot)
    cat("\nRun time       :", tiempo, unidad, "\n\n")
    cc <- cbind(dend$merge, height = dend$height, percentage =
round(junto[,
        6], 1))
    co <- dend$order
    n1 <- nrow(cc)
    n2 <- length(co)
    p <- rep(0, n1)
    for (i in 1:n1) {
        if ((cc[i, 1] < 0) & (cc[i, 2] < 0)) {
            for (k in 1:n2) {
                if (co[k] == -cc[i, 1])
                  k1 = k
                if (co[k] == -cc[i, 2])
                  k2 = k
            }
            p[i] <- (k1 + k2)/2
        }
        if ((cc[i, 1]) < 0 & (cc[i, 2] > 0)) {
            for (k in 1:n2) {
                if (co[k] == -cc[i, 1])
                  k1 = k
            }
            p[i] <- (k1 + p[cc[i, 2]])/2
        }
        if ((cc[i, 1] > 0) & (cc[i, 2] < 0)) {
            for (k in 1:n2) {
                if (co[k] == -cc[i, 2])
                  k1 = k
            }
            p[i] <- (k1 + p[cc[i, 1]])/2
        }
        if ((cc[i, 1] > 0) & (cc[i, 2] > 0)) {
            p[i] <- (p[cc[i, 1]] + p[cc[i, 2]])/2
        }
    }
    table.dend <- data.frame(dend$merge, xaxis = p, cc[, 3:4],
        groups = d0)
    plot(dend, ...)
    text(p, cc[, 3], ceiling(cc[, 4]), cex = cex.text, col =
col.text)
    return(list(table.dend = table.dend, dendrogram = dend,
duplicates = duplicate))
```

```
}
```

```
correl <-
function (x, y, method = "pearson", alternative = "two.sided")
{
    n <- length(x)
    if (method == "kendall") {
        corr <- kendall(x, y)
        stat <- corr$stat
        rho <- corr$tau
        if (alternative == "two.sided")
            pvalue <- corr$pvalue
        if (alternative == "less")
            pvalue <- 1 - corr$pvalue/2
        if (alternative == "greater")
            pvalue <- corr$pvalue/2
    }
    if (method == "spearman") {
        a <- rank(x)
        b <- rank(y)
        x <- a
        y <- b
    }
    if ((method == "pearson") | (method == "spearman")) {
        sumx <- sum(x^2) - sum(x)^2/n
        sumy <- sum(y^2) - sum(y)^2/n
        sumxy <- sum(x * y) - sum(x) * sum(y)/n
        rho <- sumxy/sqrt(sumx * sumy)
        gl <- n - 2
        stat <- rho * sqrt(gl)/(sqrt(1 - rho^2))
        if (alternative == "two.sided")
            pvalue <- 2 * (1 - pt(abs(stat), gl))
        if (alternative == "less")
            pvalue <- pt(abs(stat), gl)
        if (alternative == "greater")
            pvalue <- 1 - pt(abs(stat), gl)
    }
    list(stat = stat, rho = rho, pvalue = pvalue)
}
```

```
correlation <-
function (x, y = NULL, method = c("pearson", "kendall", "spearman"),
    alternative = "two.sided")
{
    x1 <- x
    y1 <- y
    method <- match.arg(method)
    if (is.data.frame(y))
        y1 <- as.matrix(y)
    if (is.data.frame(x))
        x1 <- as.matrix(x)
    if (!is.null(y1)) {
        if (!is.matrix(x1) & !is.matrix(y1)) {
            name.xy <- paste(deparse(substitute(x)), "and",
deparse(substitute(y)))
            xx <- cbind(x, y)
            yy <- na.omit(xx)
            nn <- length(yy[, 1])
```

```
            x <- yy[, 1]
            y <- yy[, 2]
            corr <- correl(x, y, method = method, alternative =
alternative)
            stat <- corr$stat
            coef <- corr$rho
            pvalue <- corr$pvalue
            if (method == "pearson") {
                gl <- nn - 2
                cat("\nPearson's product-moment correlation\n\n")
                cat("data:", name.xy, "\n")
                cat("t =", stat, ", df =", gl, ", p-value =",
                  pvalue, "\n")
                cat("alternative hypothesis: true rho is ")
                if (alternative == "two.sided")
                  cat("not equal to 0")
                if (alternative == "less")
                  cat("less than 0")
                if (alternative == "greater")
                  cat("greater than 0")
                cat("\nsample estimates:\ncor\n", coef, "\n")
            }
            if (method == "spearman") {
                cat("\nSpearman's rank correlation rho\n\n")
                cat("data:", name.xy, "\n")
                cat("p-value =", pvalue, "\n")
                cat("alternative hypothesis: true rho is ")
                if (alternative == "two.sided")
                  cat("not equal to 0")
                if (alternative == "less")
                  cat("less than 0")
                if (alternative == "greater")
                  cat("greater than 0")
                cat("\nsample estimates:\nrho\n", coef, "\n")
            }
            if (method == "kendall") {
                cat("\nKendall's rank correlation tau\n\n")
                cat("data:", name.xy, "\n")
                cat("z-norm = ", stat, "p-value =", pvalue, "\n")
                cat("alternative hypothesis: true rho is ")
                if (alternative == "two.sided")
                  cat("not equal to 0")
                if (alternative == "less")
                  cat("less than 0")
                if (alternative == "greater")
                  cat("greater than 0")
                cat("\nsample estimates:\ntau\n", coef, "\n")
            }
        }
        else {
            if (is.data.frame(x) | is.matrix(x)) {
                nvarx <- ncol(x)
                if (is.matrix(x))
                  x <- data.frame(x)
                nombrex <- names(x)
                if (is.matrix(x))
                  nombrex <- colnames(x)
                x <- as.matrix(x)
```

```
            }
            else {
                nvarx <- 1
                nombrex <- deparse(substitute(x))
                x <- as.matrix(x)
            }
            if (is.data.frame(y) | is.matrix(y)) {
                nvary <- ncol(y)
                nombrey <- names(y)
                if (is.matrix(y))
                  nombrey <- colnames(y)
                y <- as.matrix(y)
            }
            else {
                nvary <- 1
                nombrey <- deparse(substitute(y))
                y <- as.matrix(y)
            }
            estimate <- rep(0, nvarx * nvary)
            dim(estimate) <- c(nvarx, nvary)
            dimnames(estimate) <- list(nombrex, nombrey)
            pvalue <- estimate
            nn <- round(estimate, 0)
            for (i in 1:nvarx) {
                for (j in 1:nvary) {
                  xx <- cbind(x[, i], y[, j])
                  yy <- na.omit(xx)
                  nn[i, j] <- length(yy[, 1])
                  xa <- yy[, 1]
                  yb <- yy[, 2]
                  corr <- correl(xa, yb, method = method, alternative
= alternative)
                  estimate[i, j] <- corr$rho
                  pvalue[i, j] <- corr$pvalue
                }
            }
            names(method) = ""
            estimate <- round(estimate, 2)
            pvalue <- round(pvalue, 4)
            n1 <- unique(c(nn))
            if (length(n1) == 1)
                nn <- n1
            cat("\nCorrelation Analysis\n\nMethod     :", method)
            cat("\nAlternative:", alternative, "\n\n")
            lista <- list(correlation = estimate, pvalue = pvalue,
                n.obs = nn)
            return(lista)
        }
    }
    else {
        nvar <- ncol(x)
        estimate <- rep(0, nvar * nvar)
        nombres <- names(x)
        if (is.matrix(x))
            nombres <- colnames(x)
        dim(estimate) <- c(nvar, nvar)
        dimnames(estimate) <- list(nombres, nombres)
        pvalue <- estimate
```

```
        nn <- round(estimate, 0)
        x <- as.matrix(x)
        for (i in 1:nvar) {
            for (j in 1:nvar) {
                xx <- cbind(x[, i], x[, j])
                yy <- na.omit(xx)
                nn[i, j] <- length(yy[, 1])
                x0 <- yy[, 1]
                y0 <- yy[, 2]
                if (i == j) {
                  pvalue[i, j] = 0
                  estimate[i, j] = 1
                }
                else {
                  corr <- correl(x0, y0, method = method, alternative
= alternative)
                  estimate[i, j] <- corr$rho
                  pvalue[i, j] <- corr$pvalue
                }
            }
        }
        names(method) = ""
        estimate <- round(estimate, 2)
        diag(pvalue) <- 1
        cat("\nCorrelation Analysis\n\nMethod      :", method)
        cat("\nAlternative:", alternative, "\n\n")
        n1 <- unique(c(nn))
        if (length(n1) == 1)
            nn <- n1
        lista <- list(correlation = estimate, pvalue = pvalue,
            n.obs = nn)
        return(lista)
    }
}
```

```
cv.model <-
function (x)
{
    suma2 <- sum(x$residual^2)
    gl <- x$df.residual
    promedio <- mean(x$fitted.values)
    return(sqrt(suma2/gl) * 100/promedio)
}
```

```
cv.similarity <-
function (A)
{
    nc <- ncol(A)
    nf <- nrow(A)
    suma <- rep(0, nf * (nf - 1)/2)
    ik <- 0
    for (k1 in 1:(nf - 1)) {
        for (k2 in (k1 + 1):nf) {
            mm <- na.omit(c(A[k1, ] == A[k2, ]))
            npar <- length(mm)
            sii <- sum(mm)
            ik <- ik + 1
            suma[ik] <- sii/npar
```

```
        }
    }
    cv <- sd(suma, na.rm = TRUE) * 100/mean(suma, na.rm = TRUE)
    return(cv)
}
```

```
delete.na <-
function (x, alternative = c("less", "greater"))
{
    if (alternative == "less") {
        if (nrow(x) < ncol(x))
            a <- na.omit(x)
        else {
            a <- t(x)
            b <- na.omit(a)
            a <- t(b)
        }
        b <- cbind(a)
        return(b)
    }
    if (alternative == "greater") {
        if (nrow(x) > ncol(x))
            a <- na.omit(x)
        else {
            a <- t(x)
            b <- na.omit(a)
            a <- t(b)
        }
        b <- cbind(a)
        return(b)
    }
}
```

```
design.ab <-
function (A, B, r, number = 1, seed = 0, kinds = "Super-Duper")
{
    p <- length(A)
    q <- length(B)
    t <- rep(0, r * p * q)
    dim(t) <- c(p * q, r)
    ntr <- p * q
    if (seed != 0)
        set.seed(seed, kinds)
    trt <- random.ab(p, q)
    bloque <- c(rep(1, ntr))
    for (y in 2:r) {
        bloque <- c(bloque, rep(y, ntr))
        trt <- rbind(trt, random.ab(p, q))
    }
    n <- nrow(trt)
    factor1 <- rep(NA, n)
    factor2 <- rep(NA, n)
    for (i in 1:n) {
        factor1[i] <- A[trt[i, 1]]
        factor2[i] <- B[trt[i, 2]]
    }
    plots <- number + 1:(ntr * r) - 1
```

```
    book <- data.frame(plots, block = as.factor(bloque), factor1 =
as.factor(factor1),
        factor2 = as.factor(factor2))
    names(book)[c(3, 4)] <- c(paste(deparse(substitute(A))),
        paste(deparse(substitute(B))))
    return(book)
}
```

```
design.alpha <-
function (trt, k, r, number = 1, seed = 0, kinds = "Super-Duper")
{
    name.trt <- c(paste(deparse(substitute(trt))))
    ntr <- length(trt)
    if (seed != 0)
        set.seed(seed, kinds)
    s <- ntr/k
    if (ntr%%k != 0)
        cat("\nThe size of the block is not appropriate", "\nthe
number of treatments must be multiple of k (size block) \n")
    else {
        serie <- ""
        if (r == 2 & k <= s) {
            alpha <- matrix(0, nrow = k, ncol = r)
            alpha[2, 2] <- 1
            for (i in 3:k) {
                alpha[i, 2] <- alpha[i - 1, 2] + 1
            }
            serie <- "I"
        }
        if (r == 3 & s%%2 != 0 & k <= s) {
            alpha <- matrix(0, nrow = k, ncol = r)
            alpha[2, 2] <- 1
            alpha[2, 3] <- s - 1
            for (i in 3:k) {
                alpha[i, 2] <- alpha[i - 1, 2] + 1
                alpha[i, 3] <- alpha[i - 1, 3] - 1
            }
            serie <- "II"
        }
        if (r == 3 & s%%2 == 0 & k < s) {
            s1 <- s/2
            alpha <- matrix(0, nrow = k, ncol = r)
            alpha[2, 2] <- 1
            alpha[2, 3] <- s1
            for (i in 3:k) {
                alpha[i, 2] <- alpha[i - 1, 2] + 1
                alpha[i, 3] <- alpha[i - 2, 3] + 1
            }
            serie <- "III"
        }
        if (r == 4 & s%%2 != 0 & s%%3 != 0 & k <= s) {
            s2 <- (s + 1)/2
            alpha <- matrix(0, nrow = k, ncol = r)
            alpha[2, 2] <- 1
            alpha[2, 3] <- s - 1
            alpha[2, 4] <- s2
            for (i in 3:k) {
                alpha[i, 2] <- alpha[i - 1, 2] + 1
```

```
            alpha[i, 3] <- alpha[i - 1, 3] - 1
            alpha[i, 4] <- alpha[i - 2, 4] + 1
        }
        serie <- "IV"
    }
    if (serie == "") {
        cat("\nhelp(design.alpha): to see the series of alpha
generators\n")
    }
    else {
        nf <- nrow(alpha)
        nc <- ncol(alpha)
        cc <- rep(alpha[, 1], s)
        for (i in 2:r) {
            cc <- c(cc, rep(alpha[, i], s))
        }
        dim(cc) <- c(nf, s, r)
        for (m in 1:r) cc[, 1, m] <- alpha[, m]
        for (i in 2:s) {
            for (j in 1:nf) {
              for (m in 1:r) {
                cc[j, i, m] <- cc[j, i - 1, m] + 1
                if (cc[j, i, m] >= s)
                  cc[j, i, m] <- 0
              }
            }
        }
        for (j in 1:nf) {
            cc[j, , ] <- cc[j, , ] + (j - 1) * s
        }
        intermediate <- cc
        cat("\nalpha design (0,1) - Serie ", serie, "\n")
        E <- (ntr - 1) * (r - 1)/((ntr - 1) * (r - 1) + r *
            (s - 1))
        cat("\nParameters Alpha design\n==========================")
        cat("\ntreatmeans :", ntr)
        cat("\nBlock size :", k)
        cat("\nBlocks     :", s)
        cat("\nReplication:", r, "\n")
        cat("\nEfficiency factor\n(E )", E, "\n\n<<< Book >>>\n")
        for (m in 1:r) {
            for (j in 1:s) {
              aleatorio <- sample(1:k, k)
              cc[, j, m] <- cc[aleatorio, j, m]
            }
        }
        for (m in 1:r) {
            aleatorio <- sample(1:s, s)
            cc[, , m] <- cc[, aleatorio, m]
        }
        cc <- cc + 1
        block <- gl(s, k)
        md <- as.numeric(cc[, , 1])
        bp <- sample(1:ntr, ntr)
        trt <- trt[bp]
        mtr <- trt[md]
        book <- data.frame(block = as.factor(block), trt =
as.factor(mtr),
```

```
            replication = 1)
        for (i in 2:r) {
            md <- as.numeric(cc[, , i])
            mtr <- trt[md]
            book1 <- data.frame(block = as.factor(block),
              trt = as.factor(mtr), replication = i)
            book <- rbind(book, book1)
        }
        plots <- number + 1:(s * k * r) - 1
        cols <- as.numeric(rep(gl(k, 1), s * r))
        book <- data.frame(plots = plots, cols = cols, book)
        book <- data.frame(row.names = NULL, book)
        book$block <- gl(s * r, k)
        names(book)[4] <- name.trt
        tr <- as.character(book[, 4])
        dim(tr) <- c(k, s, r)
        if (r == 2)
            design <- list(rep1 = t(tr[, , 1]), rep2 = t(tr[,
              , 2]))
        if (r == 3)
            design <- list(rep1 = t(tr[, , 1]), rep2 = t(tr[,
              , 2]), rep3 = t(tr[, , 3]))
        if (r == 4)
            design <- list(rep1 = t(tr[, , 1]), rep2 = t(tr[,
              , 2]), rep3 = t(tr[, , 3]), rep4 = t(tr[, ,
              4]))
        return(list(book = book, alpha = alpha, intermediate =
intermediate,
            design = design))
    }
  }
}
```

```
design.bib <-
function (trt, k, number = 1, seed = 0, kinds = "Super-Duper")
{
    ntr <- length(trt)
    if (seed != 0)
        set.seed(seed, kinds)
    md <- t(combn(1:ntr, k))
    b <- nrow(md)
    bp <- sample(1:b, b)
    md <- md[bp, ]
    for (i in 1:b) {
        bi <- sample(1:k, k)
        md[i, ] <- md[i, bi]
    }
    mtr <- trt[t(md)]
    block <- gl(b, k)
    plots <- number + 1:(b * k) - 1
    book <- data.frame(plots, block = as.factor(block), trt =
as.factor(mtr))
    names(book)[3] <- c(paste(deparse(substitute(trt))))
    r <- as.numeric(table(book[, 3])[1])
    lambda <- r * (k - 1)/(ntr - 1)
    E <- lambda * ntr/(r * k)
    cat("\nParameters BIB\n==========")
    cat("\nLambda     :", lambda)
```

```
    cat("\ntreatmeans :", ntr)
    cat("\nBlock size :", k)
    cat("\nBlocks     :", b)
    cat("\nReplication:", r, "\n")
    cat("\nEfficiency factor", E, "\n\n<<< Book >>>\n")
    return(book)
}
```

```
design.crd <-
function (trt, r, number = 1, seed = 0, kinds = "Super-Duper")
{
    junto <- data.frame(trt, r)
    junto <- junto[order(junto[, 1]), ]
    TR <- as.character(junto[, 1])
    r <- as.numeric(junto[, 2])
    y <- rep(TR[1], r[1])
    tr <- length(TR)
    if (seed != 0)
        set.seed(seed, kinds)
    for (i in 2:tr) y <- c(y, rep(TR[i], r[i]))
    trat <- sample(y, length(y), replace = FALSE)
    plots <- number + 1:length(trat) - 1
    dca <- data.frame(plots, trat)
    dca[, 1] <- as.numeric(dca[, 1])
    xx <- dca[order(dca[, 2], dca[, 1]), ]
    r1 <- seq(1, r[1])
    for (i in 2:length(r)) {
        r1 <- c(r1, seq(1, r[i]))
    }
    yy <- data.frame(xx, r = r1)
    book <- yy[order(yy[, 1]), ]
    names(book)[2] <- c(paste(deparse(substitute(trt))))
    return(book)
}
```

```
design.cyclic <-
function (trt, k, r, number = 1, rowcol = FALSE, seed = 0, kinds =
"Super-Duper")
{
    name.trt <- deparse(substitute(trt))
    ntr <- length(trt)
    if ((k * (r%/%k) != r) | (ntr > 30) | (ntr < 6) | (k > 10) |
        (r > 10) | (k == 1) | (r == 1)) {
        cat("\nsee help(design.cyclic\n")
        return()
    }
    if (seed != 0)
        set.seed(seed, kinds)
    ultimo <- rbind(rep(2, 10), c(3, rep(4, 6), 6, 5, 5), c(4,
        3, 3, 3, 3, 6, 6, 3, 7, 3), c(6, rep(5, 4), 3, 3, 5,
        4, 8), c(5, 6, 6, 6, 6, 5, 5, 4, 6, 6), c(4, 4, 4, 4,
        rep(5, 6)), c(2, 4, 8, 7, 8, 8, 6, 8, 8, 9), c(4, 4,
        5, 6, 4, 4, 7, 5, 7, 6), c(3, 7, 8, 8, 7, 8, 8, 10, 8,
        8), c(3, 7, 8, 9, 3, rep(7, 5)), c(6, 6, rep(8, 6), 10,
        11), c(6, 6, 8, 9, 10, 9, rep(0, 4)), c(rep(0, 6), 8,
        12, 13, 11), c(0, 6, 6, 6, 6, rep(11, 5)), c(0, 0, 6,
        6, rep(10, 5), 11), c(0, 0, 0, 6, rep(10, 4), 12, 12),
        c(rep(0, 4), 9, 9, 9, 11, 11, 11), c(rep(0, 5), 8, 8,
```

```
        8, 13, 13))
ultimo1 <- rbind(rep(2, 15), c(5, 5, 6, 6, 9, 9, 6, 6, 11,
    11, 11, 12, 9, 7, 6), c(8, 8, 4, 9, 6, 4, 9, 9, 7, 4,
    5, 8, 6, 10, 9), c(3, 3, 8, 3, 3, 7, 11, 12, 4, 9, 8,
    4, 12, 14, 13), c(7, 6, 10, 5, 10, 5, 4, 3, 9, 7, 3,
    6, 3, 3, 15), c(5, 5, 5, 9, 7, rep(6, 9), 7), c(8, 9,
    8, 6, 10, rep(11, 5), 12, 11, 12, 13, 15), c(9, 8, 9,
    9, rep(0, 11)), c(rep(0, 4), 9, 10, 10, 10, 10, 13, 11,
    15, 14, 12, 11), c(8, 8, 8, 9, 9, 10, 10, 18, 11, rep(0,
    6)), c(rep(0, 9), rep(10, 6)), c(9, 11, 11, 12, 12, 13,
    13, 14, 14, rep(0, 6)), c(rep(0, 9), 17, 14, 17, 14,
    17, 20), c(4, 4, rep(12, 10), 18, 21, 19), c(10, 10,
    rep(0, 5), 14, 15, rep(0, 4), 16, 0), c(0, 0, rep(17,
    5), 0, 0, 20, 22, 13, 27, 0, 20), c(0, 16, 17, 0, 0,
    12, 12, 12, 15, 16, 0, 0, 0, 18, 20), c(12, 0, 0, 14,
    14, rep(0, 5), 14, 14, 14, 0, 0), c(6, 6, 6, 7, rep(0,
    11)), c(rep(0, 4), 17, 17, 0, 0, 0, 17, 0, 0, 17, 0,
    0), c(rep(0, 6), 18, 15, 21, 0, 22, 23, 0, 25, 24), c(rep(16,
    9), rep(0, 6)), c(rep(0, 9), rep(22, 6)), c(13, rep(0,
    5), 13, 20, 23, 20, 19, 18, 20, 20, 21), c(0, 16, 17,
    18, 19, 20, rep(0, 9)), c(15, 15, 15, 15, 15, 15, rep(0,
    9)), c(rep(0, 6), 21, 22, rep(12, 4), 21, 28, 12))
repite <- c(2, 4, 6, 8, 10, 3, 6, 9, 4, 8, 5, 10, 6, 7, 8,
    9, 10)
names(repite) <- 1:17
kr <- 2:10
pk <- c(1, 6, 9, 11, 14:19)
pk1 <- c(1, 6, 10, 14, 17, 19, 22, 24, 26)
primeros <- list(1, 1, 1, 1, 1, c(1, 2), c(1, 3), c(1, 2),
    c(1, 2, 4), c(1, 2, 5), c(1, 2, 3, 5), rbind(c(1, 3,
        4, 5), c(1, 3, 4, 7)), c(1, 2, 3, 4, 7), c(1, 2,
        3, 4, 5, 8), c(1, 2, 3, 4, 5, 7, 9), c(1, 2, 3, 4,
        5, 6, 8, 10), c(1, 2, 3, 4, 5, 6, 7, 10, 11))
primeros1 <- list(1, 1, 1, 1, 1, c(1, 2), c(1, 3), rbind(c(1,
    3), c(1, 4)), rbind(c(1, 2, 4), c(1, 2, 7)), rbind(c(1,
    2, 6), c(1, 3, 13)), c(1, 3, 8, 9), rbind(c(1, 3, 6,
    7), c(1, 2, 5, 15)), rbind(c(1, 2, 3, 7, 10), c(1, 3,
    9, 10, 13)), rbind(c(1, 2, 3, 4, 9, 13), c(1, 2, 3, 5,
    9, 12), c(1, 3, 8, 9, 11, 12)), rbind(c(1, 2, 3, 4, 7,
    9, 12), c(1, 2, 3, 5, 9, 12, 17)), rbind(c(1, 2, 3, 4,
    7, 9, 12, 16), c(1, 3, 4, 5, 6, 9, 12, 13)), rbind(c(1,
    2, 4, 5, 8, 9, 10, 11, 13), c(1, 2, 3, 4, 7, 9, 13, 16,
    20)))
B <- cbind(c(rep(2, 5), 3, 3, 3, 4, 4, 5, 5, 6, 7, 8, 9,
    10), c(2, 4, 6, 8, 10, 3, 6, 9, 4, 8, 5, 10, 6, 7, 8,
    9, 10), c(rep(1, 11), 2, rep(1, 5)), c(rep(1, 7), 2,
    2, 2, 1, rep(2, 6)))
nj <- r/k
r0 <- 0
inicial <- NULL
for (i in 1:nj) {
    r0 <- r0 + k
    if (ntr < 16) {
        yp <- which(c(B[, 1] == k & B[, 2] == r0))
        xp <- primeros[[yp]]
        i1 <- sum(B[1:yp, 3])
        if (!is.matrix(xp))
            final <- ultimo[i1, ntr - 5]
```

```
        if (is.matrix(xp)) {
            i0 <- sum(B[1:(yp - 1), 3]) + 1
            m <- ultimo[i0:i1, ntr - 5]
            r1 <- which(m != 0)
            final <- m[r1]
            xp <- xp[r1, ]
        }
    }
    if ((ntr >= 16) & (ntr <= 30)) {
        yp <- which(c(B[, 1] == k & B[, 2] == r0))
        xp <- primeros1[[yp]]
        i1 <- sum(B[1:yp, 4])
        if (!is.matrix(xp))
            final <- ultimo1[i1, ntr - 15]
        if (is.matrix(xp)) {
            i0 <- sum(B[1:(yp - 1), 4]) + 1
            m <- ultimo1[i0:i1, ntr - 15]
            r1 <- which(m != 0)
            final <- m[r1]
            xp <- xp[r1, ]
        }
    }
    inicial <- rbind(inicial, c(xp, final))
}
design <- NULL
BOOK <- NULL
for (irep in 1:nj) {
    if (nj == 1)
        inicio <- inicial
    if (nj > 1)
        inicio <- inicial[irep, ]
    c1 <- rep(0, ntr * k)
    dim(c1) <- c(ntr, k)
    c1[1, ] <- inicio - 1
    for (i in 2:ntr) {
        for (j in 1:k) {
            c1[i, j] <- c1[i - 1, j] + 1
            c1[i, j] <- c1[i, j]%%ntr
        }
    }
    if (!rowcol) {
        for (i in 1:ntr) {
            nr <- sample(1:k, replace = FALSE)
            c1[i, ] <- c1[i, nr]
        }
    }
    nr <- sample(1:ntr, replace = FALSE)
    c1 <- c1[nr, ]
    c1 <- t(c1)
    c1 <- as.numeric(c1) + 1
    trt1 <- trt[c1]
    book <- data.frame(irep, trt1)
    tr <- as.character(book[, 2])
    dim(tr) <- c(k, ntr)
    design <- rbind(design, list(t(tr)))
    BOOK <- rbind(BOOK, book)
}
plots <- number + 1:(nj * ntr * k) - 1
```

```
    block <- gl(nj * ntr, k)
    book <- data.frame(plots = plots, group = BOOK[, 1], block =
block,
        trt = BOOK[, 2])
    names(book)[4] <- name.trt
    cat("\ncyclic design\n")
    cat("Generator block basic:\n")
    if (is.matrix(inicial)) {
        for (i in 1:nj) cat(inicial[i, ], "\n")
    }
    else cat(inicial, "\n")
    cat("\nParameters\n==========")
    cat("\ntreatmeans :", ntr)
    cat("\nBlock size :", k)
    cat("\nReplication:", r, "\n")
    cat("\n")
    return(list(design = design, book = book))
}
```

```
design.graeco <-
function (trt1, trt2, number = 1, seed = 0, kinds = "Super-Duper")
{
    r <- length(trt1)
    if (floor(r/2) * 2 == r) {
        if (r == 6 | r > 13) {
            cat("not implemented design ", r, "x", r, ", see
help(design.graeco)\n")
            return()
        }
    }
    if (seed != 0)
        set.seed(seed, kinds)
    plots <- number + 1:(r^2) - 1
    col <- rep(gl(r, 1), r)
    row <- gl(r, r)
    C1 <- data.frame(plots, row, col)
    if ((r == 4) | (r == 8) | (r == 10) | (r == 12)) {
        if (r == 4) {
            c1 <- c(1, 2, 3, 4, 2, 1, 4, 3, 3, 4, 1, 2, 4, 3,
                2, 1)
            c2 <- c(1, 4, 2, 3, 2, 3, 1, 4, 3, 2, 4, 1, 4, 1,
                3, 2)
        }
        if (r == 8) {
            c1 <- c(1, 5, 2, 3, 7, 4, 8, 6, 2, 8, 1, 7, 3, 6,
                5, 4, 3, 4, 7, 1, 2, 5, 6, 8, 4, 3, 6, 5, 8,
                1, 7, 2, 5, 1, 8, 4, 6, 3, 2, 7, 6, 7, 4, 8,
                5, 2, 3, 1, 7, 6, 3, 2, 1, 8, 4, 5, 8, 2, 5,
                6, 4, 7, 1, 3)
            C2 <- c(1, 2, 3, 4, 5, 6, 7, 8, 2, 1, 7, 6, 8, 4,
                3, 5, 3, 7, 1, 5, 4, 8, 2, 6, 4, 6, 5, 1, 3,
                2, 8, 7, 5, 8, 4, 3, 1, 7, 6, 2, 6, 4, 8, 2,
                7, 1, 5, 3, 7, 3, 2, 8, 6, 5, 1, 4, 8, 5, 6,
                7, 2, 3, 4, 1)
        }
        if (r == 10) {
            c1 <- c(1, 5, 2, 8, 3, 10, 9, 4, 7, 6, 9, 2, 6, 3,
                8, 4, 10, 5, 1, 7, 10, 9, 3, 7, 4, 8, 5, 6, 2,
```

```
            1, 6, 10, 9, 4, 1, 5, 8, 7, 3, 2, 8, 7, 10, 9,
            5, 2, 6, 1, 4, 3, 7, 8, 1, 10, 9, 6, 3, 2, 5,
            4, 4, 1, 8, 2, 10, 9, 7, 3, 6, 5, 2, 3, 4, 5,
            6, 7, 1, 8, 9, 10, 3, 4, 5, 6, 7, 1, 2, 9, 10,
            8, 5, 6, 7, 1, 2, 3, 4, 10, 8, 9)
        c2 <- c(1, 8, 9, 7, 10, 4, 6, 5, 2, 3, 7, 2, 8, 9,
            1, 10, 5, 6, 3, 4, 6, 1, 3, 8, 9, 2, 10, 7, 4,
            5, 10, 7, 2, 4, 8, 9, 3, 1, 5, 6, 4, 10, 1, 3,
            5, 8, 9, 2, 6, 7, 9, 5, 10, 2, 4, 6, 8, 3, 7,
            1, 8, 9, 6, 10, 3, 5, 7, 4, 1, 2, 5, 6, 7, 1,
            2, 3, 4, 8, 9, 10, 2, 3, 4, 5, 6, 7, 1, 10, 8,
            9, 3, 4, 5, 6, 7, 1, 2, 9, 10, 8)
    }
    if (r == 12) {
        c1 <- c(1, 12, 6, 7, 5, 4, 10, 11, 9, 8, 2, 3, 2,
            11, 5, 8, 6, 3, 9, 12, 10, 7, 1, 4, 3, 10, 8,
            5, 7, 2, 12, 9, 11, 6, 4, 1, 4, 9, 7, 6, 8, 1,
            11, 10, 12, 5, 3, 2, 5, 4, 10, 11, 9, 8, 2, 3,
            1, 12, 6, 7, 6, 3, 9, 12, 10, 7, 1, 4, 2, 11,
            5, 8, 7, 2, 12, 9, 11, 6, 4, 1, 3, 10, 8, 5,
            8, 1, 11, 10, 12, 5, 3, 2, 4, 9, 7, 6, 9, 8,
            2, 3, 1, 12, 6, 7, 5, 4, 10, 11, 10, 7, 1, 4,
            2, 11, 5, 8, 6, 3, 9, 12, 11, 6, 4, 1, 3, 10,
            8, 5, 7, 2, 12, 9, 12, 5, 3, 2, 4, 9, 7, 6, 8,
            1, 11, 10)
        c2 <- c(1, 2, 3, 4, 9, 10, 11, 12, 5, 6, 7, 8, 2,
            1, 4, 3, 10, 9, 12, 11, 6, 5, 8, 7, 3, 4, 1,
            2, 11, 12, 9, 10, 7, 8, 5, 6, 4, 3, 2, 1, 12,
            11, 10, 9, 8, 7, 6, 5, 5, 6, 7, 8, 1, 2, 3, 4,
            9, 10, 11, 12, 6, 5, 8, 7, 2, 1, 4, 3, 10, 9,
            12, 11, 7, 8, 5, 6, 3, 4, 1, 2, 11, 1, 9, 10,
            8, 7, 6, 5, 4, 3, 2, 1, 12, 11, 10, 9, 9, 10,
            11, 12, 5, 6, 7, 8, 1, 2, 3, 4, 10, 9, 12, 11,
            6, 5, 8, 7, 2, 1, 4, 3, 11, 12, 9, 10, 7, 8,
            5, 6, 3, 4, 1, 2, 12, 11, 10, 9, 8, 7, 6, 5,
            4, 3, 2, 1)
    }
    t1 <- sample(trt1, r, replace = FALSE)
    t2 <- sample(trt2, r, replace = FALSE)
    t1 <- t1[c1]
    t2 <- t2[c2]
    C1 <- data.frame(C1[, 1:3], t1, t2)
    C1[, 4] <- trt1[C1[, 4]]
    C1[, 5] <- trt2[C1[, 5]]
}
else {
    C2 <- C1
    a <- 1:(r * r)
    dim(a) <- c(r, r)
    for (i in 1:r) {
        for (j in 1:r) {
            k <- i + j - 1
            if (k > r)
              k <- i + j - r - 1
            a[i, j] <- k
        }
    }
    m <- sample(trt1, r)
```

```
        C1 <- data.frame(C1, m[a])
        m <- sample(trt2, r)
        C2 <- data.frame(C2, m[a])
        ntr <- length(trt1)
        C1 <- data.frame(C1, B = 0)
        for (k in 1:r) {
            x <- C1[k, 4]
            i <- 1
            for (j in 1:(r^2)) {
                y <- C2[(k - 1) * r + i, 4]
                if (C1[j, 4] == x) {
                  C1[j, 5] <- y
                  i <- i + 1
                }
            }
        }
        C1[, 5] <- trt2[C1[, 5]]
    }
    C1[, 4] <- as.factor(C1[, 4])
    C1[, 5] <- as.factor(C1[, 5])
    names(C1)[4] <- c(paste(deparse(substitute(trt1))))
    names(C1)[5] <- c(paste(deparse(substitute(trt2))))
    return(C1)
}
```

```
design.lattice <-
function (k, type = "triple", number = 1, seed = 0, kinds = "Super-
Duper")
{
    if (seed != 0)
        set.seed(seed, kinds)
    cat("\nLattice design, ", type, " ", k, "x", k, "\n")
    seed = 1
    c1 <- rep(0, k * k)
    dim(c1) <- c(k, k)
    c2 <- c1
    for (a in 1:k) {
        for (b in 1:k) {
            p <- k * (a - 1) + b
            c1[a, b] <- p
        }
    }
    c2 <- t(c1)
    nt <- k * k
    t <- 1:nt
    s <- sample(t, nt)
    for (a in 1:k) {
        for (b in 1:k) {
            c1[a, b] <- s[c1[a, b]]
        }
    }
    c2 <- t(c1)
    nt <- k * k
    t <- 1:nt
    sqr <- gl(3, k * k)
    nb <- as.numeric(gl(k, k))
    block <- c(nb, nb + k, nb + 2 * k)
    latino <- as.character(design.lsd(1:k)[, 4])
```

```
    Z <- as.numeric(t(c1))
    c3 <- Z[order(latino)]
    dim(c3) <- c(k, k)
    c3 <- t(c3)
    s <- sample(1:k, k, replace = FALSE)
    c1 <- c1[s, ]
    s <- sample(1:k, k, replace = FALSE)
    c2 <- c2[s, ]
    s <- sample(1:k, k, replace = FALSE)
    c3 <- c3[s, ]
    trt <- c(as.numeric(t(c1)), as.numeric(t(c2)), as.numeric(t(c3)))
    plots <- (number - 1) + 1:(3 * k * k)
    plan <- data.frame(plots, sqr = sqr, block = block, trt = trt)
    if (type == "triple")
        return(list(square1 = c1, square2 = c2, square3 = c3,
            plan = plan))
    if (type == "simple")
        return(list(square1 = c1, square2 = c2, plan = subset(plan,
            as.numeric(plan[, 2]) < 3)))
}
```

```
design.lsd <-
function (trt, number = 1, seed = 0, kinds = "Super-Duper")
{
    r <- length(trt)
    if (seed != 0)
        set.seed(seed, kinds)
    a <- 1:(r * r)
    dim(a) <- c(r, r)
    for (i in 1:r) {
        for (j in 1:r) {
            k <- i + j - 1
            if (k > r)
                k <- i + j - r - 1
            a[i, j] <- k
        }
    }
    m <- sample(2:r, r - 1)
    a <- a[c(1, m), ]
    m <- sample(1:r, r)
    a <- a[, m]
    trat <- trt[a]
    columna <- rep(gl(r, 1), r)
    fila <- gl(r, r)
    plots <- number + 1:length(trat) - 1
    book <- data.frame(plots, row = as.factor(fila), col =
as.factor(columna),
        trat = as.factor(trat))
    names(book)[4] <- c(paste(deparse(substitute(trt))))
    return(book)
}
```

```
design.rcbd <-
function (trt, r, number = 1, seed = 0, kinds = "Super-Duper")
{
    ntr <- length(trt)
    if (seed != 0)
        set.seed(seed, kinds)
```

```
    mtr <- sample(trt, ntr, replace = FALSE)
    block <- c(rep(1, ntr))
    for (y in 2:r) {
        block <- c(block, rep(y, ntr))
        mtr <- c(mtr, sample(trt, ntr, replace = FALSE))
    }
    plots <- number + 1:(ntr * r) - 1
    book <- data.frame(plots, block = as.factor(block), trt =
as.factor(mtr))
    names(book)[3] <- c(paste(deparse(substitute(trt))))
    return(book)
}
```

```
durbin.test <-
function (judge, trt, evaluation, alpha = 0.05, group = TRUE,
    main = NULL)
{
    name.y <- paste(deparse(substitute(evaluation)))
    name.t <- paste(deparse(substitute(trt)))
    judge <- as.factor(judge)
    trt <- as.factor(trt)
    k <- unique(table(judge))
    r <- unique(table(trt))
    b <- nlevels(judge)
    ntr <- nlevels(trt)
    lambda <- r * (k - 1)/(ntr - 1)
    x <- data.frame(judge, trt, evaluation)
    z <- by(x, x$judge, function(x) rank(x$evaluation))
    y <- data.frame(c(z))
    m <- dim(y)
    n <- m[1] * m[2]
    rango <- 1:n
    for (i in 1:m[1]) {
        for (j in 1:m[2]) {
            kk = i + m[1] * (j - 1)
            rango[kk] <- y[i, j]
        }
    }
    x <- data.frame(x, rango)
    means <- tapply.stat(x[, 4], x[, 2], stat = "sum")
    names(means)[1:2] <- c(name.t, name.y)
    z <- by(x, x$trt, function(x) sum(x$rango))
    y <- as.vector(c(z))
    name <- as.character(dimnames(z)$"x$trt")
    s <- (y - r * (k + 1)/2)^2
    s1 <- sum(s)
    gl1 <- ntr - 1
    gl2 <- b * k - ntr - b + 1
    s <- 12 * (ntr - 1) * s1/(r * ntr * (k - 1) * (k + 1))
    prob <- 1 - pchisq(s, gl1)
    Tprob <- qt(1 - alpha/2, gl2)
    sdtdif <- sqrt(r * k * (k + 1) * (b * (k - 1) - s)/(6 * gl2))
    LSD <- Tprob * sdtdif
    cat("\nStudy:", main, "\n")
    cat("\nSum of ranks\n")
    print(data.frame(row.names = NULL, means))
    cat("\nDurbin Test")
    cat("\n==========")
```

```
    cat("\nValue      :", s)
    cat("\nDf 1       :", gl1)
    cat("\nP-value    :", prob)
    cat("\nAlpha      :", alpha)
    cat("\nDf 2       :", gl2)
    cat("\nt-Student  :", Tprob)
    cat("\n\nLeast Significant Difference\nbetween the sum of ranks:
",
        LSD, "\n")
    cat("\nParameters BIB")
    cat("\nLambda     :", lambda)
    cat("\ntreatmeans :", ntr)
    cat("\nBlock size :", k)
    cat("\nBlocks     :", b)
    cat("\nReplication:", r, "\n")
    if (group) {
        cat("\nGroups, Treatments and sum of the ranks\n\n")
        y <- as.numeric(y)
        output <- order.stat(name, y, LSD)
    }
    if (!group) {
        comb <- combn(ntr, 2)
        nn <- ncol(comb)
        dif <- rep(0, nn)
        pvalue <- rep(0, nn)
        stat <- rep("ns", nn)
        for (kk in 1:nn) {
            i <- comb[1, kk]
            j <- comb[2, kk]
            dif[kk] <- abs(y[comb[1, kk]] - y[comb[2, kk]])
            pvalue[kk] <- 2 * round(1 - pt(dif[kk]/sdtdif, gl2),
                4)
            if (dif[kk] >= LSD)
                stat[kk] <- "*"
        }
        tr.i <- comb[1, ]
        tr.j <- comb[2, ]
        cat("\nComparison between treatments sum of the ranks\n\n")
        print(data.frame(row.names = NULL, tr.i, tr.j, diff = dif,
            pvalue = pvalue, signf = stat))
        output <- data.frame(means, M = "", N = r)
    }
    return(output)
}
```

```
fact.nk <-
function (level, factors, blocks, seed = 0, kinds = "Super-Duper")
{
    if (seed != 0)
        set.seed(seed, kinds)
    t <- level^factors
    a <- runif(t)
    for (k in 1:factors) {
        j <- level^(factors - k)
        x <- rep(0, j)
        for (ii in 1:(level - 1)) x <- c(x, rep(ii, j))
        l <- k - 1
        if (l > 0) {
```

```
            for (i in 1:l) {
                x <- rep(x, level)
            }
        }
        a <- rbind(a, x)
    }
    nombres <- LETTERS[1:factors]
    dimnames(a) <- list(c("blocks", nombres), 1:t)
    combinado <- t(a)
    factorial <- combinado[order(combinado[, 1]), ]
    factorial[, 1] <- 1
    if (blocks > 1) {
        for (i in 2:blocks) {
            combinado[, 1] <- runif(t)
            combinado <- combinado[order(combinado[, 1]), ]
            combinado[, 1] <- i
            factorial <- rbind(factorial, combinado)
        }
    }
    row.names(factorial) <- seq(1, nrow(factorial))
    plots <- 1:(t * blocks)
    return(data.frame(plots = plots, factorial))
}
```

```
Friedman <-
function (judge, trt, evaluation, alpha = 0.05, group = TRUE,
    main = NULL)
{
    name.x <- paste(deparse(substitute(judge)))
    name.y <- paste(deparse(substitute(evaluation)))
    name.t <- paste(deparse(substitute(trt)))
    datos <- data.frame(judge, trt, evaluation)
    matriz <- by(datos[, 3], datos[, 1:2], function(x) mean(x,
        na.rm = TRUE))
    matriz <- as.data.frame(matriz[, ])
    name <- as.character(colnames(matriz))
    ntr <- length(name)
    m <- dim(matriz)
    v <- array(0, m)
    for (i in 1:m[1]) {
        v[i, ] <- rank(matriz[i, ])
    }
    vv <- as.numeric(v)
    junto <- data.frame(evaluation, trt)
    means <- tapply.stat(junto[, 1], junto[, 2], stat = "mean")
    sds <- tapply.stat(junto[, 1], junto[, 2], stat = "sd")
    nn <- tapply.stat(junto[, 1], junto[, 2], stat = "length")
    nr <- unique(nn[, 2])
    s <- array(0, m[2])
    for (j in 1:m[2]) {
        s[j] <- sum(v[, j])
    }
    means <- data.frame(means, replication = nn[, 2])
    means[, 2] <- s
    names(means)[1:2] <- c(name.t, name.y)
    rs <- array(0, m[2])
    rs <- s - m[1] * (m[2] + 1)/2
    T1 <- 12 * t(rs) %*% rs/(m[1] * m[2] * (m[2] + 1))
```

```
    T2 <- (m[1] - 1) * T1/(m[1] * (m[2] - 1) - T1)
    cat("\nStudy:", main)
    cat("\n\nSum of the ranks\n")
    print(data.frame(row.names = NULL, means))
    cat("\nFriedman's Test")
    cat("\n===============")
    A1 <- 0
    for (i in 1:m[1]) A1 <- A1 + t(v[i, ]) %*% v[i, ]
    DFerror <- (m[1] - 1) * (m[2] - 1)
    Tprob <- qt(1 - alpha/2, DFerror)
    LSD <- as.numeric(Tprob * sqrt(2 * (m[1] * A1 - t(s) %*%
        s)/DFerror))
    C1 <- m[1] * m[2] * (m[2] + 1)^2/4
    T1.aj <- (m[2] - 1) * (t(s) %*% s - m[1] * C1)/(A1 - C1)
    T2.aj <- (m[1] - 1) * T1.aj/(m[1] * (m[2] - 1) - T1.aj)
    p.value <- 1 - pchisq(T1.aj, m[2] - 1)
    p.fried <- 1 - pFriedman(T1.aj, ntr, nr)
    cat("\nChi-squard:", T1.aj)
    cat("\nPvalue    :", p.value)
    cat("\npFriedman :", p.fried)
    cat("\nAlpha     :", alpha)
    cat("\nt-Student :", Tprob)
    if (group) {
        cat("\nLSD       :", LSD)
        cat("\n\nMeans with the same letter are not significantly
different.")
        cat("\nGroup, Treatment and Sum of the ranks\n")
        s <- as.numeric(s)
        output <- order.stat(name, s, LSD)
        names(output)[2] <- "Sum of ranks"
    }
    if (!group) {
        comb <- combn(ntr, 2)
        nn <- ncol(comb)
        dif <- rep(0, nn)
        pvalue <- rep(0, nn)
        LSD <- rep(0, nn)
        stat <- rep("ns", nn)
        for (k in 1:nn) {
            i <- comb[1, k]
            j <- comb[2, k]
            dif[k] <- abs(s[comb[1, k]] - s[comb[2, k]])
            sdtdif <- sqrt(2 * (m[1] * A1 - t(s) %*% s)/DFerror)
            pvalue[k] <- 2 * round(1 - pt(dif[k]/sdtdif, DFerror),
                4)
            LSD[k] <- round(Tprob * sdtdif, 2)
            if (dif[k] >= LSD[k])
                stat[k] <- "*"
        }
        tr.i <- comb[1, ]
        tr.j <- comb[2, ]
        cat("\n\nComparison between treatments\nSum of the
ranks\n\n")
        print(data.frame(row.names = NULL, tr.i, tr.j, diff = dif,
            pvalue = pvalue, signf = stat, LSD = LSD))
        output <- data.frame(trt = means[, 1], means = means[,
            2], M = "", N = means[, 3])
    }
```

```
    return(output)
}
```

```
graph.freq <-
function (x, breaks = "sturges", counts = NULL, frequency = 1,
    plot = TRUE, nclass = NULL, xlab = "", ylab = "", ...)
{
    if (xlab == "")
        xlab = deparse(substitute(x))
    if (is.numeric(x) & is.null(counts)) {
        if (is.null(nclass)) {
            if (length(breaks) == 1) {
                x <- na.omit(x)
                if (breaks == "sturges")
                  breaks <- sturges.freq(x)$breaks
            }
        }
        else {
            amplitud <- max(x) - min(x)
            n <- length(x)
            z <- rep(0, n)
            y <- as.character(x)
            for (i in 1:n) {
                lc <- nchar(y[i])
                nd <- 0
                for (j in 1:lc) {
                  a <- substr(y[i], j, j)
                  if (a != ".")
                    nd = nd + 1
                  else break
                }
                z[i] <- lc - nd - 1
            }
            d <- max(z)
            if (d < 0)
                d = 0
            tic <- round(amplitud/nclass + 0.5 * 10^(-d), d)
            breaks <- seq(min(x), by = tic, length = nclass +
                1)
        }
        k <- length(breaks)
        n <- length(x)
        counts <- rep(0, k - 1)
        for (i in 1:n) {
            for (j in 1:(k - 1)) {
                if ((x[i] >= breaks[j]) && (x[i] < breaks[j +
                  1]))
                  counts[j] <- counts[j] + 1
            }
        }
        k <- length(counts)
        mids <- rep(0, k)
        ancho <- rep(0, k)
        for (i in 1:k) {
            mids[i] <- (breaks[i] + breaks[i + 1])/2
            ancho[i] <- (breaks[i + 1] - breaks[i])
        }
        altura <- round(1.1 * max(counts), 0)
```

```
    }
    else {
        if (is.list(x)) {
            breaks <- x$breaks
            counts <- x$counts
        }
        else breaks <- x
        k <- length(counts)
        mids <- rep(0, k)
        ancho <- rep(0, k)
        for (i in 1:k) {
            mids[i] <- (breaks[i] + breaks[i + 1])/2
            ancho[i] <- (breaks[i + 1] - breaks[i])
        }
    }
    a <- breaks[1] - ancho[1]/2
    b <- breaks[k + 1] + ancho[k]/2
    relative <- round(counts/sum(counts), 4)
    density <- relative/ancho
    histogram <- structure(list(breaks = breaks, counts = counts,
        mids = mids, relative = relative, density = density),
        class = "graph.freq")
    if (plot) {
        x <- c(a, b)
        if (frequency == 1)
            height <- round(1.1 * max(counts), 1)
        if (frequency == 2)
            height <- round(1.1 * max(relative), 4)
        if (frequency == 3)
            height <- round(1.1 * max(density), 4)
        y <- c(0, height)
        suppressWarnings(warning(plot(x, y, type = "n", xlab = xlab,
            ylab = ylab, ...)))
        if (frequency == 1) {
            for (j in 1:k) {
                suppressWarnings(warning(rect(breaks[j], 0, breaks[j
+
                  1], counts[j], ...)))
            }
        }
        if (frequency == 2) {
            for (j in 1:k) {
                suppressWarnings(warning(rect(breaks[j], 0, breaks[j
+
                  1], relative[j], ...)))
            }
        }
        if (frequency == 3) {
            for (j in 1:k) {
                suppressWarnings(warning(rect(breaks[j], 0, breaks[j
+
                  1], density[j], ...)))
            }
        }
    }
    invisible(histogram)
}
```

```
summary.graph.freq <-
function (object, ...)
{
    xx <- object$mids
    yy <- object$counts
    y1 <- sum(yy)
    zz <- object$breaks
    x <- length(xx)
    acum <- 0
    z <- rep(0, 7 * x)
    dim(z) <- c(x, 7)
    for (i in 1:x) {
        z[i, 1] <- zz[i]
        z[i, 2] <- zz[i + 1]
        z[i, 3] <- xx[i]
        z[i, 4] <- yy[i]
        z[i, 5] <- yy[i]/y1
        z[i, 6] <- yy[i] + acum
        acum <- z[i, 6]
        z[i, 7] <- z[i, 6]/y1
    }
    colnames(z) <- c("Inf", "Sup", "MC", "fi", "fri", "Fi", "Fri")
    rownames(z) <- rep(" ", x)
    return(z)
}
```

```
plot.graph.freq <-
function (x, breaks = "sturges", counts = NULL, frequency = 1,
    plot = TRUE, nclass = NULL, xlab = "", ylab = "", ...)
{
    if (xlab == "")
        xlab = deparse(substitute(x))
    if (is.numeric(x) & is.null(counts)) {
        if (is.null(nclass)) {
            if (length(breaks) == 1) {
                x <- na.omit(x)
                if (breaks == "sturges")
                  breaks <- sturges.freq(x)$breaks
            }
        }
        else {
            amplitud <- max(x) - min(x)
            n <- length(x)
            z <- rep(0, n)
            y <- as.character(x)
            for (i in 1:n) {
                lc <- nchar(y[i])
                nd <- 0
                for (j in 1:lc) {
                  a <- substr(y[i], j, j)
                  if (a != ".")
                    nd = nd + 1
                  else break
                }
                z[i] <- lc - nd - 1
            }
            d <- max(z)
            if (d < 0)
```

```
            d = 0
        tic <- round(amplitud/nclass + 0.5 * 10^(-d), d)
        breaks <- seq(min(x), by = tic, length = nclass +
            1)
    }
    k <- length(breaks)
    n <- length(x)
    counts <- rep(0, k - 1)
    for (i in 1:n) {
        for (j in 1:(k - 1)) {
            if ((x[i] >= breaks[j]) && (x[i] < breaks[j +
              1]))
              counts[j] <- counts[j] + 1
        }
    }
    k <- length(counts)
    mids <- rep(0, k)
    ancho <- rep(0, k)
    for (i in 1:k) {
        mids[i] <- (breaks[i] + breaks[i + 1])/2
        ancho[i] <- (breaks[i + 1] - breaks[i])
    }
    altura <- round(1.1 * max(counts), 0)
}
else {
    if (is.list(x)) {
        breaks <- x$breaks
        counts <- x$counts
    }
    else breaks <- x
    k <- length(counts)
    mids <- rep(0, k)
    ancho <- rep(0, k)
    for (i in 1:k) {
        mids[i] <- (breaks[i] + breaks[i + 1])/2
        ancho[i] <- (breaks[i + 1] - breaks[i])
    }
}
a <- breaks[1] - ancho[1]/2
b <- breaks[k + 1] + ancho[k]/2
relative <- round(counts/sum(counts), 4)
density <- relative/ancho
histogram <- list(breaks = breaks, counts = counts, mids = mids,
    relative = relative, density = density)
if (plot) {
    x <- c(a, b)
    if (frequency == 1)
        height <- round(1.1 * max(counts), 1)
    if (frequency == 2)
        height <- round(1.1 * max(relative), 4)
    if (frequency == 3)
        height <- round(1.1 * max(density), 4)
    y <- c(0, height)
    suppressWarnings(warning(plot(x, y, type = "n", xlab = xlab,
        ylab = ylab, ...)))
    if (frequency == 1) {
        for (j in 1:k) {
```

```
                suppressWarnings(warning(rect(breaks[j], 0, breaks[j
+
                  1], counts[j], ...)))
            }
        }
        if (frequency == 2) {
            for (j in 1:k) {
                suppressWarnings(warning(rect(breaks[j], 0, breaks[j
+
                  1], relative[j], ...)))
            }
        }
        if (frequency == 3) {
            for (j in 1:k) {
                suppressWarnings(warning(rect(breaks[j], 0, breaks[j
+
                  1], density[j], ...)))
            }
        }
    }
    invisible(histogram)
}
```

```
grid3p <-
function (x, y, z, m, n)
{
    xmax <- max(x)
    xmin <- min(x)
    ymax <- max(y)
    ymin <- min(y)
    x2 <- seq(xmin, xmax, length = m)
    y2 <- seq(ymin, ymax, length = n)
    z2 <- numeric(length = m * n)
    dim(z2) <- c(m, n)
    for (k in 1:m) {
        for (v in 1:n) {
            z2[k, v] <- interpp(x, y, z, x2[k], y2[v])$z
        }
    }
    dimnames(z2) <- list(x2, y2)
    return(z2)
}
```

```
Hcut <-
function (consensus, h, group, col.text = "blue", cex.text = 1,
    ...)
{
    dend <- consensus$dendrogram
    if (class(dend) == "hclust")
        dend <- as.dendrogram(dend)
    frame <- consensus$table.dend[consensus$table.dend[, 4] <=
        h, ]
    cut2 <- cut(dend, h)$lower
    max.groups <- as.numeric(length(cut2))
    k <- group
    p <- 0
    pclus <- rep(0, max.groups)
    for (i in 1:max.groups) {
```

```
        pclus[i] <- length(labels(cut2[[i]]))
    }
    if (k > 1)
        for (i in 1:(k - 1)) p = p + pclus[i]
    plot(cut2[[k]], ...)
    text(frame[, 3] - p, frame[, 4], labels = frame[, 5], col =
col.text,
        cex = cex.text)
    return(data.frame(numbers = pclus))
}
```

```
hgroups <-
function (hmerge)
{
    cc <- hmerge
    n1 <- nrow(cc)
    clases = paste(-cc[1, 1], -cc[1, 2], sep = "-")
    clases <- as.matrix(clases)
    for (i in 2:n1) {
        if ((cc[i, 1] < 0) & (cc[i, 2] < 0)) {
            clases <- rbind(clases, paste(-cc[i, 1], -cc[i, 2],
                sep = "-"))
        }
        if ((cc[i, 1]) < 0 & (cc[i, 2] > 0)) {
            clave <- cc[i, 2]
            parte <- clases[clave, ]
            clases <- rbind(clases, paste(-cc[i, 1], parte, sep = "-
"))
        }
        if ((cc[i, 1]) > 0 & (cc[i, 2] < 0)) {
            clave <- cc[i, 1]
            parte <- clases[clave, ]
            clases <- rbind(clases, paste(parte, -cc[i, 2], sep = "-
"))
        }
        if ((cc[i, 1]) > 0 & (cc[i, 2] > 0)) {
            clave1 <- cc[i, 1]
            clave2 <- cc[i, 2]
            parte1 <- clases[clave1, ]
            parte2 <- clases[clave2, ]
            clases <- rbind(clases, paste(parte1, parte2, sep = "-"))
        }
    }
    for (i in 1:n1) {
        x <- strsplit(clases[i, 1], spli = "-")[[1]]
        x
        unique(x)
        y <- as.numeric(unique(x))
        y <- sort(y)
        y <- as.character(y)
        ny <- length(y)
        s <- y[1]
        for (j in 2:ny) s <- paste(s, y[j], sep = "-")
        clases[i, 1] <- s
    }
    return(clases)
}
```

```
HSD.test <-
function (y, trt, DFerror, MSerror, alpha = 0.05, group = TRUE,
    main = NULL)
{
    name.y <- paste(deparse(substitute(y)))
    name.t <- paste(deparse(substitute(trt)))
    junto <- subset(data.frame(y, trt), is.na(y) == FALSE)
    means <- tapply.stat(junto[, 1], junto[, 2], stat = "mean")
    sds <- tapply.stat(junto[, 1], junto[, 2], stat = "sd")
    nn <- tapply.stat(junto[, 1], junto[, 2], stat = "length")
    means <- data.frame(means, std.err = sds[, 2]/sqrt(nn[, 2]),
        replication = nn[, 2])
    names(means)[1:2] <- c(name.t, name.y)
    ntr <- nrow(means)
    Tprob <- qtukey(1 - alpha, ntr, DFerror)
    nr <- unique(nn[, 2])
    nfila <- c("Alpha", "Error Degrees of Freedom", "Error Mean
Square",
        "Critical Value of Studentized Range")
    nvalor <- c(alpha, DFerror, MSerror, Tprob)
    cat("\nStudy:", main)
    cat("\n\nHSD Test for", name.y, "\n")
    xtabla <- data.frame(..... = nvalor)
    row.names(xtabla) <- nfila
    print(xtabla)
    cat("\nTreatment Means\n")
    print(data.frame(row.names = NULL, means))
    if (group) {
        if (length(nr) == 1) {
            HSD <- Tprob * sqrt(MSerror/nr)
            cat("\nHonestly Significant Difference", HSD)
        }
        else {
            nr1 <- 1/mean(1/nn[, 2])
            HSD <- Tprob * sqrt(MSerror/nr1)
            cat("\nHonestly Significant Difference", HSD)
            cat("\nHarmonic Mean of Cell Sizes ", nr1)
            cat("\n\nDifferent HSD for each comparison")
        }
        cat("\nMeans with the same letter are not significantly
different.")
        cat("\n\nGroups, Treatments and means\n")
        output <- order.group(means[, 1], means[, 2], means[,
            4], MSerror, Tprob, means[, 3], parameter = 0.5)
    }
    if (!group) {
        comb <- combn(ntr, 2)
        nn <- ncol(comb)
        dif <- rep(0, nn)
        pvalue <- rep(0, nn)
        for (k in 1:nn) {
            i <- comb[1, k]
            j <- comb[2, k]
            dif[k] <- abs(means[i, 2] - means[j, 2])
            sdtdif <- sqrt(MSerror * (1/means[i, 4] + 1/means[j,
                4]))
            pvalue[k] <- round(1 - ptukey(dif[k] * sqrt(2)/sdtdif,
                ntr, DFerror), 4)
```

```
        }
        tr.i <- comb[1, ]
        tr.j <- comb[2, ]
        cat("\nComparison between treatments means\n\n")
        print(data.frame(row.names = NULL, tr.i, tr.j, diff = dif,
            pvalue = pvalue))
        output <- data.frame(trt = means[, 1], means = means[,
            2], M = "", N = means[, 4], std.err = means[, 3])
    }
    return(output)
}
```

```
index.bio <-
function (data, method = c("Margalef", "Simpson.Dom", "Simpson.Div",
    "Berger.Parker", "McIntosh", "Shannon"), level = 95, nboot = 500,
    console = TRUE)
{
    method <- match.arg(method)
    x <- data
    if (length(x) > 1) {
        if (method == "Margalef")
            formula1 <- expression((length(x) - 1)/log(sum(x)))
        if (method == "Simpson.Dom")
            formula1 <- expression(sum((x/sum(x))^2))
        if (method == "Simpson.Div")
            formula1 <- expression(1 - sum((x/sum(x))^2))
        if (method == "Berger.Parker")
            formula1 <- expression(max(x)/sum(x))
        if (method == "McIntosh")
            formula1 <- expression((sum(x) - sqrt(sum(x^2)))/(sum(x)
-
                sqrt(sum(x))))
        if (method == "Shannon")
            formula1 <- expression(-sum((x/sum(x)) *
log((x/sum(x))^2)))
        index <- eval(formula1)
        n = length(x)
        estimador <- rep(0, n)
        for (i in 1:nboot) {
            x <- sample(data, n, replace = TRUE)
            estimador[i] <- eval(formula1)
        }
        lic = as.numeric(quantile(estimador, (1 - 0.01 * level)/2,
            type = 6))
        lsc = as.numeric(quantile(estimador, (1 + 0.01 * level)/2,
            type = 6))
        if (console) {
            cat("\nMethod:", method, "\n")
            cat("\nThe index:", index, "\n\n")
            cat(level, "percent confidence interval:\n", lic,
                ";", lsc, "\n\n")
        }
    }
    if (length(x) > 1)
        return(data.frame(row.names = NULL, method = method,
            index = index, confidence = level, lic = lic, lsc = lsc,
            nboot = nboot))
    else return("Error data < 2")
```

```
}
```

```
index.smith <-
function (data, ...)
{
    A <- as.matrix(data)
    n <- nrow(A)
    m <- ncol(A)
    TC <- sum(A)^2/(n * m)
    media <- mean(A)
    m1 <- 1:m
    n1 <- 1:n
    k0 <- m1[m/m1 == m%/%m1]
    r0 <- n1[n/n1 == n%/%n1]
    lk0 <- length(k0)
    lr0 <- length(r0)
    k0 <- k0[-lk0]
    r0 <- r0[-lr0]
    lk0 <- length(k0)
    lr0 <- length(r0)
    x <- rep(0, lk0 * lr0)
    V <- x
    CV <- x
    Width <- x
    Length <- x
    l <- 0
    for (k in k0) {
        for (p in r0) {
            l <- l + 1
            ss <- 0
            k3 <- 0
            k4 <- 0
            for (i in 1:(n/p)) {
                k3 <- k4 + 1
                k4 <- k3 + p - 1
                k1 <- 0
                k2 <- 0
                for (j in 1:(m/k)) {
                  k1 <- k2 + 1
                  k2 <- k1 + k - 1
                  ss <- ss + (sum(A[k3:k4, k1:k2]))^2/(k * p)
                }
            }
            V[l] <- ss - TC
            Length[l] <- p
            Width[l] <- k
            x[l] <- k * p
            V[l] <- V[l]/(n * m - 1)
            CV[l] <- sqrt(V[l]) * 100/media
        }
    }
    z <- log(x[-1])
    y <- log(V[1]/V[-1])
    model <- lm(y ~ 0 + z)
    b <- as.numeric(coef(model))
    tabla <- cbind(Size = x, Width, Length, plots = n * m/x,
        Vx = V, CV = round(CV, 1))
    uniformity <- tabla[order(tabla[, 1]), ]
```

```
    model <- lm(CV ~ I(log(x)))
    coeff <- coef(model)
    size <- 1:max(x)
    cv <- coeff[1] + coeff[2] * log(size)
    plot(size, cv, ...)
    points(x, CV)
    return(list(model = model, uniformity = uniformity))
}
```

```
intervals.freq <-
function (breaks)
{
    n <- length(breaks) - 1
    y <- rep(0, 2 * n)
    dim(y) <- c(n, 2)
    for (i in 1:n) {
        y[i, 1] <- breaks[i]
        y[i, 2] <- breaks[i + 1]
    }
    colnames(y) <- c("lower", "upper")
    return(y)
}
```

```
join.freq <-
function (breaks, join)
{
    join <- join[-1]
    breaks <- breaks[-join]
    return(breaks)
}
```

```
Kendall <-
function (data1, data2)
{
    n <- length(data1)
    n2 <- 0
    n1 <- 0
    is <- 0
    n0 <- n - 1
    for (j in 1:n0) {
        jj <- j + 1
        for (k in jj:n) {
            a1 <- data1[j] - data1[k]
            a2 <- data2[j] - data2[k]
            aa <- a1 * a2
            if (!is.na(aa)) {
                if (aa) {
                  n1 <- n1 + 1
                  n2 <- n2 + 1
                  if (aa > 0)
                    is <- is + 1
                  else is = is - 1
                }
                else {
                  if (a1)
                    n1 <- n1 + 1
                  if (a2)
                    n2 <- n2 + 1
```

```
                }
            }
        }
    }
    tau <- is/(sqrt(n1) * sqrt(n2))
    z <- is/sqrt(vark(data1, data2))
    prob <- 2 * pnorm(-abs(z))
    return(list(stat = z, tau = tau, pvalue = prob))
}
```

```
Kruskal <-
function (y, trt, alpha = 0.05, group = TRUE, main = NULL)
{
    name.y <- paste(deparse(substitute(y)))
    name.t <- paste(deparse(substitute(trt)))
    junto <- subset(data.frame(y, trt), is.na(y) == FALSE)
    N <- nrow(junto)
    junto[, 1] <- rank(junto[, 1])
    means <- tapply.stat(junto[, 1], junto[, 2], stat = "sum")
    sds <- tapply.stat(junto[, 1], junto[, 2], stat = "sd")
    nn <- tapply.stat(junto[, 1], junto[, 2], stat = "length")
    means <- data.frame(means, replication = nn[, 2])
    names(means)[1:2] <- c(name.t, name.y)
    ntr <- nrow(means)
    DFerror <- N - ntr
    rs <- 0
    U <- 0
    for (i in 1:ntr) {
        rs <- rs + means[i, 2]^2/means[i, 3]
        U <- U + 1/means[i, 3]
    }
    S <- (sum(junto[, 1]^2) - (N * (N + 1)^2)/4)/(N - 1)
    H <- (rs - (N * (N + 1)^2)/4)/S
    cat("\nStudy:", main)
    cat("\nKruskal-Wallis test's\n")
    cat("\nValue:", H)
    cat("\ndegrees of freedom:", ntr - 1)
    p.chisq <- 1 - pchisq(H, ntr - 1)
    p.kw <- 1 - pKruskalWallis(H, ntr, N, U)
    cat("\nPvalue chisq  :", p.chisq)
    cat("\npKruskalWallis:", p.kw, "\n")
    DFerror <- N - ntr
    Tprob <- qt(1 - alpha/2, DFerror)
    MSerror <- S * ((N - 1 - H)/(N - ntr))
    cat("\nMean of the ranks\n")
    means[, 2] <- means[, 2]/means[, 3]
    print(data.frame(row.names = NULL, means))
    nr <- unique(means[, 3])
    if (group) {
        Tprob <- qt(1 - alpha/2, DFerror)
        cat("\nt-Student:", Tprob)
        cat("\nAlpha     :", alpha)
        if (length(nr) == 1) {
            LSD <- Tprob * sqrt(2 * MSerror/nr)
            cat("\nLSD       :", LSD, "\n")
        }
        else {
            nr1 <- 1/mean(1/nn[, 2])
```

```
            LSD1 <- Tprob * sqrt(2 * MSerror/nr1)
            cat("\nLSD       :", LSD1, "\n")
            cat("\nHarmonic Mean of Cell Sizes ", nr1)
        }
        cat("\nMeans with the same letter are not significantly
different\n")
        cat("\nGroups, Treatments and mean of the ranks\n")
        output <- order.group(means[, 1], means[, 2], means[,
            3], MSerror, Tprob, std.err = sqrt(MSerror/means[,
            3]))
    }
    if (!group) {
        comb <- combn(ntr, 2)
        nn <- ncol(comb)
        dif <- rep(0, nn)
        pvalue <- rep(0, nn)
        LSD <- rep(0, nn)
        stat <- rep("ns", nn)
        for (k in 1:nn) {
            i <- comb[1, k]
            j <- comb[2, k]
            dif[k] <- abs(means[i, 2] - means[j, 2])
            sdtdif <- sqrt(S * ((N - 1 - H)/(N - ntr)) * (1/means[i,
                3] + 1/means[j, 3]))
            pvalue[k] <- 2 * round(1 - pt(dif[k]/sdtdif, DFerror),
                4)
            LSD[k] <- round(Tprob * sdtdif, 2)
            if (dif[k] >= LSD[k])
                stat[k] <- "*"
        }
        tr.i <- comb[1, ]
        tr.j <- comb[2, ]
        cat("\nComparison between treatments mean of the ranks\n")
        print(data.frame(row.names = NULL, tr.i, tr.j, diff = dif,
            pvalue = pvalue, signf = stat, LSD = LSD))
        output <- data.frame(trt = means[, 1], means = means[,
            2], M = "", N = means[, 3])
    }
    return(output)
}
```

```
kurtosis <-
function (x)
{
    x <- na.omit(x)
    n <- length(x)
    suma <- sum((x - mean(x))^4)/(var(x))^2
    k <- n * (n + 1) * suma/((n - 1) * (n - 2) * (n - 3)) - 3 *
        (n - 1)^2/((n - 2) * (n - 3))
    return(k)
}
```

```
lastC <-
function (x)
{
    y <- sub(" +$", "", x)
    p1 <- nchar(y)
    cc <- substr(y, p1, p1)
```

```
    return(cc)
}
```

```
lineXtester <-
function (replications, lines, testers, y)
{
    name.y <- deparse(substitute(y))
    cat("\nANALYSIS LINE x TESTER: ", name.y, "\n")
    replications <- as.factor(replications)
    lines <- as.factor(lines)
    testers <- as.factor(testers)
    datos <- data.frame(Replications = replications, Lines = lines,
        Testers = testers, Y = y)
    l <- length(levels(datos[, 2]))
    r <- length(levels(datos[, 1]))
    t <- length(levels(datos[, 3]))
    Treatments <- as.factor(paste(datos[, 2], datos[, 3]))
    modelo1 <- aov(Y ~ Replications + Treatments, data = datos)
    matriz1 <- as.matrix(anova(modelo1))
    modelo4 <- aov(Y ~ Lines * Testers, data = datos)
    matriz4 <- as.matrix(anova(modelo4))
    datos2 <- na.omit(datos)
    mm <- tapply(datos2[, 4], datos2[, 2:3], mean, na.rm = TRUE)
    cmm <- ncol(mm)
    rmm <- nrow(mm)
    SCA <- mm
    for (i in 1:rmm) {
        for (j in 1:cmm) {
            SCA[i, j] <- round(mm[i, j] - mean(mm[, j], na.rm = TRUE)
-
                mean(mm[i, ], na.rm = TRUE) + mean(mm, na.rm = TRUE),
                3)
        }
    }
    mm <- tapply(datos2[, 4], datos2[, 2], mean, na.rm = TRUE)
    GCA.lines <- round(mm - mean(datos2[, 4], na.rm = TRUE),
        3)
    mm <- tapply(datos2[, 4], datos2[, 3], mean, na.rm = TRUE)
    GCA.testers <- round(mm - mean(datos2[, 4], na.rm = TRUE),
        3)
    Crosses <- as.factor(paste(datos2[, 2], datos2[, 3]))
    modelo3 <- aov(Y ~ Crosses, data = datos2)
    matriz3 <- as.matrix(anova(modelo3))
    datos3 <- subset(datos, is.na(datos[, 2]) | is.na(datos[,
        3]))
    Parents <- as.factor(paste(datos3[, 2], datos3[, 3]))
    modelo2 <- aov(Y ~ Parents, data = datos3)
    matriz2 <- as.matrix(anova(modelo2))
    matriz5 <- matriz1[2, ] - matriz2[1, ] - matriz3[1, ]
    matriz <- rbind(matriz1[1:2, ], matriz2[1, ], matriz5, matriz3[1,
        ], matriz4[1:3, ], matriz1[3, ])
    total1 <- sum(matriz1[, 1])
    total2 <- sum(matriz1[, 2])
    matriz5 <- c(total1, total2, NA, NA, NA)
    matriz <- rbind(matriz, matriz5)
    for (i in 1:9) {
        matriz[i, 3] <- matriz[i, 2]/matriz[i, 1]
        matriz[i, 4] <- round(matriz[i, 3]/matriz[9, 3], 3)
```

```
        matriz[i, 5] <- round(1 - pf(matriz[i, 4], matriz[i,
            1], matriz[9, 1]), 4)
        if (i == 6 | i == 7) {
            matriz[i, 4] <- round(matriz[i, 3]/matriz[8, 3],
                3)
            matriz[i, 5] <- round(1 - pf(matriz[i, 4], matriz[i,
                1], matriz[8, 1]), 4)
        }
    }
    matriz[9, 4] <- NA
    matriz[9, 5] <- NA
    rownames(matriz) <- c("Replications", "Treatments", "Parents",
        "Parents vs. Crosses", "Crosses", "Lines", "Testers",
        "Lines X Testers", "Error", "Total")
    cm <- matriz[9, 3]
    s1 <- sqrt(cm/(r * t))
    s2 <- sqrt(cm/(r * l))
    s3 <- sqrt(cm/r)
    s4 <- sqrt(2 * cm/(r * t))
    s5 <- sqrt(2 * cm/(r * l))
    s6 <- sqrt(2 * cm/r)
    cov1 <- (matriz[6, 3] - matriz[8, 3])/(r * t)
    cov2 <- (matriz[7, 3] - matriz[8, 3])/(r * l)
    cov3 <- (((l - 1) * matriz[6, 3] + (t - 1) * matriz[7, 3])/(l +
        t - 2) - matriz[8, 3])/(r * (2 * l * t - l - t))
    cov4 <- ((matriz[6, 3] - matriz[9, 3]) + (matriz[7, 3] -
        matriz[9, 3]) + (matriz[8, 3] - matriz[9, 3]))/(3 * r)
    +(6 * r * cov3 - r * (l + t) * cov3)/(3 * r)
    F <- 0
    var.A0 <- cov3 * (4/(1 + F))^2
    var.D0 <- ((matriz[8, 3] - matriz[9, 3])/r) * (2/(1 + F))^2
    F <- 1
    var.A1 <- cov3 * (4/(1 + F))^2
    var.D1 <- ((matriz[8, 3] - matriz[9, 3])/r) * (2/(1 + F))^2
    c1 <- matriz[6, 2] * 100/matriz[5, 2]
    c2 <- matriz[7, 2] * 100/matriz[5, 2]
    c3 <- matriz[8, 2] * 100/matriz[5, 2]
    cat("\nANOVA with parents and crosses",
"\n============================\n")
    matriz1 <- matriz[c(1, 2, 3, 4, 5, 9, 10), ]
    print(matriz1, na.print = "")
    cat("\nANOVA for line X tester analysis",
"\n================================\n")
    matriz1 <- matriz[6:9, ]
    print(matriz1, na.print = "")
    cat("\nANOVA for line X tester analysis including parents",
        "\n=========================================\n")
    print(matriz, na.print = "")
    cat("\nGCA Effects:", "\n==========")
    cat("\nLines Effects:\n")
    print(GCA.lines)
    cat("\nTesters Effects:\n")
    print(GCA.testers)
    cat("\nSCA Effects:", "\n==========\n")
    print(SCA)
    cat("\nStandard Errors for Combining Ability Effects:",
"\n=====================================")
    cat("\nS.E. (gca for line)   :", s1)
```

```
    cat("\nS.E. (gca for tester) :", s2)
    cat("\nS.E. (sca effect)     :", s3)
    cat("\nS.E. (gi - gj)line    :", s4)
    cat("\nS.E. (gi - gj)tester  :", s5)
    cat("\nS.E. (sij - skl)tester:", s6, "\n")
    cat("\nGenetic Components:", "\n==================")
    cat("\nCov H.S. (line)    :", cov1)
    cat("\nCov H.S. (tester) :", cov2)
    cat("\nCov H.S. (average):", cov3)
    cat("\nCov F.S. (average):", cov4)
    cat("\nF = 0, Aditive genetic variance :", var.A0)
    cat("\nF = 1, Aditive genetic variance :", var.A1)
    cat("\nF = 0, Variance due to Dominance:", var.D0)
    cat("\nF = 1, Variance due to Dominance:", var.D1, "\n")
    cat("\nProportional contribution of lines, testers", "\n and
their interactions to total variance",
        "\n=========================================")
    cat("\nContributions of lines  :", c1)
    cat("\nContributions of testers:", c2)
    cat("\nContributions of lxt    :", c3, "\n")
    return(modelo1)
}
```

```
LSD.test <-
function (y, trt, DFerror, MSerror, alpha = 0.05, p.adj = c("none",
    "holm", "hochberg", "bonferroni", "BH", "BY", "fdr"), group =
TRUE,
    main = NULL)
{
    p.adj <- match.arg(p.adj)
    name.y <- paste(deparse(substitute(y)))
    name.t <- paste(deparse(substitute(trt)))
    junto <- subset(data.frame(y, trt), is.na(y) == FALSE)
    means <- tapply.stat(junto[, 1], junto[, 2], stat = "mean")
    sds <- tapply.stat(junto[, 1], junto[, 2], stat = "sd")
    nn <- tapply.stat(junto[, 1], junto[, 2], stat = "length")
    means <- data.frame(means, std.err = sds[, 2]/sqrt(nn[, 2]),
        replication = nn[, 2])
    names(means)[1:2] <- c(name.t, name.y)
    ntr <- nrow(means)
    nk <- choose(ntr, 2)
    if (p.adj == "none")
        Tprob <- qt(1 - alpha/2, DFerror)
    else {
        a <- 1e-06
        b <- 1
        for (i in 1:100) {
            x <- (b + a)/2
            d <- p.adjust(x, n = nk, p.adj) - alpha
            fa <- p.adjust(a, n = nk, p.adj) - alpha
            if (d * fa < 0)
                b <- x
            if (d * fa > 0)
                a <- x
        }
        Tprob <- qt(1 - x/2, DFerror)
    }
    nr <- unique(nn[, 2])
```

```
    nfila <- c("Alpha", "Error Degrees of Freedom", "Error Mean
Square",
        "Critical Value of t")
    nvalor <- c(alpha, DFerror, MSerror, Tprob)
    cat("\nStudy:", main)
    cat("\n\nLSD t Test for", name.y, "\n")
    if (p.adj != "none")
        cat("P value adjustment method:", p.adj, "\n")
    xtabla <- data.frame(..... = nvalor)
    row.names(xtabla) <- nfila
    print(xtabla)
    cat("\nTreatment Means\n")
    print(data.frame(row.names = NULL, means))
    if (group) {
        if (length(nr) == 1) {
            LSD <- Tprob * sqrt(2 * MSerror/nr)
            cat("\nLeast Significant Difference", LSD)
        }
        else {
            nr1 <- 1/mean(1/nn[, 2])
            LSD1 <- Tprob * sqrt(2 * MSerror/nr1)
            cat("\nLeast Significant Difference", LSD1)
            cat("\nHarmonic Mean of Cell Sizes ", nr1)
        }
        cat("\nMeans with the same letter are not significantly
different.")
        cat("\n\nGroups, Treatments and means\n")
        output <- order.group(means[, 1], means[, 2], means[,
            4], MSerror, Tprob, means[, 3])
    }
    if (!group) {
        comb <- combn(ntr, 2)
        nn <- ncol(comb)
        dif <- rep(0, nn)
        pvalue <- rep(0, nn)
        for (k in 1:nn) {
            i <- comb[1, k]
            j <- comb[2, k]
            dif[k] <- abs(means[i, 2] - means[j, 2])
            sdtdif <- sqrt(MSerror * (1/means[i, 4] + 1/means[j,
                4]))
            pvalue[k] <- 2 * (1 - pt(dif[k]/sdtdif, DFerror))
            if (p.adj != "none")
                pvalue[k] <- p.adjust(pvalue[k], n = nk, p.adj)
            pvalue[k] <- round(pvalue[k], 4)
        }
        tr.i <- comb[1, ]
        tr.j <- comb[2, ]
        cat("\nComparison between treatments means\n\n")
        print(data.frame(row.names = NULL, tr.i, tr.j, diff = dif,
            pvalue = pvalue))
        output <- data.frame(trt = means[, 1], means = means[,
            2], M = "", N = means[, 4], std.err = means[, 3])
    }
    return(output)
}
```

```
Montecarlo <-
```

```
function (data, k, ...)
{
    if (is.list(data)) {
        f <- data$counts
        breaks <- data$breaks
        tot <- sum(f)
        n <- length(data$mids)
        breaks <- c(breaks, 2 * breaks[n + 1] - breaks[n])
        y <- rep(0, n + 1)
        for (i in 1:n) {
            y[i + 1] <- y[i] + f[i]/tot
        }
        y <- c(y, 1)
        ns <- length(breaks)
        sk <- runif(k)
        w <- rep(0, k)
        for (j in 1:k) {
            for (i in 1:(ns - 1)) {
                if ((sk[j] >= y[i]) & (sk[j] <= y[i + 1])) {
                  w[j] <- (breaks[i] * (y[i + 1] - sk[j]) + breaks[i
+
                    1] * (sk[j] - y[i]))/(y[i + 1] - y[i])
                  break
                }
            }
        }
    }
    else {
        s <- density(data, ...)
        x <- s$x
        ancho <- s$bw
        ns <- length(x)
        y <- s$y
        A <- rep(0, (ns - 1))
        for (i in 1:(ns - 1)) A[i] <- (y[i] + y[i + 1]) * ancho/2
        total <- sum(A)
        pa <- rep(0, (ns - 1))
        pa[1] <- A[1]
        for (i in 2:(ns - 1)) pa[i] <- pa[i - 1] + A[i]
        pa <- pa/total
        x <- x[-1]
        sk <- runif(k)
        w <- data.frame(orden = 1:k, sk, aleatorio = 0)
        w <- w[order(w[, 2]), ]
        start <- 1
        for (i in 1:k) {
            for (j in start:ns) {
                if (w[i, 2] < pa[j]) {
                  w[i, 3] <- x[j]
                  start <- j
                  break
                }
            }
        }
        w <- w[order(w[, 1]), 3]
    }
    return(w)
}
```

```
Nonadditivity <-
function (y, factor1, factor2, df, MSerror)
{
    name.y <- paste(deparse(substitute(y)))
    conjunto <- subset(data.frame(y, factor1, factor2), is.na(y) ==
        FALSE)
    zz <- tapply(conjunto[, 1], conjunto[, c(2, 3)], mean)
    nn <- tapply(conjunto[, 1], conjunto[, c(2, 3)], length)
    r <- 1/mean(1/nn)
    sc <- df * MSerror
    x1 <- rownames(zz)
    y1 <- colnames(zz)
    fila <- length(x1)
    col <- length(y1)
    total <- fila * col
    x <- character(length = total)
    y <- character(length = total)
    z <- numeric(length = total)
    k <- 0
    for (i in 1:fila) {
        for (j in 1:col) {
            k <- k + 1
            x[k] <- x1[i]
            y[k] <- y1[j]
            z[k] <- zz[i, j]
        }
    }
    yy <- data.frame(x, y, z)
    yy[, 1] <- as.factor(yy[, 1])
    yy[, 2] <- as.factor(yy[, 2])
    modelo.y <- lm(yy[, 3] ~ yy[, 1] + yy[, 2])
    y.est <- predict(modelo.y)
    residual <- residuals(modelo.y)
    q <- y.est^2
    modelo.q <- lm(q ~ yy[, 1] + yy[, 2])
    Nonadditivity <- residuals(modelo.q)
    modelo.e <- lm(residual ~ Nonadditivity)
    anva <- anova(modelo.e)
    beta <- coef(modelo.e)[2]
    P <- r * anva[1, 2]/beta
    Q <- P/beta
    Sna <- as.numeric(P^2/Q)
    if (r == 1) {
        sc <- sc - Sna
        df <- df - 1
    }
    anva <- anova(modelo.e)
    anva["Residuals", 1] <- df
    anva["Residuals", 2] <- sc
    anva["Residuals", 3] <- sc/df
    anva["Nonadditivity", 2:3] <- r * anva["Nonadditivity", 2:3]
    anva["Nonadditivity", 4] <- anva["Nonadditivity",
3]/anva["Residuals",
        3]
    anva["Nonadditivity", 5] <- 1 - pf(anva["Nonadditivity",
        4], 1, df)
    cat("\nTukey's test of nonadditivity\n")
```

```
    cat(name.y, "\n")
    cat("\nP :", P)
    cat("\nQ :", Q, "\n\n")
    print(anva)
    output <- list(P = P, Q = Q, anva = anva)
    return(output)
}
```

```
normal.freq <-
function (histogram, frequency = 1, ...)
{
    xx <- histogram$mids
    if (frequency == 1)
        yy <- histogram$counts
    if (frequency == 2)
        yy <- histogram$counts/sum(histogram$counts)
    if (frequency == 3)
        yy <- histogram$density
    media <- sum(yy * xx)/sum(yy)
    variancia <- sum(yy * (xx - media)^2)/sum(yy)
    zz <- histogram$breaks
    x1 <- xx[1] - 4 * (zz[2] - zz[1])
    z <- length(zz)
    x2 <- xx[z - 1] + 4 * (zz[z] - zz[z - 1])
    x <- seq(x1, x2, length = 200)
    y <- rep(0, 200)
    area <- 0
    for (k in 1:(z - 1)) area = area + yy[k] * (zz[k + 1] - zz[k])
    for (i in 1:200) {
        y[i] <- area * exp(-((x[i] - media)^2)/(2 *
variancia))/sqrt(2 *
            pi * variancia)
    }
    lines(x, y, ...)
    abline(h = 0)
}
```

```
ojiva.freq <-
function (histogram, ...)
{
    xx <- histogram$mids
    yy <- histogram$counts
    zz <- histogram$breaks
    y1 <- sum(yy)
    nx <- length(xx)
    x <- c(zz, 2 * zz[nx + 1] - zz[nx])
    y <- rep(0, nx + 1)
    for (i in 1:nx) {
        y[i + 1] <- y[i] + yy[i]/y1
    }
    probability <- c(y, 1)
    plot(x, probability, ...)
    table <- data.frame(x = zz, probability = y)
    return(table)
}
```

```
order.group <-
function (trt, means, N, MSerror, Tprob, std.err, parameter = 1)
```

```
{
    N <- rep(1/mean(1/N), length(N))
    n <- length(means)
    z <- data.frame(trt, means, N, std.err)
    w <- z[order(z[, 2], decreasing = TRUE), ]
    M <- rep("", n)
    k <- 1
    j <- 1
    k <- 1
    cambio <- n
    cambio1 <- 0
    chequeo = 0
    M[1] <- letters[k]
    while (j < n) {
        chequeo <- chequeo + 1
        if (chequeo > n)
            break
        for (i in j:n) {
            minimo <- Tprob * sqrt(parameter * MSerror * (1/N[i] +
                1/N[j]))
            s <- abs(w[i, 2] - w[j, 2]) <= minimo
            if (s) {
                if (lastC(M[i]) != letters[k])
                  M[i] <- paste(M[i], letters[k], sep = "")
            }
            else {
                k <- k + 1
                cambio <- i
                cambio1 <- 0
                ja <- j
                for (jj in cambio:n) M[jj] <- paste(M[jj], " ",
                  sep = "")
                M[cambio] <- paste(M[cambio], letters[k], sep = "")
                for (v in ja:cambio) {
                  if (abs(w[v, 2] - w[cambio, 2]) > minimo) {
                    j <- j + 1
                    cambio1 <- 1
                  }
                  else break
                }
                break
            }
        }
        if (cambio1 == 0)
            j <- j + 1
    }
    w <- data.frame(w, stat = M)
    trt <- as.character(w$trt)
    means <- as.numeric(w$means)
    N <- as.numeric(w$N)
    std.err <- as.numeric(w$std.err)
    for (i in 1:n) {
        cat(M[i], "\t", trt[i], "\t", means[i], "\n")
    }
    output <- data.frame(trt, means, M, N, std.err)
    return(output)
}
```

```
order.stat <-
function (treatment, means, minimum)
{
    n <- length(means)
    z <- data.frame(treatment, means)
    w <- z[order(z[, 2], decreasing = TRUE), ]
    M <- rep("", n)
    k <- 1
    k1 <- 0
    j <- 1
    i <- 1
    cambio <- n
    cambio1 <- 0
    chequeo = 0
    M[1] <- letters[k]
    while (j < n) {
        chequeo <- chequeo + 1
        if (chequeo > n)
            break
        for (i in j:n) {
            s <- abs(w[i, 2] - w[j, 2]) <= minimum
            if (s) {
                if (lastC(M[i]) != letters[k])
                  M[i] <- paste(M[i], letters[k], sep = "")
            }
            else {
                k <- k + 1
                cambio <- i
                cambio1 <- 0
                ja <- j
                for (jj in cambio:n) M[jj] <- paste(M[jj], " ",
                  sep = "")
                M[cambio] <- paste(M[cambio], letters[k], sep = "")
                for (v in ja:cambio) {
                  if (abs(w[v, 2] - w[cambio, 2]) > minimum) {
                    j <- j + 1
                    cambio1 <- 1
                  }
                  else break
                }
                break
            }
        }
        if (cambio1 == 0)
            j <- j + 1
    }
    w <- data.frame(w, stat = M)
    trt <- as.character(w$treatment)
    means <- as.numeric(w$means)
    for (i in 1:n) {
        cat(M[i], "\t", trt[i], "\t", means[i], "\n")
    }
    output <- data.frame(trt, means, M)
    return(output)
}
```

```
path.analysis <-
function (corr.x, corr.y)
```

```
{
    if (ncol(corr.y) > 1)
        corr.y <- t(corr.y)
    Direct <- solve(corr.x, corr.y)
    n <- ncol(corr.x)
    Coeff <- corr.x
    for (i in 1:n) {
        for (j in 1:n) {
            Coeff[i, j] <- Direct[j] * corr.x[i, j]
        }
    }
    Residual <- 1 - t(Direct) %*% corr.y
    cat("Correlations\n============\n")
    print(corr.x)
    cat("\n============\n")
    print(corr.y)
    cat("\n\nDirect(Diagonal) and indirect effect path coefficients",
        "\n=============================================================\n")
    print(Coeff)
    cat("\nResidual Effect^2 = ", Residual, "\n")
}
```

```
PBIB.test <-
function (block, trt, replication, y, k, method = "lsd", alpha =
0.05)
{
    block.adj <- as.factor(block)
    trt.unadj <- as.factor(trt)
    replication <- as.factor(replication)
    name.y <- paste(deparse(substitute(y)))
    modelo <- formula(paste(name.y, "~ replication + trt.unadj+
block.adj%in%replication"))
    model <- lm(modelo)
    SCt <- anova(model)[2, 2]
    ntr <- nlevels(trt.unadj)
    r <- nlevels(replication)
    s <- ntr/k
    obs <- ntr * r
    b <- s * r
    glt <- ntr - 1
    glerror <- df.residual(model)
    Ee <- deviance(model)/glerror
    Eb <- anova(model)[3, 3]
    mean.trt <- tapply.stat(y, trt, mean)[, 2]
    X <- rep(0, obs * ntr)
    dim(X) <- c(obs, ntr)
    for (i in 1:obs) {
        tr <- trt[i]
        X[i, tr] <- 1
    }
    R <- rep(0, obs * r)
    dim(R) <- c(obs, r)
    for (i in 1:obs) {
        rp <- replication[i]
        R[i, rp] <- 1
    }
    Z <- rep(0, obs * b)
    dim(Z) <- c(obs, b)
```

```
for (i in 1:obs) {
    rb <- block[i]
    Z[i, rb] <- 1
}
N <- t(X) %* %Z
In <- diag(1, obs)
c0 <- t(Z) %* %(In - (1/r) * X %* %t(X)) %* %y
Js <- diag(s)
Ir <- diag(r)
Jr <- matrix(1, r, r)
Js <- matrix(1, s, s)
Ib <- diag(b)
Iv <- diag(ntr)
q <- k - floor(k/s) * s
if (q <= s/2)
    g <- floor(k/s)
if (q > s/2)
    g <- floor(k/s) + 1
phi <- r * (Eb - Ee)/((r - 1) * Ee)
lambda <- 1/(r * k * (1/phi + 1) - k)
W <- t(N) %* %N - k * Ib - g * kronecker((Jr - Ir), Js)
inversa <- pseudoinverse(Ib - lambda * W)
tauIntra <- t(X) %* %y/r - lambda * N %* %inversa %* %c0
vartau <- (Ee/r) * (Iv + lambda * N %* %inversa %* %t(N))
vardif <- matrix(0, ntr, ntr)
for (i in 1:(ntr - 1)) {
    for (j in (i + 1):ntr) {
        vardif[i, j] <- vartau[i, i] + vartau[j, j] - 2 *
            vartau[i, j]
        vardif[j, i] <- vardif[i, j]
    }
}
cat("\nANALYSIS PBIB: ", name.y, "\nClass level information\n")
cat("\nBlocks: ", b)
cat("\nTrts  : ", ntr)
cat("\n\nNumber of observations: ", length(y), "\n\n")
print(anova(model))
cat("\ncoefficient of variation:", round(cv.model(model),
    1), "%\n")
cat(name.y, "Means:", mean(y, na.rm = TRUE), "\n")
cat("\nTreatments\n")
cat("\nParameters PBIB")
cat("\ntreatmeans :", ntr)
cat("\nBlock size :", k)
cat("\nBlocks/rep :", b/r)
cat("\nReplication:", r, "\n")
E <- (ntr - 1) * (r - 1)/((ntr - 1) * (r - 1) + r * (s -
    1))
cat("\nEfficiency factor", E, "\n")
comb <- combn(ntr, 2)
nn <- ncol(comb)
dif <- rep(0, nn)
stdt <- rep(0, nn)
pvalue <- rep(0, nn)
for (k in 1:nn) {
    i <- comb[1, k]
    j <- comb[2, k]
    dif[k] <- abs(tauIntra[i] - tauIntra[j])
```

```
        stdt[k] <- sqrt(vartau[i, i] + vartau[j, j] - 2 * vartau[i,
            j])
        tc <- dif[k]/stdt[k]
        if (method == "lsd")
            pvalue[k] <- 2 * round(1 - pt(tc, glerror), 4)
        if (method == "tukey")
            pvalue[k] <- round(1 - ptukey(tc, ntr, glerror),
                4)
    }
    tr.i <- comb[1, ]
    tr.j <- comb[2, ]
    cat("\nComparison between treatments means\n")
    cat("\n<<< to see the objects: comparison and means  >>>\n\n")
    comparison <- data.frame(row.names = NULL, tr.i, tr.j, diff =
dif,
        stderr = stdt, pvalue = pvalue)
    means <- data.frame(trt = 1:ntr, means = mean.trt, mean.adj =
as.numeric(tauIntra),
        N = r, std.err = sqrt(diag(vartau)))
    return(list(comparison = comparison, means = means, vartau =
vartau))
}
```

```
polygon.freq <-
function (histogram, frequency = 1, ...)
{
    xx <- histogram$mids
    zz <- histogram$breaks
    if (frequency == 1)
        yy <- histogram$counts
    if (frequency == 2)
        yy <- histogram$counts/sum(histogram$counts)
    if (frequency == 3)
        yy <- histogram$density
    x1 <- xx[1] - zz[2] + zz[1]
    z <- length(zz)
    x2 <- xx[z - 1] + zz[z] - zz[z - 1]
    xx <- c(x1, xx, x2)
    yy <- c(0, yy, 0)
    lines(xx, yy, ...)
    abline(h = 0)
}
```

```
random.ab <-
function (p, q)
{
    t <- rep(0, 3 * p * q)
    dim(t) <- c(p * q, 3)
    k <- 0
    for (i in 1:p) {
        for (j in 1:q) {
            k <- k + 1
            t[k, 1] <- i
            t[k, 2] <- j
            t[k, 3] <- runif(1)
        }
    }
    return(t[order(t[, 3]), -3])
```

```
}
```

```
reg.homog <-
function (trt, y, x)
{
    sumx <- function(x) (nrow(x) - 1) * var(x)
    datos <- data.frame(trt, y, x)
    sumxy <- by(datos[, -1], trt, function(x) sumx(x))
    t.vec <- as.character(unique(datos$trt))
    r.total <- nrow(datos)
    n.trt <- length(t.vec)
    sx <- 0
    sy <- 0
    sxy <- 0
    residual <- 0
    for (i in 1:n.trt) {
        a1 <- data.frame(sumxy[t.vec[i]])
        a2 <- as.matrix(a1)
        residual <- residual + a2[2, 2] - a2[1, 2]^2/a2[1, 1]
        sx <- sx + a2[1, 1]
        sy <- sy + a2[2, 2]
        sxy <- sxy + a2[1, 2]
    }
    B <- sy - sxy^2/sx
    A <- residual
    diff <- B - A
    gl.trt <- n.trt - 1
    gl.r <- r.total - 2 * n.trt
    f.cal <- ((B - A)/gl.trt)/(A/gl.r)
    p.value <- 1 - pf(f.cal, gl.trt, gl.r)
    resp <- "homogeneity of regressions exists"
    if (p.value <= 0.05)
        resp <- "homogeneity of regressions does not exists"
    cat("\nTest of Homogeneity of regressions\n\n")
    cat("Total of simple regressions: ", n.trt, "\n")
    cat("Total of residual          : ", A, "\n")
    cat("Difference for homogeneity : ", B - A, "\n\n")
    cat("D.f. for the homogeneity   : ", gl.trt, "\n")
    cat("Residual degrees of freedom: ", gl.r, "\n")
    cat("F calculated value         : ", f.cal, "\n")
    cat("P.value                    : ", p.value, "\n\n")
    cat("Criterion                  : ", resp, "\n\n")
    output <- list(regressions = n.trt, residual = A, Difference = B,
        DF.homgeneity = gl.trt, DF.Residual = gl.r, F.value = f.cal,
        P.value = p.value, Criterion = resp)
    return(output)
}
```

```
resampling.cv <-
function (A, size, npoints)
{
    nc.A <- ncol(A)
    orden <- 1:nc.A
    y <- trunc(seq(2, (nc.A - 1), length = npoints))
    marcador <- unique(y)
    nmarca <- length(marcador)
    CV <- rep(0, nmarca)
    inicio <- Sys.time()
```

```
    i <- 0
    for (k in marcador) {
        cv.m <- 0
        for (m in 1:size) {
            muestra <- sample(orden, k)
            B <- A[, muestra]
            cv.m <- cv.m + cv.similarity(B)
        }
        i <- i + 1
        CV[i] <- cv.m/size
    }
    final <- Sys.time()
    time <- final - inicio
    cat("\nTime of process ...", time, "\n")
    tabla.cv <- data.frame(marcador = marcador, CV)
    modelo <- lm(CV ~ I(1/marcador))
    return(list(model = modelo, table.cv = tabla.cv))
}
```

```
resampling.model <-
function (k, data, model)
{
    modelo <- model
    parte <- strsplit(model, "~")[[1]]
    model <- as.formula(model)
    ecuacion <- lm(model, data = data)
    xx <- data.frame(anova(ecuacion), NA)
    yy <- ecuacion$model[[1]]
    fc <- xx[, 4]
    names(xx) <- c("Df", "Sum Sq", "Mean Sq", "F value", "Pr(>F)",
        "Resampling")
    m <- nrow(data)
    gk <- nrow(xx) - 1
    f <- rep(0, gk)
    cuenta <- rep(0, gk)
    model <- paste("y", "~", parte[2])
    model <- as.formula(model)
    for (i in 1:k) {
        y <- sample(yy, m)
        resample <- lm(model, data = data)
        for (j in 1:gk) {
            f[j] <- anova(resample)[j, 4]
            if (f[j] >= fc[j])
                cuenta[j] <- cuenta[j] + 1
        }
    }
    for (j in 1:gk) {
        xx[j, 6] <- cuenta[j]/k
    }
    cat("\nResampling of the experiments\n")
    cat(rep("-", 14), "\n")
    cat("Proposed model:", modelo, "\n")
    cat("---\n")
    cat("Resampling of the analysis of variancia for the proposed
model\n")
    cat("Determination of the P-Value by Resampling\n")
    cat("Samples:", k, "\n\n")
    xx <- as.matrix(xx)
```

```
    print(xx, na.print = "")
    cat("---\n\n")
    return(list(solution = xx, acum = cuenta, samples = k))
}
```

```
similarity <-
function (A)
{
    nc <- ncol(A)
    nf <- nrow(A)
    matriz <- rep(0, nf * nf)
    dim(matriz) <- c(nf, nf)
    dimnames(matriz) <- list(row.names(A), row.names(A))
    for (k1 in 1:(nf - 1)) {
        for (k2 in (k1 + 1):nf) {
            mm <- na.omit(c(A[k1, ] == A[k2, ]))
            npar <- length(mm)
            sii <- sum(mm)
            matriz[k2, k1] <- sii/npar
        }
    }
    distance <- as.dist(matriz)
    return(distance)
}
```

```
simulation.model <-
function (k, file, model, categorical = NULL)
{
    modelo <- model
    parte <- strsplit(model, "~")[[1]]
    model <- as.formula(model)
    posicion <- 0
    if (length(categorical) > 0) {
        posicion <- categorical
        n <- length(posicion)
        for (i in 1:n) {
            pos <- posicion[i]
            file[, pos] <- as.factor(file[, pos])
        }
    }
    ecuacion <- lm(model, data = file)
    xx <- data.frame(anova(ecuacion))
    xx[, 2] <- xx[, 4]
    fc <- xx[, 4]
    names(xx) <- c("Df", "F value", "% Acceptance", "% Rejection",
        "Criterion")
    gl <- anova(ecuacion)$Df
    gk <- length(gl) - 1
    xx <- xx[-(gk + 1), ]
    predicho <- predict(ecuacion)
    m <- length(predicho)
    sd.model <- sqrt(deviance(ecuacion)/gl[gk + 1])
    f <- rep(0, k)
    cuenta <- rep(0, gk)
    model <- paste("y", "~", parte[2])
    model <- as.formula(model)
    for (i in 1:k) {
        errores <- rnorm(m, 0, sd.model)
```

```
        y <- predicho + errores
        simula <- lm(model, data = file)
        for (j in 1:gk) {
            f[j] <- anova(simula)[j, 4]
            if (f[j] >= fc[j])
                cuenta[j] <- cuenta[j] + 1
        }
    }
    for (j in 1:gk) {
        xx[j, 3] <- cuenta[j] * 100/k
        xx[j, 4] <- 100 - xx[j, 3]
        if (xx[j, 3] > 50)
            xx[j, 5] <- "acceptable"
        if (xx[j, 3] == 50)
            xx[j, 5] <- "uncertain"
        if (xx[j, 3] < 50)
            xx[j, 5] <- "nonacceptable"
    }
    cat("\nSimulation of experiments\nUnder the normality
assumption\n")
    cat(rep("-", 16), "\n")
    cat("Proposed model:", modelo, "\n")
    print(anova(ecuacion))
    cat("---\n")
    cat("Validation of the analysis of variancia for the proposed
model\n")
    cat("Simulations:", k, "\n\n")
    print(xx)
    cat("---\n\n")
}
```

```
skewness <-
function (x)
{
    x <- na.omit(x)
    n <- length(x)
    s <- sqrt(var(x))
    suma <- sum((x - mean(x))^3)/s^3
    k <- n * suma/((n - 1) * (n - 2))
    return(k)
}
```

```
stability.nonpar <-
function (data, variable = NULL, ranking = FALSE)
{
    row.names(data) <- data[, 1]
    data <- data[, -1]
    audpc <- as.matrix(data)
    rnames <- rownames(data)
    cnames <- colnames(data)
    n <- ncol(audpc)
    k <- nrow(audpc)
    lineas <- matrix(0, k)
    ldev <- matrix(0, k)
    s2 <- matrix(0, k)
    s1 <- matrix(0, k)
    mr <- matrix(0, k)
    ambientes <- matrix(0, 1, n)
```

```
Maudpc <- mean(audpc)
for (i in 1:k) {
    lineas[i] <- mean(audpc[i, ])
    ldev[i] <- lineas[i] - Maudpc
}
cyld <- matrix(0, k, n)
rcyld <- matrix(0, k, n)
raudpc <- matrix(0, k, n)
dimnames(raudpc) <- list(rnames, cnames)
for (j in 1:n) ambientes[j] <- mean(audpc[, j])
for (i in 1:k) {
    for (j in 1:n) cyld[i, j] <- audpc[i, j] - ldev[i]
}
for (j in 1:n) {
    rcyld[, j] <- rank(cyld[, j])
    raudpc[, j] <- rank(audpc[, j])
}
nn = n * (n - 1)/2
for (i in 1:k) {
    sumq <- 0
    for (j in 1:(n - 1)) for (jj in (j + 1):n) sumq <- sumq +
        abs(rcyld[i, j] - rcyld[i, jj])
    s1[i] = sumq/nn
}
for (i in 1:k) {
    s2[i] <- var(rcyld[i, ])
}
k2 <- k^2
es1 <- (k2 - 1)/(3 * k)
es2 <- (k2 - 1)/12
vs1 <- k2 * ((k^2 - 4) * (n + 3) + 30)/(45 * k^2 * n * (n -
    1))
vs2 <- (k2 - 1) * (2 * (k2 - 4) * (n - 1) + 5 * (k2 - 1))/(360 *
    n * (n - 1))
alphas <- 1 - 0.05/k
chi.ind <- qchisq(alphas, 1)
chi.sum <- qchisq(0.95, k)
z1 <- (s1 - es1)^2/vs1
z2 <- (s2 - es2)^2/vs2
suma.z1 <- sum(z1)
suma.z2 <- sum(z2)
mr <- rank(lineas)
stat1 <- data.frame(lineas, mr, s1, z1, s2, z2)
row.names(stat1) <- rnames
stat2 <- data.frame(MEAN = Maudpc, es1, es2, vs1, vs2, chi.ind,
    chi.sum)
cat("\n")
cat("Nonparametric Method for Stability Analysis\n")
cat("-----------------------------------------\n\n")
cat("Estimation and test of nonparametric measures\n")
cat("Variable:", variable, "\n\n")
if (ranking) {
    cat("Ranking...\n")
    print(raudpc)
    cat("\n")
}
cat("Statistics...\n")
names(stat1) <- c("Mean", "Rank", "s1", "Z1", "s2", "Z2")
```

```
    print(round(stat1, 2))
    cat("------------------------")
    cat("\nSum of Z1: ", suma.z1)
    cat("\nSum of Z2: ", suma.z2)
    cat("\n------------------------\n\n")
    cat("Test...\n")
    cat("The Z-statistics are measures of stability. The test for the
significance\n")
    cat("of the sum of Z1 or Z2 are compared to a Chi-Square value of
chi.sum. \n")
    cat("individual Z1 or Z2 are compared to a Chi-square value of
chi.ind.\n\n")
    print(stat2)
    cat("---\n")
    cat("expectation and variance: es1, es2, vs1, vs2\n")
}
```

```
stability.par <-
function (data, rep, MSerror, alpha = 0.1, main = NULL, cova = FALSE,
    name.cov = NULL, file.cov = 0)
{
    KK <- "Environmental index"
    y <- data
    if (cova) {
        x <- as.matrix(file.cov)
        KK <- name.cov
    }
    A <- nrow(y)
    M <- ncol(y)
    N <- rep
    MKE <- MSerror
    RR <- main
    FM0 <- qf(1 - alpha, M - 1, M * (A - 1) * (N - 1))
    dimA <- rep(0, A)
    dim(dimA) <- A
    SHV <- dimA
    MV <- dimA
    SU <- dimA
    MV1 <- dimA
    FF <- dimA
    GY <- dimA
    U1 <- dimA
    G <- dimA
    SI <- dimA
    B <- dimA
    SA <- dimA
    S <- dimA
    FS <- dimA
    FSS <- dimA
    R <- dimA
    GYS <- dimA
    F1 <- dimA
    GY <- dimA
    NN <- dimA
    X1 <- dimA
    X1M <- dimA
    W <- dimA
    GYY <- dimA
```

```
MMM <- dimA
dimA <- rep(0, M)
dim(dimA) <- M
SHM <- dimA
MM <- dimA
II <- dimA
X2 <- dimA
X2M <- dimA
dimA <- rep(0, A * M)
dim(dimA) <- c(A, M)
U <- dimA
G1 <- dimA
SV <- 0
SM <- 0
GG1 <- 0
L <- 0
SMES <- 0
SS <- sum(y^2)
SHT <- sum(y)
for (i in 1:A) {
    SHV[i] <- sum(y[i, ])
    MV[i] <- SHV[i]/M
    SV <- SV + SHV[i]^2/M
}
FK <- SHT^2/(A * M)
SKV <- (SV - FK) * N
MKV = SKV/(A - 1)
for (j in 1:M) {
    SHM[j] <- sum(y[, j])
    MM[j] <- SHM[j]/A
    SM <- SM + SHM[j]^2/A
    II[j] <- MM[j] - SHT/(A * M)
}
SKM <- (SM - FK) * N
MKM <- SKM/(M - 1)
SKT <- (SS - FK) * N
SKVM <- SKT - SKV - SKM
MKVM <- SKVM/((A - 1) * (M - 1))
for (i in 1:A) {
    for (j in 1:M) {
        U[i, j] <- y[i, j] - MM[j]
        SU[i] <- SU[i] + U[i, j]
    }
    U1[i] <- SU[i]/M
}
b <- 1/((M - 1) * (A - 1) * (A - 2))
for (i in 1:A) {
    for (j in 1:M) {
        G[i] <- G[i] + (U[i, j] - U1[i])^2
    }
    GG1 <- GG1 + G[i]
}
for (i in 1:A) {
    SI[i] <- b * (A * (A - 1) * G[i] - GG1) * N
}
if (cova)
    ZZ <- sum(x^2)
if (!cova)
```

```
    IN <- sum(II^2)
for (i in 1:A) {
    B[i] <- 0
    for (j in 1:M) {
        if (cova)
            B[i] <- B[i] + ((U[i, j] - U1[i]) * x[j]/ZZ)
        if (!cova)
            B[i] <- B[i] + ((U[i, j] - U1[i]) * II[j]/IN)
    }
}
for (i in 1:A) {
    SA[i] <- 0
    for (j in 1:M) {
        if (cova)
            SA[i] <- SA[i] + (U[i, j] - U1[i] - B[i] * x[j])^2
        if (!cova)
            SA[i] <- SA[i] + (U[i, j] - U1[i] - B[i] * II[j])^2
    }
}
for (i in 1:A) {
    L <- L + SA[i]
}
for (i in 1:A) {
    S[i] <- (A/((A - 2) * (M - 2))) * (SA[i] - L/(A * (A -
        1))) * N
}
KI <- ((A - 1) * (M - 2))/A
SS <- 0
for (i in 1:A) {
    SS <- SS + S[i] * KI
}
SKMB <- SS
MS <- SKMB/((A - 1) * (M - 2))
SKH <- SKVM - SKMB
MKH <- SKH/(A - 1)
SHKLT <- M * (A - 1) * (N - 1)
T05 <- qt(0.95, SHKLT)
DMV05 <- T05 * sqrt(2 * MKE/(N * M))
for (i in 1:A) {
    SMES <- SMES + MV[i]
}
MES <- SMES/A
for (i in 1:A) {
    if (MV[i] > MES)
        MV1[i] <- 1
    if (MV[i] >= (MES + DMV05))
        MV1[i] <- 2
    if (MV[i] >= (MES + 2 * DMV05))
        MV1[i] <- 3
    if (MV[i] < MES)
        MV1[i] <- -1
    if (MV[i] <= (MES - DMV05))
        MV1[i] <- -2
    if (MV[i] <= (MES - 2 * DMV05))
        MV1[i] <- -3
}
FV <- MKV/MKVM
FM <- MKM/MKE
```

```
FVM <- MKVM/MKE
FH <- MKH/MS
FMS <- MS/MKE
for (i in 1:A) {
    FS[i] <- SI[i]/MKE
    FSS[i] <- S[i]/MKE
}
SHF2 <- (A - 1) * (M - 1)
SHF1 = (A - 1)
F05 <- qf(0.95, SHF1, SHF2)
F01 <- qf(0.99, SHF1, SHF2)
pvalue <- round(1 - pf(FV, SHF1, SHF2), 3)
DD <- paste("", pvalue)
if (pvalue < 0.001)
    DD <- "<0.001"
SHF2 <- M * (A - 1) * (N - 1)
SHF1 <- M - 1
F05 <- qf(0.95, SHF1, SHF2)
F01 <- qf(0.99, SHF1, SHF2)
pvalue <- round(1 - pf(FM, SHF1, SHF2), 3)
NNN <- paste("", pvalue)
if (pvalue < 0.001)
    NNN <- "<0.001"
SHF2 <- M * (A - 1) * (N - 1)
SHF1 <- (A - 1) * (M - 1)
F05 <- qf(0.95, SHF1, SHF2)
F01 <- qf(0.99, SHF1, SHF2)
pvalue <- round(1 - pf(FVM, SHF1, SHF2), 3)
LL <- paste("", pvalue)
if (pvalue < 0.001)
    LL <- "<0.001"
SHF2 <- (A - 1) * (M - 2)
SHF1 <- A - 1
F05 <- qf(0.95, SHF1, SHF2)
F01 <- qf(0.99, SHF1, SHF2)
pvalue <- round(1 - pf(FH, SHF1, SHF2), 3)
HH <- paste("", pvalue)
if (pvalue < 0.001)
    HH <- "<0.001"
SHF2 <- M * (A - 1) * (N - 1)
SHF1 <- (A - 1) * (M - 2)
F05 <- qf(0.95, SHF1, SHF2)
F01 <- qf(0.99, SHF1, SHF2)
pvalue <- round(1 - pf(FMS, SHF1, SHF2), 4)
BB <- paste("", pvalue)
if (pvalue < 0.001)
    BB <- "<0.001"
SHF2 <- M * (A - 1) * (N - 1)
SHF1 <- M - 1
F05 <- qf(0.95, SHF1, SHF2)
F01 <- qf(0.99, SHF1, SHF2)
for (i in 1:A) {
    if (FS[i] >= F01)
        NN[i] <- "**"
    if ((FS[i] < F01) & (FS[i] >= F05))
        NN[i] <- "*"
    if (FS[i] < F05)
        NN[i] <- "ns"
```

```
    }
    SHF2 <- M * (A - 1) * (N - 1)
    SHF1 <- M - 2
    F05 <- qf(0.95, SHF1, SHF2)
    F01 <- qf(0.99, SHF1, SHF2)
    for (i in 1:A) {
        if (FSS[i] >= F01)
            MMM[i] <- "**"
        if ((FSS[i] < F01) & (FSS[i] >= F05))
            MMM[i] <- "*"
        if (FSS[i] < F05)
            MMM[i] <- "ns"
    }
    cat("\n", "INTERACTIVE PROGRAM FOR CALCULATING SHUKLA'S STABILITY
VARIANCE AND KANG'S")
    cat("\n", "                          YIELD - STABILITY (YSi)
STATISTICS")
    cat("\n\n", RR, "\n", KK, " - covariate \n")
    cat("\n", "Analysis of Variance")
    cat("\n", rep("-", 35))
    cat("\n", "Source          d.f.   Sum of Squares  Mean Squares
F  p.value")
    cat("\n", rep("-", 35))
    fuentes <- c("TOTAL       ", "GENOTYPES", "ENVIRONMENTS",
        "INTERACTION", "HETEROGENEITY", "RESIDUAL", "POOLED ERROR")
    gl <- c(A * M - 1, A - 1, M - 1, (M - 1) * (A - 1), A - 1,
        (A - 1) * (M - 2), M * (A - 1) * (N - 1))
    SC <- round(c(SKT, SKV, SKM, SKVM, SKH, SKMB), 4)
    SC <- c(SC, 0)
    CM <- round(c(MKV, MKM, MKVM, MKH, MS, MKE), 4)
    CM <- c(0, CM)
    Fca1 <- round(c(FV, FM, FVM, FH, FMS), 2)
    Fcal <- c(0, Fcal, 0)
    resul <- c(0, DD, NNN, LL, HH, BB, 0)
    Z <- data.frame(fuentes, gl, SC, CM, Fcal, resul)
    Z <- as.matrix(Z)
    Z[1, 4:6] <- " "
    Z[7, c(3, 5, 6)] <- "       "
    for (i in 1:7) {
        cat("\n", Z[i, 1], "\t", Z[i, 2], "\t", Z[i, 3], "\t",
            Z[i, 4], "\t", Z[i, 5], Z[i, 6])
    }
    cat("\n", rep("-", 35))
    for (i in 1:A) {
        for (j in 1:M) {
            X1[i] <- X1[i] + y[i, j]
        }
        X1M[i] <- X1[i]/M
    }
    for (j in 1:M) {
        for (i in 1:A) {
            X2[j] <- X2[j] + y[i, j]
        }
        X2M[j] <- X2[j]/A
    }
    SH <- 0
    for (j in 1:M) {
        SH <- SH + X2[j]
```

```
    }
    MM1 <- SH/(A * M)
    for (i in 1:A) {
        for (j in 1:M) {
            W[i] <- W[i] + (y[i, j] - X1M[i] - X2M[j] + MM1)^2 *
                N
        }
    }
    cat("\n\n", "Stability statistics")
    cat("\n", rep("-", 35))
    cat("\nGenotype    MEANS       Sigma-square      s-square
Ecovalence")
    cat("\n", rep("-", 35))
    MV <- round(MV, 6)
    SI <- round(SI, 6)
    S <- round(S, 6)
    W <- round(W, 6)
    Z <- data.frame(MV, SI, NN, S, MMM, W)
    Z <- as.matrix(Z)
    for (i in 1:A) {
        cat("\n", i, "\t", Z[i, 1], "\t", Z[i, 2], Z[i, 3], "\t",
            Z[i, 4], Z[i, 5], "\t", Z[i, 6])
    }
    FF <- SI/MKE
    SHF2 <- M * (A - 1) * (N - 1)
    SHF1 <- M - 1
    F05 <- qf(0.95, SHF1, SHF2)
    F01 <- qf(0.99, SHF1, SHF2)
    for (i in 1:A) {
        if (FF[i] < FM0)
            F1[i] <- 0
        if (FF[i] >= FM0)
            F1[i] <- -2
        if (FF[i] >= F05)
            F1[i] <- -4
        if (FF[i] >= F01)
            F1[i] <- -8
    }
    for (i in 1:A) {
        R0 <- 1
        for (j in 1:A) {
            if (MV[j] < MV[i])
                R0 <- R0 + 1
        }
        R[i] <- R0
    }
    for (i in 1:A) GY[i] <- R[i] + MV1[i]
    for (i in 1:A) GYS[i] <- GY[i] + F1[i]
    SGYS <- 0
    for (i in 1:A) SGYS <- SGYS + GYS[i]
    MGYS <- SGYS/A
    for (i in 1:A) {
        GYY[i] <- ""
        if (GYS[i] > MGYS)
            GYY[i] <- "+"
    }
    cat("\n\nSignif. codes:  0 '**' 0.01 '*' 0.05 'ns' 1\n\n")
    cat("Simultaneous selection for yield and stability  (++)\n\n")
```

```
    names <- c("Genotype", "Yield", "Rank", "Adj.rank", "Adjusted",
        "Stab.var", "Stab.rating", "YSi", "...")
    Z <- data.frame(Genotype = rownames(y), MV, R, MV1, GY, SI,
        F1, GYS, GYY)
    names(Z) <- names
    print(data.frame(row.names = NULL, Z))
    cat("\n", "Yield Mean:", MES)
    cat("`\n", "YS    Mean:", MGYS)
    cat("\n", "LSD (0.05):", DMV05)
    cat("\n", rep("-", 11))
    cat("\n", "+   selected genotype")
    cat("\n", "++  Reference: Kang, M. S. 1993. Simultaneous
selection for yield")
    cat("\n", "and stability: Consequences for growers. Agron. J.
85:754-757.")
    cat("\n")
}
```

```
stat.freq <-
function (histogram)
{
    xx <- histogram$mids
    yy <- histogram$counts
    media <- sum(yy * xx)/sum(yy)
    variancia <- sum(yy * (xx - media)^2)/(sum(yy) - 1)
    zz <- histogram$breaks
    z <- length(zz)
    names(yy) <- 1:length(yy)
    id <- as.numeric(names(yy[max(yy) == yy]))
    x1 <- xx[1] - (zz[2] - zz[1])
    x2 <- xx[z - 1] + (zz[z] - zz[z - 1])
    zz <- c(x1, zz, x2)
    yy <- c(0, yy, 0)
    z <- z + 2
    names(yy) <- 1:length(yy)
    total <- sum(yy)
    suma <- 0
    i <- 0
    while (suma <= total/2) {
        i <- i + 1
        suma <- suma + yy[i]
    }
    tic <- zz[i + 1] - zz[i]
    mediana <- zz[i] + (total/2 - suma + yy[i]) * tic/yy[i]
    mediana <- as.numeric(mediana)
    id <- as.numeric(names(yy[max(yy) == yy]))
    modas <- length(id)
    clases <- rep(0, 2 * modas)
    for (i in 1:modas) {
        j <- id[i]
        k <- 2 * i - 1
        clases[k] <- zz[j]
        clases[k + 1] <- zz[j + 1]
    }
    dim(clases) <- c(2, modas)
    clases <- t(clases)
    colnames(clases) <- c("[-", "-]")
    mode <- rep(0, modas)
```

```
    for (i in 1:modas) {
        j <- id[i]
        delta1 <- yy[j] - yy[j - 1]
        delta2 <- yy[j] - yy[j + 1]
        tic <- zz[j + 1] - zz[j]
        mode[i] <- zz[j] + delta1 * tic/(delta1 + delta2)
    }
    Moda <- cbind(clases, mode)
    lista <- list(variance = variancia, mean = media, median =
mediana,
        mode = Moda)
    return(lista)
}
```

```
strip.plot <-
function (BLOCKS, COL, ROW, Y)
{
    name.y <- paste(deparse(substitute(Y)))
    name.col <- paste(deparse(substitute(COL)))
    name.row <- paste(deparse(substitute(ROW)))
    name.block <- paste(deparse(substitute(BLOCKS)))
    cat("\nANALYSIS STRIP PLOT: ", name.y, "\nClass level
information\n")
    COL <- as.factor(COL)
    ROW <- as.factor(ROW)
    BLOCKS <- as.factor(BLOCKS)
    nrep <- length(unique(BLOCKS))
    nCOL <- length(unique(COL))
    nROW <- length(unique(ROW))
    cat("\n", name.col, "\t   (COL): ", unique(as.character(COL)))
    cat("\n", name.row, "\t   (ROW): ", unique(as.character(ROW)))
    cat("\n", name.block, "\t(BLOCKS): ",
unique(as.character(BLOCKS)))
    cat("\n\nNumber of observations: ", length(Y), "\n\n")
    model <- aov(Y ~ BLOCKS + COL + BLOCKS:COL + ROW + BLOCKS:ROW +
        COL:ROW)
    cat("model Y:", name.y, " ~ BLOCKS + COL  + BLOCKS:COL + ROW +
BLOCKS:ROW+ Error(BLOCKS:COL+BLOCKS:ROW)+ COL:ROW\n\n")
    mm <- anova(model)
    nn <- mm[3, ]
    nn1 <- row.names(mm)[3]
    nn2 <- row.names(mm)[4]
    row.names(mm)[4] <- " "
    mm[3, ] <- mm[4, ]
    mm[4, ] <- nn
    row.names(mm)[3] <- nn2
    row.names(mm)[4] <- nn1
    mm[2, 4] <- mm[2, 3]/mm[3, 3]
    mm[2, 5] <- 1 - pf(mm[2, 4], mm[2, 1], mm[3, 1])
    mm[4, 4] <- mm[4, 3]/mm[5, 3]
    mm[4, 5] <- 1 - pf(mm[4, 4], mm[4, 1], mm[5, 1])
    print(mm)
    DFE <- df.residual(model)
    MSE <- deviance(model)/DFE
    medy <- mean(Y)
    cat("\nCoeff var", "\tMean", name.y, "\n")
    cat(sqrt(MSE) * 100/medy, "\t", medy, "\n")
    gl.a <- mm[3, 1]
```

```
    Ea <- mm[3, 3]
    gl.b <- mm[5, 1]
    Eb <- mm[5, 3]
    gl.c <- mm[7, 1]
    Ec <- mm[7, 3]
    output <- list(anva = mm, model = model, gl.a = gl.a, gl.b =
gl.b,
        gl.c = gl.c, Ea = Ea, Eb = Eb, Ec = Ec)
    return(output)
}
```

```
sturges.freq <-
function (x)
{
    maximo <- max(x)
    minimo <- min(x)
    amplitud <- maximo - minimo
    n <- length(x)
    k <- round(1 + 3.33 * log10(n), 0)
    y <- as.character(x)
    z <- rep(0, n)
    for (i in 1:n) {
        lc <- nchar(y[i])
        nd <- 0
        for (j in 1:lc) {
            a <- substr(y[i], j, j)
            if (a != ".")
                nd = nd + 1
            else break
        }
        z[i] <- lc - nd - 1
    }
    d <- max(z)
    if (d < 0)
        d <- 1
    tic <- round(amplitud/k + 0.5 * 10^(-d), d)
    clases <- seq(minimo, maximo, tic)
    nc <- length(clases)
    if (maximo > clases[nc])
        clases <- c(clases, clases[nc] + tic)
    lista <- list(maximum = maximo, minimum = minimo, amplitude =
amplitud,
        classes = k, interval = tic, breaks = clases)
    return(lista)
}
```

```
table.freq <-
function (object)
{
    xx <- object$mids
    yy <- object$counts
    y1 <- sum(yy)
    zz <- object$breaks
    x <- length(xx)
    acum <- 0
    z <- rep(0, 7 * x)
    dim(z) <- c(x, 7)
    for (i in 1:x) {
```

```
        z[i, 1] <- zz[i]
        z[i, 2] <- zz[i + 1]
        z[i, 3] <- xx[i]
        z[i, 4] <- yy[i]
        z[i, 5] <- yy[i]/y1
        z[i, 6] <- yy[i] + acum
        acum <- z[i, 6]
        z[i, 7] <- z[i, 6]/y1
    }
    colnames(z) <- c("Inf", "Sup", "MC", "fi", "fri", "Fi", "Fri")
    rownames(z) <- rep(" ", x)
    return(z)
}
```

```
tapply.stat <-
function (y, x, stat = "mean")
{
    cx <- deparse(substitute(x))
    cy <- deparse(substitute(y))
    x <- data.frame(c1 = 1, x)
    y <- data.frame(v1 = 1, y)
    nx <- ncol(x)
    ny <- ncol(y)
    namex <- names(x)
    namey <- names(y)
    if (nx == 2)
        namex <- c("c1", cx)
    if (ny == 2)
        namey <- c("v1", cy)
    namexy <- c(namex, namey)
    for (i in 1:nx) {
        x[, i] <- as.character(x[, i])
    }
    z <- NULL
    for (i in 1:nx) {
        z <- paste(z, x[, i], sep = "&")
    }
    w <- NULL
    for (i in 1:ny) {
        m <- tapply(y[, i], z, stat)
        m <- as.matrix(m)
        w <- cbind(w, m)
    }
    nw <- nrow(w)
    c <- rownames(w)
    v <- rep("", nw * nx)
    dim(v) <- c(nw, nx)
    for (i in 1:nw) {
        for (j in 1:nx) {
            v[i, j] <- strsplit(c[i], "&")[[1]][j + 1]
        }
    }
    rownames(w) <- NULL
    junto <- data.frame(v[, -1], w)
    junto <- junto[, -nx]
    names(junto) <- namexy[c(-1, -(nx + 1))]
    return(junto)
}
```

```
vark <-
function (x, y)
{
    ties.x <- rle(sort(x))$lengths
    ties.y <- rle(sort(y))$lengths
    n <- length(x)
    t1 <- n * (n - 1) * (2 * n + 5)
    t2 <- sum(ties.x * (ties.x - 1) * (2 * ties.x + 5))
    t3 <- sum(ties.y * (ties.y - 1) * (2 * ties.y + 5))
    v1 <- (t1 - t2 - t3)/18
    t1 <- sum(ties.x * (ties.x - 1) * (ties.x - 2))
    t2 <- sum(ties.y * (ties.y - 1) * (ties.y - 2))
    v2 <- (t1 * t2)/(9 * n * (n - 1) * (n - 2))
    t1 <- sum(ties.x * (ties.x - 1)) * sum(ties.y * (ties.y -
        1))
    v3 <- t1/(2 * n * (n - 1))
    v1 + v2 + v3
}
```

```
waerden.test <-
function (y, trt, alpha = 0.05, group = TRUE, main = NULL)
{
    name.y <- paste(deparse(substitute(y)))
    name.t <- paste(deparse(substitute(trt)))
    junto <- subset(data.frame(y, trt), is.na(y) == FALSE)
    N <- nrow(junto)
    junto[, 1] <- qnorm(round(rank(junto[, 1])/(N + 1), 3))
    S <- sum(junto[, 1]^2)/(N - 1)
    means <- tapply.stat(junto[, 1], junto[, 2], stat = "mean")
    nn <- tapply.stat(junto[, 1], junto[, 2], stat = "length")
    means <- data.frame(means, replication = nn[, 2])
    names(means)[1:2] <- c(name.t, name.y)
    ntr <- nrow(means)
    DFerror <- N - ntr
    T1 <- 0
    for (i in 1:ntr) {
        T1 <- T1 + means[i, 2]^2 * means[i, 3]
    }
    T1 <- T1/S
    cat("\nStudy:", main)
    cat("\nVan der Waerden (Normal Scores) test's\n")
    cat("\nValue :", T1)
    p.chisq <- 1 - pchisq(T1, ntr - 1)
    cat("\nPvalue:", p.chisq)
    cat("\nDegrees of freedom: ", ntr - 1)
    cat("\n\nMeans of the normal score\n")
    print(data.frame(row.names = NULL, means))
    MSerror <- S * ((N - 1 - T1)/(N - ntr))
    nr <- unique(means[, 3])
    if (group) {
        Tprob <- qt(1 - alpha/2, DFerror)
        cat("\nt-Student:", Tprob)
        cat("\nAlpha    :", alpha)
        if (length(nr) == 1) {
            LSD <- Tprob * sqrt(2 * MSerror/nr)
            cat("\nLSD      :", LSD, "\n")
        }
```

```
        else {
            nr1 <- 1/mean(1/nn[, 2])
            LSD1 <- Tprob * sqrt(2 * MSerror/nr1)
            cat("\nLSD      :", LSD1, "\n")
            cat("\nHarmonic Mean of Cell Sizes ", nr1)
        }
        cat("\nMeans with the same letter are not significantly
different\n")
        cat("\nGroups, Treatments and means of the normal score\n")
        output <- order.group(means[, 1], means[, 2], means[,
            3], MSerror, Tprob, std.err = sqrt(MSerror/means[,
            3]))
    }
    if (!group) {
        comb <- combn(ntr, 2)
        nn <- ncol(comb)
        dif <- rep(0, nn)
        pvalue <- rep(0, nn)
        for (k in 1:nn) {
            i <- comb[1, k]
            j <- comb[2, k]
            dif[k] <- abs(means[i, 2] - means[j, 2])
            sdtdif <- sqrt(S * ((N - 1 - T1)/(N - ntr)) * (1/means[i,
                3] + 1/means[j, 3]))
            pvalue[k] <- 2 * round(1 - pt(dif[k]/sdtdif, DFerror),
                4)
        }
        tr.i <- comb[1, ]
        tr.j <- comb[2, ]
        cat("\nComparison between treatments means\nmean of the
normal score\n")
        print(data.frame(row.names = NULL, tr.i, tr.j, diff = dif,
            pvalue = pvalue))
        output <- data.frame(trt = means[, 1], means = means[,
            2], M = "", N = means[, 3])
    }
    return(output)
}
```

```
waller <-
function (K, q, f, Fc)
{
    a0 <- 1
    b0 <- 20
    for (i in 1:50) {
        t <- (b0 + a0)/2
        g <- function(x, q, f, Fc) x^(-(q + 3)/2) * (f + q *
            Fc/x)^(-(f + q - 1)/2)
        b <- function(x, q, f, Fc) sqrt((f + q) * (x - 1)/(f *
            x + q * Fc))
        h <- function(z, q, f) ((f + q + z^2)/(f + q - 1)) *
            dt(z, q + f) + z * pt(z, q + f)
        n0 <- function(x) sqrt(x - 1) * g(x, q, f, Fc) * h(t *
            b(x, q, f, Fc), q, f)
        d0 <- function(x) sqrt(x - 1) * g(x, q, f, Fc) * h(-t *
            b(x, q, f, Fc), q, f)
        n1 <- function(x) sqrt(x - 1) * g(x, q, f, Fc) * h(a0 *
            b(x, q, f, Fc), q, f)
        d1 <- function(x) sqrt(x - 1) * g(x, q, f, Fc) * h(-a0 *
```

```
            b(x, q, f, Fc), q, f)
        IN0 <- integrate(n0, 1, Inf)$value
        ID0 <- integrate(d0, 1, Inf)$value
        IN1 <- integrate(n1, 1, Inf)$value
        ID1 <- integrate(d1, 1, Inf)$value
        if ((K - IN0/ID0) * (K - IN1/ID1) <= 0)
            b0 <- t
        if ((K - IN0/ID0) * (K - IN1/ID1) > 0)
            a0 <- t
        if (abs(b0 - a0) <= 5e-04)
            break
    }
    return(round(t, 3))
}
```

```
waller.test <-
function (y, trt, DFerror, MSerror, Fc, K = 100, group = TRUE,
    main = NULL)
{
    name.y <- paste(deparse(substitute(y)))
    name.t <- paste(deparse(substitute(trt)))
    junto <- subset(data.frame(y, trt), is.na(y) == FALSE)
    means <- tapply.stat(junto[, 1], junto[, 2], stat = "mean")
    sds <- tapply.stat(junto[, 1], junto[, 2], stat = "sd")
    nn <- tapply.stat(junto[, 1], junto[, 2], stat = "length")
    means <- data.frame(means, std.err = sds[, 2]/sqrt(nn[, 2]),
        replication = nn[, 2])
    names(means)[1:2] <- c(name.t, name.y)
    ntr <- nrow(means)
    Tprob <- waller(K, ntr - 1, DFerror, Fc)
    nr <- unique(nn[, 2])
    nfila <- c("K ratio", "Error Degrees of Freedom", "Error Mean
Square",
        "F value", "Critical Value of Waller")
    nvalor <- c(K, DFerror, MSerror, Fc, Tprob)
    cat("\nStudy:", main)
    cat("\n\nWaller-Duncan K-ratio t Test for", name.y, "\n")
    cat("\nThis test minimizes the Bayes risk under additive")
    cat("\nloss and certain other assumptions.\n")
    xtabla <- data.frame(..... = nvalor)
    row.names(xtabla) <- nfila
    print(xtabla)
    cat("\nTreatment Means\n")
    print(data.frame(row.names = NULL, means))
    if (length(nr) == 1) {
        MSD <- Tprob * sqrt(2 * MSerror/nr)
        cat("\nMinimum Significant Difference", MSD)
    }
    else {
        cat("\nDifferent MSD for each comparison")
    }
    if (group) {
        cat("\nMeans with the same letter are not significantly
different.")
        cat("\n\nComparison of treatments\n\nGroups, Treatments and
means\n")
        output <- order.group(means[, 1], means[, 2], means[,
            4], MSerror, Tprob, means[, 3])
    }
```

```
    if (!group) {
        comb <- combn(ntr, 2)
        nn <- ncol(comb)
        dif <- rep(0, nn)
        MSD1 <- rep(0, nn)
        for (k in 1:nn) {
            i <- comb[1, k]
            j <- comb[2, k]
            dif[k] <- abs(means[i, 2] - means[j, 2])
            MSD1[k] <- Tprob * sqrt(MSerror * (1/means[i, 4] +
                1/means[j, 4]))
        }
        tr.i <- comb[1, ]
        tr.j <- comb[2, ]
        cat("\nComparison between treatments means\n\n")
        if (length(nr) == 1) {
            significant = dif > MSD
            print(data.frame(row.names = NULL, tr.i, tr.j, diff =
dif,
                significant))
        }
        else {
            significant = dif > MSD1
            print(data.frame(row.names = NULL, tr.i, tr.j, diff =
dif,
                MSD = MSD1, significant))
        }
        output <- data.frame(trt = means[, 1], means = means[,
            2], M = "", N = means[, 4], std.err = means[, 3])
    }
    return(output)
}
```

```
wxyz <-
function (model, x, y, z)
{
    datos <- data.frame(x, y, z)
    lista <- by(datos, list(x = x, y = y), mean)
    pp <- as.matrix(lista)
    m <- nrow(pp)
    n <- ncol(pp)
    x1 <- rownames(pp)
    xx <- as.numeric(x1)
    y1 <- colnames(pp)
    yy <- as.numeric(y1)
    z1 <- rep(NA, m * n)
    dim(z1) <- c(m, n)
    for (i in 1:m) {
        for (j in 1:n) {
            if ((length(pp[i, j][[1]][3]) > 0) && !(is.na(pp[i,
                j][[1]][3])))
                z1[i, j] <- pp[i, j][[1]][3]
            else z1[i, j] <- predict(model, data.frame(x = xx[i],
                y = yy[j]))
        }
    }
    dimnames(z1) <- list(x1, y1)
    return(z1)
}
```

ANEXO 9.

MANUAL RÁPIDO DE USO
DE LA LIBRERÍA AGRICOLAE

MANUAL RÁPIDO DE USO DE LA LIBRERÍA AGRICOLAE

Ing. Felipe de Mendiburu

Agosto - 2009

INDICE GENERAL

PRESENTACIÓN

"Agricolae" es una librería de funciones para R. R es un sistema de programación funcional exclusivo para el manejo de datos en estadística y ciencias afines como las matemáticas, en ambientes como Windows, Linux y Mac.

"Agricolae" ofrece una amplia funcionalidad en el diseño de experimentos, especialmente para experimentos en la agricultura y mejoramientos de plantas, los cuales pueden también ser usados para otros propósitos. Soporta la planificación de diseños latice, alfa, cíclico, bloques incompletos balanceado, bloques completos aleatorios, latino y Greco latino. También hay varios procedimientos de análisis de datos experimentales, tales como las comparaciones de tratamientos de Waller-Duncan, Bonferroni o los clásicos como LSD y Tukey, y comparaciones no-paramétricas como Kruskal-Wallis, Friedman y Durbin, análisis de estabilidad y otros procedimientos aplicados en genética, así como además en procedimientos en biodiversidad y estadística descriptiva.

Para más detalles del uso de "agricolae" está en el manual de referencia y el sistema de ayudas del sistema en html, que puede ser invocado desde el menú de R.

1 INSTALACIÓN DE AGRICOLAE Y USO EN R

1.1 INSTALACIÓN

El programa principal de R debe ser instalado en la plataforma de su computadora (Windows, Linux o Mac). Si aún no está instalado, este debe ser descargado del proyecto R (www.r-project.org) del CRAN de un repositorio. Como es libre, no se requiere ninguna identificación. Las librerías pueden ser incorporadas mediante un proceso de instalación, directamente desde la plataforma de R.

"Agricolae" es una librería para R; como tal, su instalación es igual que cualquier otra librería de R.

Para Windows se requiere el programa R versión 2.9.0 o superior.

Si el programa R está instalado en Windows o en otra plataforma, la instalación de "agricolae" puede hacerse directamente desde la consola de R en conexión con Internet, así:

```
install.packages("agricolae")
```

Se selecciona un repositorio y el sistema instala automáticamente.

Si no se tiene conexión a Internet, es necesario el archivo agricolae_1.0-7. zip para Windows, el cual debe ser obtenido de Internet y almacenarse en un medio de computación.

El archivo agricolae_1.0-7.zip puede ser descargado del repositorio de R en la siguientes direcciones: www.r-project.org y
http://cran.at.r-project.org/web/packages/agricolae/index.html

El archivo puede ser incorporado directamente a R instalando desde la consola ejecutando la siguiente instrucción si el archivo esta en la dirección E:
install.packages("E:/agricolae_1.0-7.zip")
También puede ser instalado desde el menú de R:

Packages, Install package(s) from local zip files … , y seleccionar el archivo zip. No require desempaquetar.

"Agricolae" para su completa funcionalidad requiere otras librerías.

SuppDists: para las distribuciones kruskal y friedman, utilizadas en las funciones kruskal() y friedman().
corpcor: para la inversa generalizada utilizada en la función PBIB.test()
klaR: para la función triplot() utilizada en la función AMMI()
akima: para el uso de la función interpp( ) utilizada en grid3p() para interpolación.

1.2 USO EN R

Como "agricolae" es una librería de funciones, éstas funcionan cuando se las invoca directamente desde la consola de R; es decir las condiciones para su funcionalidad dependen de las funciones y datos de R y de otras librerías.

Las siguientes órdenes son frecuentes:
Cargar la librería a la memoria: library(agricolae)
Descargar: detach(package:agricolae)

Una vez cargada la librería, ésta puede ser usada así:
Listar la base de datos: data(package="agricolae")
Cargar los datos de camote: data(sweetpotato)
Ver su estructura: str(sweetpotato)
Editar su contenido: fix(sweetpotato)

Para continuar con la línea de comandos, siempre debe cerrar las ventanas abiertas con alguna orden de R.

Para ayudas: help(sweetpotato); ? sweetpotato
Para la búsqueda de alguna función: apropos("design")

```
[1] "design.ab"      "design.alpha"   "design.bib"     "design.crd"
[5] "design.cyclic"  "design.graeco"  "design.lattice" "design.lsd"
[9] "design.rcbd"    "plot.design"
```

Para uso de simbolos que no esta en el teclado en español, por ejempo: ~, [,], &, ^, |. <, >, {, }, \, % y otros, utilice la tabla 6.10.

2 ESTADÍSTICA DESCRIPTIVA

La librería agricolae proporciona algunas funciones complementarias al programa R, específicamente con el manejo del histograma.

2.1 HISTOGRAMA

El histograma es la representación de datos agrupados en clases contínuas. La función es graph.freq(). Las funciones asociadas a graph.freq() son: polygon.freq, table.freq, stat.freq, intervals.freq, sturges.freq, join.freq, ojiva.freí y normal.freq

Ejemplo 1.1 Datos generados en R. (peso de estudiantes). Figura 2.1

```
c( 68, 53, 69.5, 55, 71, 63, 76.5, 65.5, 69, 75, 76, 57, 70.5, 71.5,
56, 81.5, 69, 59, 67.5, 61, 68, 59.5, 56.5, 73, 61, 72.5, 71.5,
59.5, 74.5, 63) -> peso
```

cargar la librería agricolae:

```
library(agricolae)
par(mfrow=c(2,2))
h1<- graph.freq(peso, col="yellow", frequency =1, main="frecuencia
absoluta\nh1")
h2<- graph.freq(peso,  frequency =2 , main="polígono de
frecuencia\nh2")
polygon.freq(h2, col="blue", lwd=2, frequency =2)
h3<- graph.freq(peso, col="brown", frequency =3 ,
main="densidad\nh3")
h4<- graph.freq(peso, col="blue", frequency =3 , main="densidad
normal\nh4", density=4)
normal.freq(h4, col="red", lty=4,lwd=2, frequency=3)
```

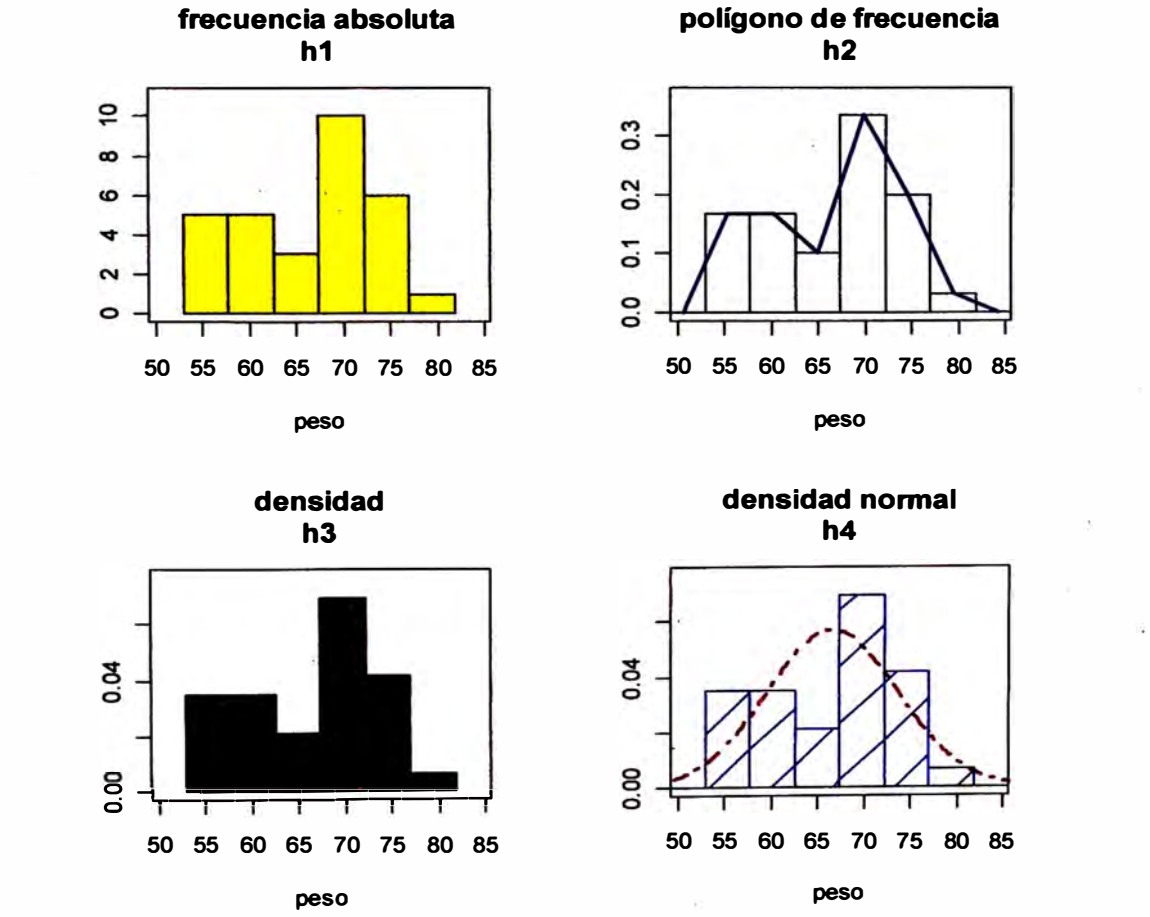


Figura 2.1 Histogramas, poligono y densidad de su normal

2.2 MANEJANDO ESCALAS

Se refiere a los cambios de escala en los ejes. Figura 2.2

```
h5<- graph.freq(peso, axes=FALSE, frequency =1, main="frecuencia
absoluta\nh5")
axis(1,h5$breaks,las=2)
axis(2,h5$count)

h6<- graph.freq(peso, axes=FALSE, nclass=5, main="frecuencia con 5
clases\nh6")
axis(1,h6$breaks,las=2)
axis(2,seq(0,10))
normal.freq(h6,col="red")
h7<- graph.freq(peso, density=6, col="blue", frequency =3 ,
main="densidad\nh7")
lines(density(peso),col="brown",lwd=2)
h8<- graph.freq(peso, border=0, frequency =3 , main="polígono y
densidad\nh8")
polygon.freq(h8,col="blue", frequency =3)
lines(density(peso),col="brown",lwd=2)
```

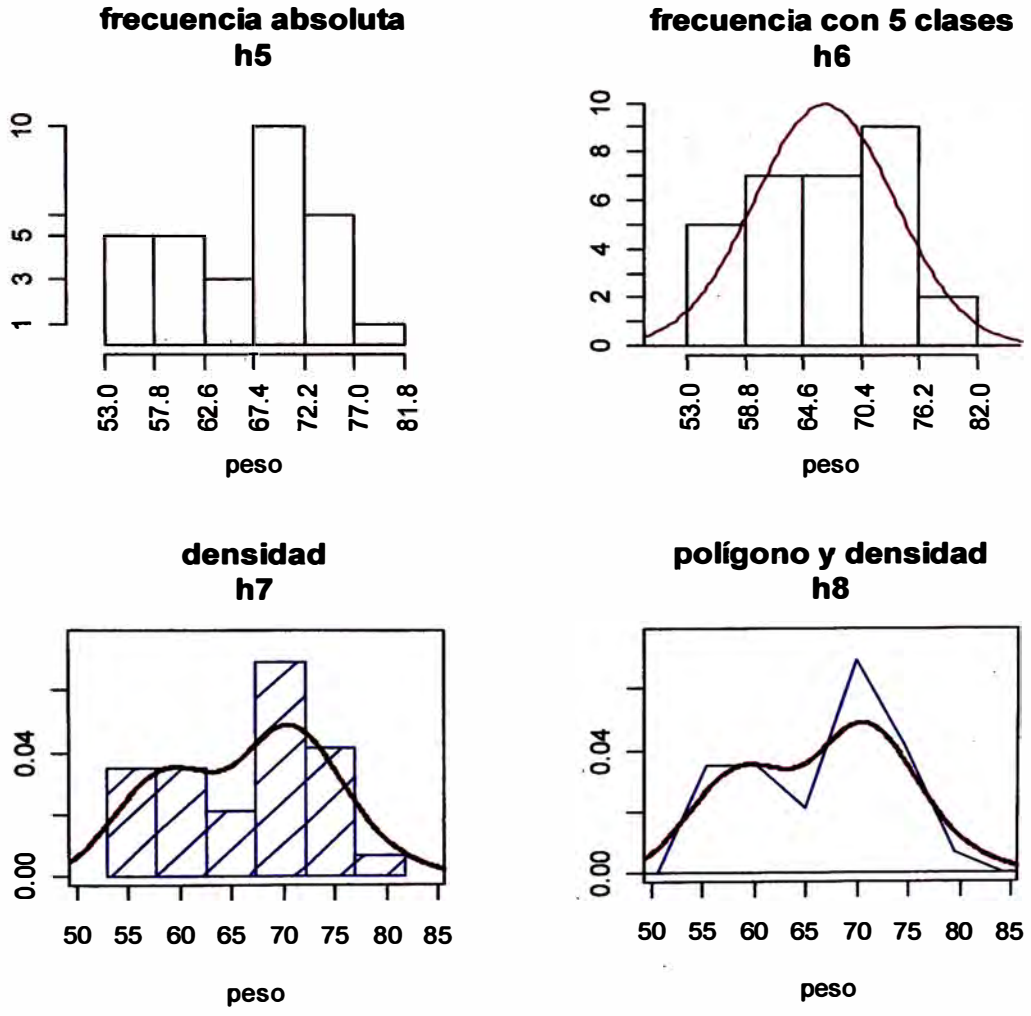


Figura 2.2. Cambio de la escala de los ejes de coordenadas

```
h9<-ojiva.freq(h5,axes=FALSE,type="b", main="ojiva de h5\nh9", col="red")
axis(2,round(h9[,2],1),las=2)
axis(1,round(h9[,1],1),las=2)
```

2.3 TABLAS DE FRECUENCIA Y ESTADÍSTICAS

Redondeado a dos decimales:

```
round(table.freq(h6), 2)
```

Inf	Sup	MC	fi	fri	Fi	Fri
53.0	58.8	55.9	5	0.17	5	0.17
58.8	64.6	61.7	7	0.23	12	0.40
64.6	70.4	67.5	7	0.23	19	0.63
70.4	76.2	73.3	9	0.30	28	0.93
76.2	82.0	79.1	2	0.07	30	1.00

```
stat.freq(h6)
$variance
[1] 50.42133
$mean
[1] 66.72667
$median
[1] 67.08571
$mode
       [-   -]     mode
[1,] 70.4 76.2 71.68889
```

2.4 REPRODUCIENDO HISTOGRAMAS Y LA FUNCION HIST() DE R

La clase de graph.freq() es graph.freí. Figura 2.5

Reproduciendo el histograma 6 (5 clases)

```
h10<-plot(h6, axes=FALSE, main="frecuencia con 5 clases\nh10")
axis(1,h6$breaks,las=2)
axis(2,seq(0,10))
normal.freq(h6,col="red")
summary(h6)
```

Inf	Sup	MC	fi	fri	Fi	Fri
53.0	58.8	55.9	5	0.16666667	5	0.1666667
58.8	64.6	61.7	7	0.23333333	12	0.4000000
64.6	70.4	67.5	7	0.23333333	19	0.6333333
70.4	76.2	73.3	9	0.30000000	28	0.9333333
76.2	82.0	79.1	2	0.06666667	30	1.0000000

La clase de la funcion hist() de R es histogram; sin embargo, es posible establecer una compatibilidad restringida a la escala de frecuencia relativa, y en algunos casos, el número de frecuencia no es el solicitado.

```
hh <- hist(peso,nclass=5, plot=FALSE) # Reporta 7 clases
# hist(peso,nclass=4) # Reporta 4 clases
```

Para recuperar a frecuencias relativas se puede usar graph.freq al objeto hh, pero las clases no pueden ser modificadas.

```
h11<-graph.freq(hh, frequency=2, col=colors()[367]
,main="relativa\nh11" ,axes=F)
axis(1,h11$breaks,las=2)
axis(2,round(h11$relative,2),las=2)
```

Las funciones del histograma de agricolae funcionan bien sobre el objeto creado por hist() de R.

2.5 HISTOGRAMA A PARTIR DE DATOS AGRUPADOS

Si se tiene la siguiente tabla de frecuencia para los intervalos indicados:

0-10	10-20	20-30	30-40	40-50
3	8	15	18	6

Organizando en R se tiene:

```
clases <- c(0, 10, 20, 30, 40, 50)
frec <- c(3, 8, 15, 18, 6)
```

h12 <- graph.freq(clases,counts=frec,xlab="Clases", main="Clases\nh12")

summary(h12)

Inf	Sup	MC	fi	fri	Fi	Fri
0	10	5	3	0.06	3	0.06
10	20	15	8	0.16	11	0.22
20	30	25	15	0.30	26	0.52
30	40	35	18	0.36	44	0.88
40	50	45	6	0.12	50	1.00

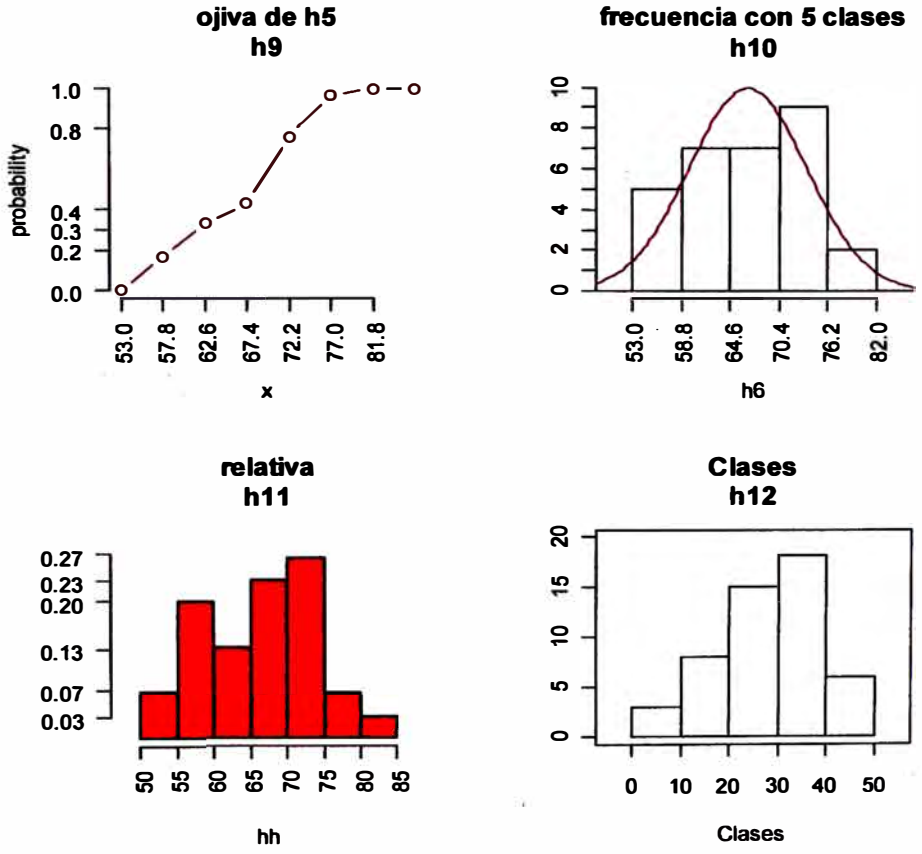


Figura 2.5 Nuevas escalas para los histogramas

Todas las funciones de agricolae pueden aplicarse, incluyendo plot().

2.6 JUNTANDO CLASES

Con la información de los pesos de los estudiantes, los intervalos originales pueden ser cambiados, juntando por ejemplo:

```
intervals.freq(h5$breaks)
     lower upper
[1,]  53.2  58.0
[2,]  58.0  62.8
[3,]  62.8  67.6
[4,]  67.6  72.4
[5,]  72.4  77.2
[6,]  77.2  82.0
```

```
nuevas<- join.freq(h5$breaks,1:2)
intervals.freq(nuevas)
     lower upper
[1,]  53.2  62.8
[2,]  62.8  67.6
[3,]  67.6  72.4
[4,]  72.4  77.2
[5,]  77.2  82.0
h13 <- graph.freq(peso, breaks=nuevas)
```

3 DISEÑO DE EXPERIMENTOS

Para la generación de los diseños experimentales se requiere los nombres de tratamientos, el número de repeticiones y otros parámetros según el tipo de diseño:

3.1 COMPLETAMENTE AL AZAR

```
trt <- c("A", "B", "C")
repeticion <- c(4, 3, 4)
plan1 <- design.crd(trt,r=repeticion)
```

```
   plots trt r
1      1   A 1
2      2   C 1
3      3   A 2
4      4   A 3
5      5   B 1
6      6   C 2
7      7   C 3
8      8   B 2
9      9   A 4
10    10   B 3
11    11   C 4
```

Para excel.
write.csv(plan1,"plan1.csv",row.names=FALSE)

3.2 BLOQUES COMPLETAMENTE AL AZAR

```
trt <- c("A", "B", "C")
repeticion <- 4
plan2 <- design.rcbd(trt,r=repeticion, seed=55, number=101)
t(matrix(plan2[,3],c(3,4)))
     [,1] [,2] [,3]
[1,] "A"  "C"  "B"
[2,] "A"  "B"  "C"
[3,] "B"  "C"  "A"
[4,] "C"  "A"  "B"
```

El plan puede ser enviado a excel como libro de campo.

3.3 CUADRADO LATINO

```
trt <- c("A", "B", "C", "D")
plan3 <- design.lsd(trt, seed=55, number=101)
t(matrix(plan3[,4],c(4,4)))
     [,1] [,2] [,3] [,4]
[1,] "A"  "B"  "D"  "C"
[2,] "B"  "C"  "A"  "D"
[3,] "D"  "A"  "C"  "B"
[4,] "C"  "D"  "B"  "A"
```

3.4 GRECO LATINO

```
T1 <- c("A", "B", "C", "D")
T2 <- 1:4

plan4 <- design.graeco(T1,T2, seed=55, number=101)
t(matrix(paste(plan4[,4],plan4[,5]),c(4,4)))

     [,1]  [,2]  [,3]  [,4]
[1,] "A 3" "D 4" "C 1" "B 2"
[2,] "D 1" "A 2" "B 3" "C 4"
[3,] "C 2" "B 1" "A 4" "D 3"
[4,] "B 4" "C 3" "D 2" "A 1"
```

3.5 BLOQUES INCOMPLETOS BALANCEADOS

```
trt <- c("A", "B", "C", "D", "E" )
k <- 4
plan5 <- design.bib(trt,k, seed=55, number=101)

Parameters BIB
==============
Lambda     : 3
treatmeans : 5
Block size : 4
Blocks     : 5
Replication: 4

Efficiency factor 0.9375

<<< Book >>>
```

Según la información producida, son 5 bloques de tamaño 4, siendo la matriz formada:

```
t(matrix(plan5[,3],c(4,5)))
     [,1] [,2] [,3] [,4]
[1,] "B"  "E"  "C"  "A"
[2,] "D"  "A"  "C"  "B"
[3,] "B"  "C"  "E"  "D"
[4,] "C"  "D"  "A"  "E"
[5,] "E"  "B"  "D"  "A"
```

Se puede observar que los tratamientos tienen 4 repeticiones. El parámetro lambda es 3; este significa que un par de tratamientos esta junto en 3 oportunidades> Por ejemplo, B y E se encuentran en el bloque I, III y V, y así sucesivamente.

3.6 CÍCLICO

Se usa para 6 a 30 tratamientos. Las repeticiones es un múltiplo del tamaño del bloque; si son 6 tratamientos y el tamaño es 3, entonces las repeticiones pueden ser 6, 9, 12, etc.

```
trt <- c("A", "B", "C", "D", "E", "F" )
plan5 <- design.cyclic(trt,k=3, r=6, seed=55, number=101)

cyclic design
Generator block basic:
1 2 4
1 3 2

Parameters
==========
treatmeans : 6
Block size : 3
Replication: 6

> plan5$design[[1]]
     [,1] [,2] [,3]
[1,] "F"  "A"  "C"
[2,] "A"  "D"  "B"
[3,] "B"  "C"  "E"
[4,] "D"  "F"  "C"
[5,] "A"  "D"  "E"
[6,] "B"  "E"  "F"

> plan5$design[[2]]
     [,1] [,2] [,3]
[1,] "D"  "E"  "C"
[2,] "E"  "F"  "D"
[3,] "B"  "C"  "D"
[4,] "A"  "F"  "E"
[5,] "C"  "B"  "A"
[6,] "B"  "F"  "A"
```

Se han generado 12 bloques de 4 tratamientos cada uno.

3.7 LATICE

Se requiere un número de tratamientos de un cuadrado perfecto, por ejemplo 9, 16, 25, 36, 49, etc.

Puede generar simple (2 rep.) o triple (3 rep.)

Generando triple para 9 tratamientos 3x3.

```
plan6 <- design.lattice(k=3, seed=55, number=101)
plan6
```

```
$square1
     [,1] [,2] [,3]
[1,]    1    4    8
[2,]    2    9    3
[3,]    5    7    6
$square2
     [,1] [,2] [,3]
[1,]    2    1    5
[2,]    9    4    7
[3,]    3    8    6

$square3
     [,1] [,2] [,3]
[1,]    2    4    6
[2,]    3    1    7
[3,]    9    8    5

$plan
   plots sqr block trt
1    101   1     1   1
2    102   1     1   4
...
27   127   3     9   5
```

3.8 ALFA

Son los diseños generados por los arreglos alpha de Patterson & Williams. Son similares a los diseños latices, pero los cuadros son rectangulares de s bloques por k tratamientos por bloque. El número de tratamientos debe ser igual a s*k y el total de unidades experimentales r*s*k.

```
plan7 <- design.alpha(1:15,k=3,r=2,seed=55)

alpha design (0,1) - Serie  I

Parameters Alpha design
=======================
treatmeans : 15
Block size : 3
Blocks     : 5
Replication: 2

Efficiency factor
(E ) 0.6363636

<<< Book >>>
plan7$design
$design$rep1
     [,1] [,2] [,3]
[1,] "8"  "4"  "10"
[2,] "1"  "12" "14"
[3,] "6"  "2"  "15"
[4,] "7"  "3"  "11"
[5,] "13" "9"  "5"
```

```
$design$rep2
     [,1] [,2] [,3]
[1,] "13" "7"  "1"
[2,] "8"  "5"  "14"
[3,] "4"  "11" "15"
[4,] "6"  "3"  "12"
[5,] "10" "2"  "9"
```

4 ANÁLISIS DE DISEÑOS

Las funciones de agricolae son: BIB.test(), PBIB.test(), LSD.test(), HSD.test(), durbin.test(), kruskal(), friedman() y waerden.test.

Para todo análisis estadístico, los datos deben estar organizados por columna. Para la demostración se utilizará la base de datos de "agricolae".

Los datos de "sweetpotato" corresponden a un experimento completamente aleatorio en campo con parcelas de 50 plantas de camote y sometidas al efecto de virus y un control sin virus (ver manual de referencia de la librería).

```
data(sweetpotato)
modelo<-aov(yield~virus, data=sweetpotato)
cv.model(modelo)
[1] 17.16660

attach(sweetpotato)
mean(yield)
[1] 27.625
df<-df.residual(model)
MSerror<-deviance(model)/df
```

4.1 LSD

```
compara <- LSD.test(yield, virus,df,MSerror)

Study:
LSD t Test for yield

                          ......
Alpha                    0.050000
Error Degrees of Freedom 8.000000
Error Mean Square       22.489167
Critical Value of t      2.306004

Treatment Means
  virus    yield  std.err replication
1    cc 24.40000 2.084067           3
2    fc 12.86667 1.246774           3
3    ff 36.33333 4.233727           3
4    oo 36.90000 2.482606           3

Least Significant Difference 8.928965
Means with the same letter are not significantly different.
```

```
Groups, Treatments and means
a         oo       36.9
a         ff       36.33333
 b        cc       24.4
  c       fc       12.86667
```

En la función LSD.test() se realizó la comparación múltiple. Para obtener las probabilidades de las comparaciones se debe indicar que no se requiere grupos, así:

```
compara <- LSD.test(yield, virus,df, MSerror, group=F)

Treatment Means
  virus    yield  std.err replication
1    cc 24.40000 2.084067           3
2    fc 12.86667 1.246774           3
3    ff 36.33333 4.233727           3
4    oo 36.90000 2.482606           3

Comparison between treatments means

  tr.i tr.j       diff pvalue
1    1    2 11.5333333 0.0176
2    1    3 11.9333333 0.0151
3    1    4 12.5000000 0.0121
4    2    3 23.4666667 0.0003
5    2    4 24.0333333 0.0003
6    3    4  0.5666667 0.8873
```

Aquí, "cc" es el tratamiento 1 y "fc", el tratamiento 2; por lo tanto, la comparación de ambos es 1 vs 2 y el p.valor es 0.0176, indicando que son diferentes significativamente y así sucesivamente.

4.2 BONFERRONI

Mediante LSD.test se puede hacer ajustes a las probabilidades encontradas como bonferroni

```
compara <- LSD.test(yield, virus,df, MSerror, group=F, p.adj= "bon")

tr.i tr.j       diff pvalue
   1    2 11.5333333 0.1058
   1    3 11.9333333 0.0904
   1    4 12.5000000 0.0725
   2    3 23.4666667 0.0018
   2    4 24.0333333 0.0015
   3    4  0.5666667 1.0000
```

Otras pruebas de comparación pueden ser aplicadas, tales como "tukey", "waller-duncan".

Para "tukey", la función HSD.test(), y para "waller-duncan" waller.test(). Los parámetros son los mismos; en el caso de Waller, requiere además el valor de F-calculado del ANVA.

4.3 TUKEY

```
compara1 <- HSD.test(yield, virus,df, MSerror)

HSD Test for yield

                                        ......
Alpha                                  0.05000
Error Degrees of Freedom               8.00000
Error Mean Square                     22.48917
Critical Value of Studentized Range    4.52881

Treatment Means
  virus    yield  std.err replication
1    cc 24.40000 2.084067           3
2    fc 12.86667 1.246774           3
3    ff 36.33333 4.233727           3
4    oo 36.90000 2.482606           3

Honestly Significant Difference 12.39967
Means with the same letter are not significantly different.

Groups, Treatments and means
a         oo       36.9
ab        ff       36.33333
 bc       cc       24.4
  c       fc       12.86667
```

4.4 WALLER DUNCAN

```
# análisis de variancia:

anova(modelo)

Analysis of Variance Table

Response: yield
          Df  Sum Sq Mean Sq F value    Pr(>F)
virus      3 1170.21  390.07  17.345 0.0007334 ***
Residuals  8  179.91   22.49
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

El valor de F calculado es 17.345, entonces:

```
compara2 <- waller.test(yield, virus,df, MSerror,Fc= 17.345,
group=F)

Waller-Duncan K-ratio t Test for yield

This test minimizes the Bayes risk under additive
loss and certain other assumptions.

                              ......
K ratio                   100.00000
Error Degrees of Freedom    8.00000
Error Mean Square          22.48917
F value                    17.34500
Critical Value of Waller    2.23600
```

```
Treatment Means
  virus    yield  std.err replication
1    cc 24.40000 2.084067           3
2    fc 12.86667 1.246774           3
3    ff 36.33333 4.233727           3
4    oo 36.90000 2.482606           3

Minimum Significant Difference 8.657906

Comparison between treatments means

  tr.i tr.j       diff significant
1    1    2 11.5333333        TRUE
2    1    3 11.9333333        TRUE
3    1    4 12.5000000        TRUE
4    2    3 23.4666667        TRUE
5    2    4 24.0333333        TRUE
6    3    4  0.5666667       FALSE
```

Se indica que el efecto del virus "ff" no es significativamente del control "oo".

El objeto encontrado "compara" y tiene información para hacer otros procedimientos.

```
compara2
  trt    means M N  std.err
1  cc 24.40000   3 2.084067
2  fc 12.86667   3 1.246774
3  ff 36.33333   3 4.233727
4  oo 36.90000   3 2.482606
```

En el caso de la prueba "tukey" la columna M contienen las letras del agrupamiento de la comparación de tratamientos:

```
compara1
  trt    means   M N  std.err
1  oo 36.90000   a 3 2.482606
2  ff 36.33333  ab 3 4.233727
3  cc 24.40000  bc 3 2.084067
4  fc 12.86667   c 3 1.246774
```

4.5 GRÁFICOS DE LA COMPARACIÓN

Ambos objetos pueden ser utilizados para obtener graficos de barra, Figura 4.5.

compara1, para las funciones bar.group() y bar.err()
compara2, para la función bar.err()

```
par(mfrow=c(2,2))
c1<-colors()[480]; c2=colors()[65]; c3=colors()[15];
c4=colors()[140]
G1<-bar.group(compara1, ylim=c(0,45), main="Tukey\nG1",col=c1)
G2<-bar.group(compara1, horiz=T, xlim=c(0,45),
main="Tukey\nG2",col=c2)
G3<-bar.err(compara2, ylim=c(0,45), col=c3, main="Desviación
estandar\nG3")
G4<-bar.err(compara2, horiz=T, xlim=c(0,45), col=c4,
std=F,main="Error estandar\nG4")
```

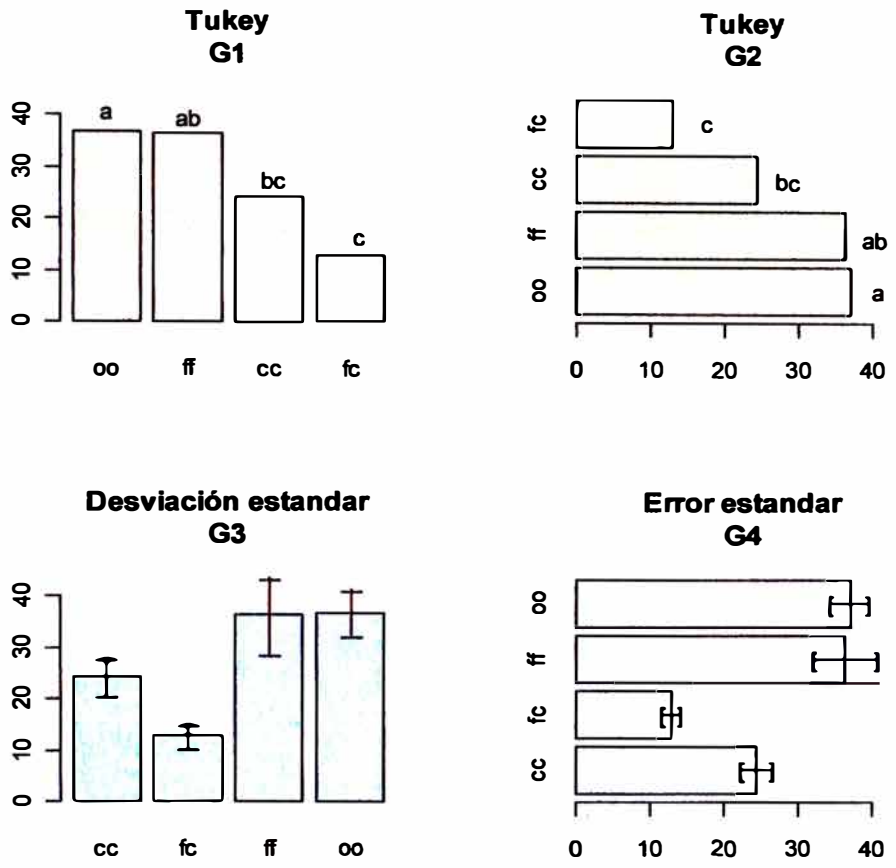


Figura 4.5 Comparacion entre tratamientos.

4.6 ANÁLISIS DE BLOQUES INCOMPLETOS BALANCEADO

Éstos pueden provenir de diseños balanceados o parcialmente balanceados. La función BIB.test() es para diseños balanceados, y PBIB.test(), para diseños parcialmente balanceados. Para el ejemplo, se utilizarán los datos de "agricolae".

```
#Example linear estimation and design of experiments. D.D.Joshi.
1987
# Profesor de Estatistica, Institute of Social Sciences Agra, India
# 6 variedades de trigo en 10 bloques de 3 parcelas cada una.
bloque<-gl(10,3)
variedad<-c(1,2,3,1,2,4,1,3,5,1,4,6,1,5,6,2,3,6,2,4,5,2,5,6,3,4,5,3,
4,6)
y<-c(69,54,50,77,65,38,72,45,54,63,60,39,70,65,54,65,68,67,57,60,62,
59,65,63,75,62,61,59,55,56)
modelo<-BIB.test(block=bloque, trt=variedad, y)

ANALYSIS BIB:  y
Class level information

Block:  1 2 3 4 5 6 7 8 9 10
Trt  :  1 2 3 4 5 6

Number of observations:  30
```

```
Analysis of Variance Table

Response: y
             Df  Sum Sq Mean Sq F value  Pr(>F)
block.unadj   9  466.97   51.89  0.9019 0.54712
trt.adj       5 1156.44  231.29  4.0206 0.01629 *
Residuals    15  862.89   57.53
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

coefficient of variation: 12.6 %
y Means: 60.3

Treatments
  trt means mean.adj StdError.adj
1   1  70.2 75.13333     3.728552
2   2  60.0 58.71667     3.728552
3   3  59.4 58.55000     3.728552
4   4  55.0 54.96667     3.728552
5   5  61.4 60.05000     3.728552
6   6  55.8 54.38333     3.728552

LSD test
Std.diff   : 5.363111
Alpha      : 0.05
LSD        : 11.43120

Parameters BIB
Lambda     : 2
treatmeans : 6
Block size : 3
Blocks     : 10
Replication: 5

Efficiency factor 0.8

<<< Book >>>

Means with the same letter are not significantly different.

Comparison of treatments

Groups, Treatments and means
a        1       75.13333
 b       5       60.05
 b       2       58.71667
 b       3       58.55
 b       4       54.96667
 b       6       54.38333
```

Se puede utilizar Tukey y Waller-Duncan. También se puede obtener las probabilidades de la comparación. Sólo se debe indicar group=FALSE, así:

```
modelo<-BIB.test(block=bloque, trt=variedad, y, group=F, method=
"tukey")
```

```
Comparison between treatments means
   tr.i tr.j       diff pvalue
1     1    2 16.4166667 0.0705
2     1    3 16.5833333 0.0666
3     1    4 20.1666667 0.0191
...
14    4    6  0.5833333 1.0000
15    5    6  5.6666667 0.8908
```

El objeto "modelo" encontrado puede ser utilizado para las funciones de bar.group(), y bar.err() para las gráficos de barras, en la misma forma como se realizo anteriormente.

4.7 ANÁLISIS DE BLOQUES INCOMPLETOS PARCIALMENTE BALANCEADO

La función PBIB.test() puede ser utilizada para los diseños latice y alfa.

Considere el caso siguiente: Construir el diseño alfa con 30 tratamientos, 2 repeticiones y tamaño del bloque igual a 3.

```
library(agricolae)
library(corpcor)
# alpha design
trt<-1:30
r<-2
k<-3
plan<-design.alpha(trt,k,r,seed=5)

alpha design (0,1) - Serie  I

Parameters Alpha design
=======================
treatmeans : 30
Block size : 3
Blocks     : 10
Replication: 2

Efficiency factor
(E ) 0.6170213

<<< Book >>>
```

El Plan generado es plan$book

Suponga que la observación correspondiente a cada unidad experimental es:

```
y<-c(5,2,7,6,4,9,7,6,7,9,6,2,1,1,3,2,4,6,7,9,8,7,6,4,3,2,2,1,1,2,
     1,1,2,4,5,6,7,8,6,5,4,3,1,1,2,5,4,2,7,6,6,5,6,4,5,7,6,5,5,4)
```

Se construye la tabla de datos para el análisis. En teoría, se supone que uno aplica un diseño y realiza el experimento; posteriormente observa a partir de cada unidad experimental las variables de estudio.

```
tabla<-data.frame(plan$book,rdto=y)
rm(y,trt)
```

El análisis:

```
attach(tabla)
modelo <- PBIB.test(block, trt, replication, rdto, k=3)
detach(tabla)

ANALYSIS PBIB:  rdto
Class level information

Blocks:  20
Trts  :  30

Number of observations:  60

Analysis of Variance Table

Response: rdto
                       Df  Sum Sq Mean Sq F value  Pr(>F)
replication             1   0.600   0.600  0.2931 0.59901
trt.unadj              29 188.933   6.515  3.1831 0.02334 *
replication:block.adj  18 112.886   6.271  3.0641 0.03125 *
Residuals              11  22.514   2.047
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

coefficient of variation: 31.6 %
rdto Means: 4.533333

Treatments

Parameters PBIB
treatmeans : 30
Block size : 3
Blocks/rep : 10
Replication: 2

Efficiency factor 0.6170213

Comparison between treatments means

<<< to see the objects: comparison and means  >>>
```

Las medias ajustadas pueden ser extraídas del modelo

```
modelo$means

   trt means mean.adj N  std.err
1    1   7.5 6.514615 2 1.323025
2    2   4.5 2.862294 2 1.320192
...
29  29   3.0 5.052705 2 1.323025
30  30   3.5 6.028477 2 1.320192
```

Las comparaciones

```
modelo$comparison
```

```
    tr.i tr.j       diff   stderr pvalue
1      1    2 3.65232195 1.785288 0.0654
2      1    3 0.78397461 1.785288 0.6690
3      1    4 2.53471831 1.831020 0.1936
...
434   28   30 0.86957356 1.860419 0.6494
435   29   30 0.97577182 1.591954 0.5524
```

Los datos sobre las medias ajustadas y su error estándar pueden ser graficados, figura 4.7, dado que el objeto creado es muy similar a los objetos generados por las comparaciones múltiples.

```
par(mfrow=c(2,2),cex=0.6)
C1<-bar.err(modelo$means[1:7, c(1,3,2,4,5)], ylim=c(0,9), col=0,
main="C1", std=F)
C2<-bar.err(modelo$means[8:15, c(1,3,2,4,5)], ylim=c(0,9), col=0,
main="C2", std=F)
C3<-bar.err(modelo$means[16:22, c(1,3,2,4,5)], ylim=c(0,9), col=0,
main="C3", std=F)
C4<-bar.err(modelo$means[23:30, c(1,3,2,4,5)], ylim=c(0,9), col=0,
main="C4", std=F)
```

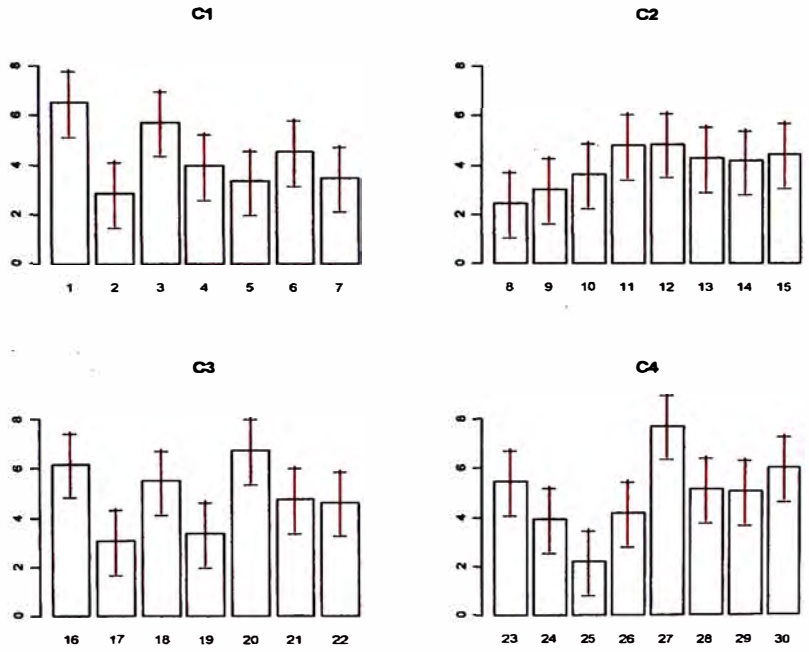


Figura 4.7. Desviación estándar en cada tratamiento.

Análisis de lattice balanceado 3x3, 9 tratamientos, 4 repeticiones.

Crear los datos de la tabla x.x en un archivo de texto: latice3x3.txt y leer con R:

```
A<-read.table("latice3x3.txt", header=T)
attach(A)
```

sqr	block	trt	yield								
1	1	1	48.76	1	1	4	14.46	1	1	3	19.68
1	2	8	10.83	1	2	6	30.69	1	2	7	31.00
1	3	5	12.54	1	3	9	42.01	1	3	2	23.00
2	4	5	11.07	2	4	8	22.00	2	4	1	41.00
2	5	2	22.00	2	5	7	42.80	2	5	3	12.90
2	6	9	47.43	2	6	6	28.28	2	6	4	49.95
3	7	2	27.67	3	7	1	50.00	3	7	6	25.00
3	8	7	30.00	3	8	5	24.00	3	8	4	45.57
3	9	3	13.78	3	9	8	24.00	3	9	9	30.00
4	10	6	37.00	4	10	3	15.42	4	10	5	20.00
4	11	4	42.37	4	11	2	30.00	4	11	8	18.00
4	12	9	39.00	4	12	7	23.80	4	12	1	43.81

```
modelo2<-PBIB.test(block,trt,sqr,yield,k=3)

ANALYSIS PBIB:  yield
Class level information

Blocks:  12
Trts  :  9

Number of observations:  36

Analysis of Variance Table

Response: yield
                      Df Sum Sq Mean Sq F value  Pr(>F)
replication            3  133.2    44.4  0.6859 0.57361
trt.unadj              8 3749.4   468.7  7.2390 0.00042 ***
replication:block.adj  8  368.2    46.0  0.7108 0.67917
Residuals             16 1035.9    64.7
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

coefficient of variation: 27.6 %
yield Means: 29.16167

Treatments

Parameters PBIB
treatmeans : 9
Block size : 3
Blocks/rep : 3
Replication: 4

Efficiency factor 0.75

Comparison between treatments means

<<< to see the objects: comparison and means  >>>

detach(A)
modelo2$means
```

```
  trt   means mean.adj N  std.err
1   1 45.8925 44.36090 4 3.641342
2   2 25.6675 26.06676 4 3.641342
3   3 15.4450 15.08383 4 3.641342
4   4 38.0875 39.07214 4 3.641342
5   5 16.9025 17.37476 4 3.641342
6   6 30.2425 31.15052 4 3.641342
7   7 31.9000 31.82067 4 3.641342
8   8 18.7075 18.08200 4 3.641342
9   9 39.6100 39.44343 4 3.641342
```

```
modelo2$comparison
```

```
   tr.i tr.j       diff   stderr pvalue
1     1    2 18.2941444 5.289807 0.0032
2     1    3 29.2770756 5.289807 0.0000
3     1    4  5.2887627 5.289807 0.3322
...
35    7    9  7.6227578 5.289807 0.1688
36    8    9 21.3614257 5.289807 0.0010
```

4. 8 COMPARACIONES NO-PARAMÉTRICAS

Las funciones correspondientes son: kruskal(), waerden.test(), friedman() y durbin.test().

Kruskal() para muestras de N poblaciones o datos provenientes de un experimento completamente aleatorio.

waerden.test(), similar a kruskal-wallis, utiliza score normal en vez de rangos como kruskal.

Friedman() para evaluación de jueces, para un número completo de tratamientos o el diseño de bloques completos al azar.

durbin.test() para bloques incompletos, muy utilizado para pruebas de degustación, donde los jueces evaluan un numero pequeño de tratamientos provenientes de un numero grande.

Datos del libro de montgomery: corn
Incluidos en la librería agricolae

```
library(SuppDists)
library(agricolae)
data(corn)
attach(corn)
str(corn)
'data.frame':   34 obs. of  3 variables:
 $ method     : int  1 1 1 1 1 1 1 1 1 2 ...
 $ observation: int  83 91 94 89 89 96 91 92 90 91 ...
 $ rx         : num  11 23 28.5 17 17 31.5 23 26 19.5 23 ...
```

4.9 KRUSKAL - WALLIS

```
compara<-kruskal(observation,method,group=TRUE, main="corn")

Study: corn
Kruskal-Wallis test's
Value: 25.62884
degrees of freedom: 3
Pvalue chisq  : 1.140573e-05
pKruskalWallis: 1.279183e-09

Mean of the ranks
  method observation replication
1      1    21.83333           9
2      2    15.30000          10
3      3    29.57143           7
4      4     4.81250           8

t-Student: 2.042272
Alpha    : 0.05
LSD      : 4.9175

Harmonic Mean of Cell Sizes  8.351284

Means with the same letter are not significantly different

Groups, Treatments and mean of the ranks
a         3        29.57143
 b        1        21.83333
  c       2        15.3
   d      4        4.8125
```

El objeto "compara" tiene la misma estructura de las comparaciones (figura 4.9).

```
par(mfrow=c(1,2),cex=0.8,mar=c(3,3,1,0))
bar.group(compara,ylim=c(0,35),col=colors()[45])
bar.err(compara,ylim=c(0,35),col=colors()[25],std=F, main="Std.Err")
```

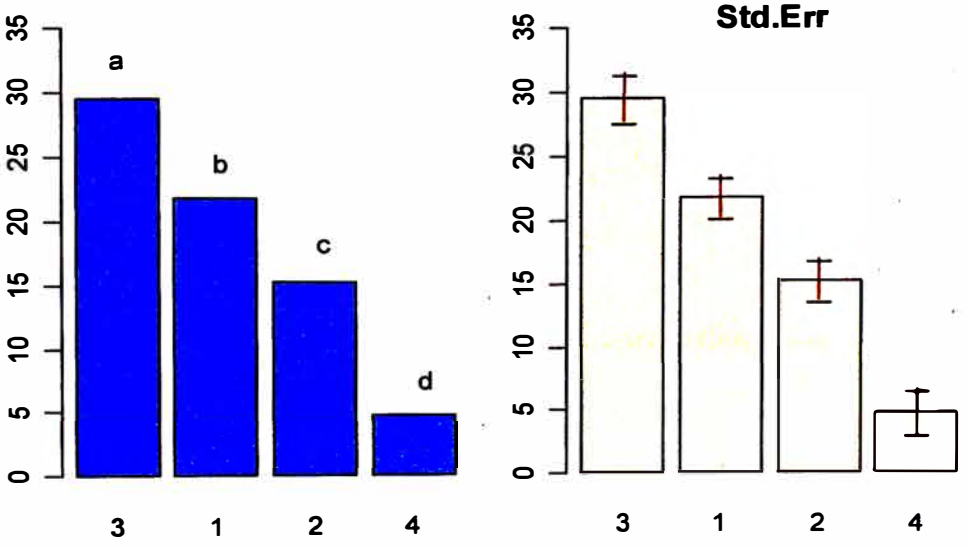


Figura 4.9. Comparación según Waller-Duncan.

4.10 FRIEDMAN

```
friedman()

library(SuppDists)
library(agricolae)
data(grass)
attach(grass)
compara<-friedman(judge,trt, evaluation,alpha=0.05, group=FALSE,
main="Data of the book of Conover")

Study: Data of the book of Conover

Sum of the ranks
  trt evaluation replication
1  t1       38.0          12
2  t2       23.5          12
3  t3       24.5          12
4  t4       34.0          12

Friedman's Test
===============
Chi-squard: 8.097345
Pvalue    : 0.04404214
pFriedman : 0.03939295
Alpha     : 0.05
t-Student : 2.034515

Comparison between treatments
Sum of the ranks

  tr.i tr.j diff pvalue signf   LSD
1    1    2 14.5 0.0148     * 11.48
2    1    3 13.5 0.0226     * 11.48
3    1    4  4.0 0.4834    ns 11.48
4    2    3  1.0 0.8604    ns 11.48
5    2    4 10.5 0.0718    ns 11.48
6    3    4  9.5 0.1018    ns 11.48
```

4.11 WAERDEN

Waerden.test()

Con datos de camote en la base agricolae

```
data(sweetpotato)
attach(sweetpotato)
compara<-waerden.test(yield,virus,alpha=0.01,group=TRUE)
detach(sweetpotato)

Study:
Van der Waerden (Normal Scores) test's

Value : 8.40998
Pvalue: 0.03825667
Degrees of freedom:  3
```

```
Means of the normal score
  virus      yield replication
1    cc -0.2328353           3
2    fc -1.0601764           3
3    ff  0.6885684           3
4    oo  0.6044433           3

t-Student: 3.355387
Alpha    : 0.01
LSD      : 1.322487

Means with the same letter are not significantly different

Groups, Treatments and means of the normal score
a         ff      0.6885684
a         oo      0.6044433
ab        cc      -0.2328353
 b        fc      -1.060176
```

Las probabilidades de comparacion se obtienen con el parámetro group=FALSE.

```
compara<-waerden.test(yield,virus,group=F)

Comparison between treatments means
mean of the normal score


  tr.i tr.j      diff pvalue
1    1    2 0.8273411 0.0690
2    1    3 0.9214037 0.0476
3    1    4 0.8372786 0.0664
4    2    3 1.7487448 0.0022
5    2    4 1.6646197 0.0030
6    3    4 0.0841251 0.8364
```

4.12 DURBIN

durbin(). Ejemplo: Myles Hollander (pag 311) Fuente: W. Moore and C.I. Bliss. (1942)

```
dias <-gl(7,3)
quimico<-c("A","B","D","A","C","E","C","D","G","A","F","G",
"B","C","F", "B","E","G","D","E","F")
toxico<-c(0.465,0.343,0.396,0.602,0.873,0.634,0.875,0.325,0.330,
0.423,0.987,0.426,0.652,1.142,0.989,0.536,0.409,0.309,
0.609,0.417,0.931)

compara<-durbin.test(dias,quimico,toxico,group=F,
main="Logaritmo de la dosis tóxica")

Study: Logaritmo de la dosis tóxica

Sum of ranks
  quimico toxico
1       A      5
2       B      5
3       C      9
```

```
4       D       5
5       E       5
6       F       8
7       G       5

Durbin Test
===========
Value       : 7.714286
Df 1        : 6
P-value     : 0.2597916
Alpha       : 0.05
Df 2        : 8
t-Student   : 2.306004

Least Significant Difference
between the sum of ranks:  5.00689

Parameters BIB
Lambda      : 1
treatmeans  : 7
Block size  : 3
Blocks      : 7
Replication: 3

Comparison between treatments sum of the ranks

   tr.i tr.j diff pvalue signf
1     1    2    0 1.0000    ns
2     1    3    4 0.1026    ns
...
21    6    7    3 0.2044    ns
```

5 ANÁLISIS DE ESTABILIDAD

En "agricolae" se tiene dos métodos para el estudio de estabilidad y el modelo AMMI. Éstos son: un modelo paramétrico para una selección simultanea en rendimiento y estabilidad "SHUKLA'S STABILITY VARIANCE AND KANG'S", y un método no paramétrico de Haynes, basado en el rango de los datos.

5.1 ESTABILIDAD PARAMETRICA

Uso del modelo paramétrico función stability.par()

Preparar un tabla de datos, donde las filas sean los genotipos y las columnas, los ambientes. Los datos deben corresponder a promedios de rendimiento u otra variable medida. Determinar la variancia del error común para todos los ambientes y el número de repeticiones que fue evaluado cada genotipo. Si las repeticiones son diferentes, hallar un promedio armónico que representará al conjunto. Finalmente asigne un nombre a cada fila que representará al genotipo. Para el ejemplo se considera 5 ambientes:

```
v1 <- c(10.2, 8.8, 8.8, 9.3, 9.6, 7.2, 8.4, 9.6, 7.9, 10, 9.3, 8.0, 10.1, 9.4, 10.8, 6.3, 7.4)
v2 <- c(7, 7.8, 7.0, 6.9, 7, 8.3, 7.4, 6.5, 6.8, 7.9, 7.3, 6.8, 8.1, 7.1, 7.1, 6.4, 4.1)
v3 <- c(5.3, 4.4, 5.3, 4.4, 5.5, 4.6, 6.2, 6.0, 6.5, 5.3, 5.7, 4.4, 4.2, 5.6, 5.8, 3.9, 3.8)
v4 <- c(7.8, 5.9, 7.3, 5.9, 7.8, 6.3, 7.9, 7.5, 7.6, 5.4, 5.6, 7.8, 6.5, 8.1, 7.5, 5.0, 5.4)
v5 <- c(9, 9.2, 8.8, 10.6, 8.3, 9.3, 9.6, 8.8, 7.9, 9.1, 7.7, 9.5, 9.4, 9.4, 10.3, 8.8, 8.7)
```

Para 17 genotipos, se identifica por letras.

```
estudio <- data.frame(v1, v2, v3, v4, v5)
rownames(estudio) <- LETTERS[1:17]
```

Se asume una variancia del error de 2 y 4 repeticiones.

```
estabilidad <- stability.par(estudio, rep=4, MSerror=2)

INTERACTIVE PROGRAM FOR CALCULATING SHUKLA'S STABILITY VARIANCE AND
KANG'S
                        YIELD - STABILITY (YSi) STATISTICS

Environmental index  - covariate

Analysis of Variance
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Source            d.f.    Sum of Squares   Mean Squares           F
p.value
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
TOTAL             84     1035.6075
GENOTYPES         16      120.0875          7.5055          2.65  0.003
ENVIRONMENTS       4      734.2475        183.5619         91.78 <0.001
INTERACTION       64      181.2725          2.8324          1.42  0.033
HETEROGENEITY     16       52.7128          3.2945          1.23  0.281
RESIDUAL          48      128.5597          2.6783          1.34  0.081
POOLED ERROR     240                        2.0000
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Stability statistics
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
Genotype MEANS   Sigma-square     s-square       Ecovalence
 - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -
 1       7.86    1.671833 ns     2.209084 ns      6.567031
 2       7.22    1.822233 ns     1.977299 ns      7.097855
 3       7.44    0.233967 ns     0.134103 ns      1.492208
 4       7.42    4.079567 ns     1.443859 ns     15.064913
 5       7.64    2.037967 ns     2.369090 ns      7.859266
 6       7.14    5.161967 *      6.763106 *      18.885149
 7       7.90    1.759300 ns     1.058092 ns      6.875737
 8       7.68    1.757167 ns     2.028880 ns      6.868208
 9       7.34    5.495300 *      0.423680 ns     20.061619
 10      7.54    4.129967 ns     5.125514 ns     15.242796
 11      7.12    3.848900 ns     4.360772 ns     14.250796
 12      7.30    2.675300 ns     3.610982 ns     10.108678
 13      7.66    3.473167 ns     2.198229 ns     12.924678
 14      7.92    0.806233 ns     1.097156 ns      3.511972
 15      8.30    1.951300 ns     1.459578 ns      7.553384
 16      6.08    3.647833 ns     4.919102 ns     13.541149
 17      5.88    3.598500 ns     4.353030 ns     13.367031

Signif. codes:  0 '**' 0.01 '*' 0.05 'ns' 1

Simultaneous selection for yield and stability  (++)
```

```
 Genotype Yield Rank Adj.rank Adjusted Stab.var Stab.rating YSi ...
1        A  7.86   14        1       15 1.671833           0  15   +
2        B  7.22    5       -1        4 1.822233           0   4
3        C  7.44    9        1       10 0.233967           0  10   +
4        D  7.42    8        1        9 4.079567          -2   7
5        E  7.64   11        1       12 2.037967           0  12   +
6        F  7.14    4       -1        3 5.161967          -4  -1
7        G  7.90   15        1       16 1.759300           0  16   +
8        H  7.68   13        1       14 1.757167           0  14   +
9        I  7.34    7       -1        6 5.495300          -4   2
10       J  7.54   10        1       11 4.129967          -2   9   +
11       K  7.12    3       -1        2 3.848900           0   2
12       L  7.30    6       -1        5 2.675300           0   5
13       M  7.66   12        1       13 3.473167           0  13   +
14       N  7.92   16        1       17 0.806233           0  17   +
15       O  8.30   17        2       19 1.951300           0  19   +
16       P  6.08    2       -2        0 3.647833           0   0
17       Q  5.88    1       -3       -2 3.598500           0  -2

Yield Mean: 7.378824
 YS    Mean: 8.352941
 LSD (0.05): 0.7384513
 - - - - - - - - - - -
 +   selected genotype
 ++  Reference: Kang, M. S. 1993. Simultaneous selection for yield
 and stability: Consequences for growers. Agron. J. 85:754-757.
```

Los genotipos seleccionados son: A, C, E, G, H, J, M, N y O. Estos genotipos tienen un rendimiento más alto y una variación más baja. Según el ANVA, la interacción es significativa.
Si se tiene un índice ambiental, se puede agregar al modelo como una covariable. Así por ejemplo la altitud de las localidades.

```
altitud<-c(1200, 1300, 800, 1600, 2400)
estabilidad <- stability.par(estudio,rep=4,MSerror=2,cova=TRUE,
cova=altitud)
```

5.2 ESTABILIDAD NO PARAMÉTRICA

Para estabilidad no-paramétrica, la función es stability.nonpar().
Esta función requiere que la primera columna contenga los nombres de los genotipos y las otras columnas de los ambientes.

```
datos <- data.frame(nombre=row.names(estudio), estudio)
modelo<-stability.nonpar(datos, "YIELD", ranking=TRUE)

Nonparametric Method for Stability Analysis
-------------------------------------------

Estimation and test of nonparametric measures
Variable: YIELD

Ranking...
    v1   v2 v3   v4   v5
A 16.0  8.0  9 14.0  8.0
B  7.5 14.0  5  5.5 10.0
C  7.5  8.0  9  9.0  6.0
```

```
D  9.5  6.0  5  5.5 17.0
E 12.5  8.0 11 14.0  3.0
F  2.0 17.0  7  7.0 11.0
G  6.0 13.0 16 16.0 15.0
H 12.5  3.0 15 10.5  6.0
I  4.0  4.5 17 12.0  2.0
J 14.0 15.0  9  2.5  9.0
K  9.5 12.0 13  4.0  1.0
L  5.0  4.5  5 14.0 14.0
M 15.0 16.0  3  8.0 12.5
N 11.0 10.5 12 17.0 12.5
O 17.0 10.5 14 10.5 16.0
P  1.0  2.0  2  1.0  6.0
Q  3.0  1.0  1  2.5  4.0

Statistics...
  Mean Rank  s1   Z1   s2   Z2
A 7.86   14 5.4 0.02 21.5 0.04
B 7.22    5 6.2 0.12 25.7 0.02
C 7.44    9 3.0 2.73  7.5 1.83
D 7.42    8 7.4 1.20 36.5 1.05
E 7.64   11 5.6 0.00 21.8 0.03
F 7.14    4 7.8 1.81 39.2 1.55
G 7.90   15 5.2 0.08 18.7 0.19
H 7.68   13 6.2 0.12 25.3 0.01
I 7.34    7 8.8 3.87 51.5 5.08
J 7.54   10 7.2 0.94 34.3 0.71
K 7.12    3 7.8 1.81 43.0 2.43
L 7.30    6 7.4 1.20 34.7 0.77
M 7.66   12 7.6 1.49 38.2 1.36
N 7.92   16 4.2 0.82 14.8 0.57
O 8.30   17 7.0 0.71 31.7 0.40
P 6.08    2 6.6 0.35 27.7 0.09
Q 5.88    1 7.0 0.71 32.3 0.46
------------------------
Sum of Z1:  17.97158
Sum of Z2:  16.59462
------------------------

Test...
The Z-statistics are measures of stability. The test for the
significance
of the sum of Z1 or Z2 are compared to a Chi-Square value of
chi.sum.
individual Z1 or Z2 are compared to a Chi-square value of chi.ind.

      MEAN      es1 es2      vs1   vs2  chi.ind  chi.sum
1 7.378824 5.647059  24 2.566667 148.8 8.843605 27.58711
---
expectation and variance: es1, es2, vs1, vs2
```

5.3 AMMI

El modelo AMMI utiliza el biplot de las componentes principales generado por los ambientes y genotipos. Si existe interacción genotipo-ambiente y el porcentaje de las dos componentes principales explica más del 50% de la variación total, entonces el biplot es una buena alternativa para estudiar la estabilidad de los genotipos.

Los datos para AMMI deben provenir de experimentos similares conducidos en diferentes ambientes. Se requiere homogeneidad de variancia del error. El análisis se realiza con el combinado de los experimentos.

Los datos pueden estar organizados por columna, así:
Ambiente, genotipo, repetición, variable

Los datos también pueden ser los promedios de los genotipos en cada ambiente, pero es necesario considerar un promedio armónico para las repeticiones y una variancia común del error. Los datos deben estar organizados en columnas, para: ambiente, genotipo, variable

Al ejecutar AMMI, este genera los graficos de BIPLOT, ver figrua 5.3.
Para la aplicación, se considera los datos utilizados:

```
rdto <- c(estudio[,1], estudio[,2], estudio[,3], estudio[,4],
estudio[,5])
ambiente <- gl(5,17)
genotipo <- rep(rownames(estudio),5)

modelo<-AMMI(ENV=ambiente,  GEN=genotipo,  REP=4,  Y=rdto,  MSE=2,
ylim=c(-2,2),xlim=c(-2,2), number=FALSE)

ANALYSIS AMMI:  rdto
Class level information
ENV:  1 2 3 4 5
GEN:  A B C D E F G H I J K L M N O P Q
REP:  4

Number of means:  85

Dependent Variable: rdto

Analysis of variance
           Df   Sum Sq    Mean Sq  F value       Pr(>F)
ENV         4 734.2475 183.561882
REP(ENV)   15
GEN        16 120.0875   7.505471 3.752735 3.406054e-06
ENV:GEN    64 181.2725   2.832382 1.416191 3.279630e-02
Residuals 240 480.0000   2.000000

Coeff var       Mean rdto
19.16584         7.378824

Analysis
    percent acum Df   Sum.Sq  Mean.Sq F.value   Pr.F
PC1    38.0 38.0 19 68.96258 3.629609    1.81 0.0225
PC2    29.8 67.8 17 54.02864 3.178155    1.59 0.0675
PC3    22.5 90.3 15 40.84756 2.723170    1.36 0.1680
PC4     9.6 99.9 13 17.43370 1.341054    0.67 0.7915
PC5     0.0 99.9 11  0.00000 0.000000    0.00 1.0000

modelo<-AMMI(ENV=ambiente,  GEN=genotipo,  REP=4,  Y=rdto,  MSE=2,
graph="triplot",number=F)
```

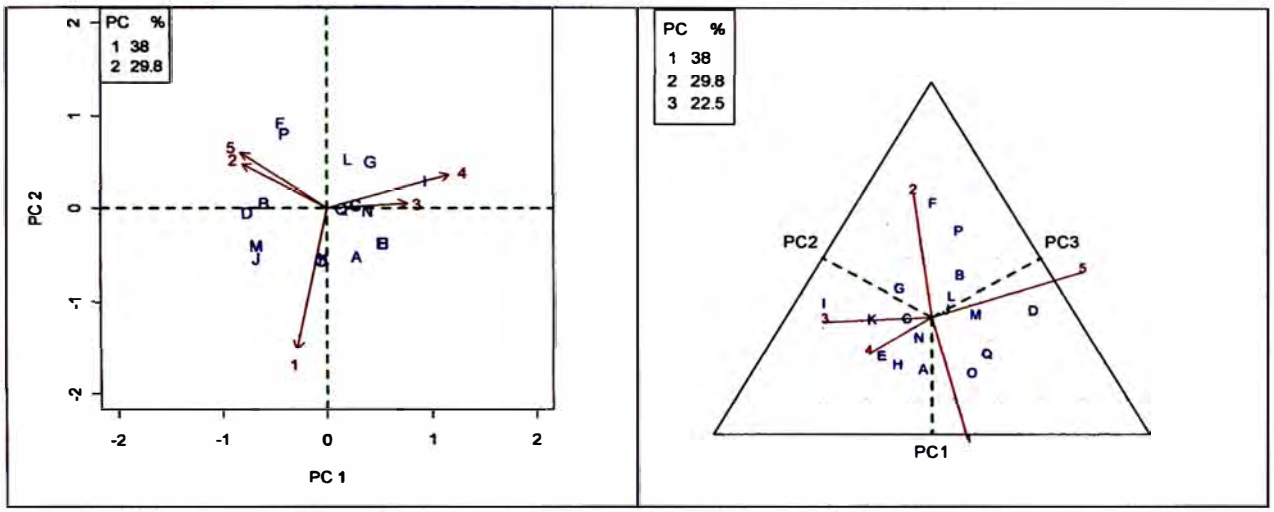


Figura 5.3. Biplot y Triplot

En este caso, la interacción es significativa. Las dos primeras componentes explican el 67.8%; entonces el biplot puede proporcionar información de la relación de genotipos y ambientes. Con el triplot, se explicaría el 90.3%.

6 FUNCIONES ESPECIALES

6.1 CONSENSO DE DENDROGRAMA

El consenso es el grado o semejanza de los vértices de un árbol respecto a sus ramas del dendrograma construido. La funcion a aplicar es consensus().

Los datos deben corresponder a una tabla, con nombre en las filas y en columna las variables. Para el ejemplo, de la base de "agricolae", los datos "pamCIP" corresponden a marcadores moleculares de 43 entradas (filas) y 107 marcadores (columnas).

El programa identifica duplicados en las filas y sólo utiliza 25 filas que no son diferentes. El resultado es un dendrograma, figura 6.1.1.

```
data(pamCIP)
rownames(pamCIP)<-substr(rownames(pamCIP),1,6)
par(cex=0.8)
output<-consensus( pamCIP,distance="binary", method="complete",
nboot=500)
```

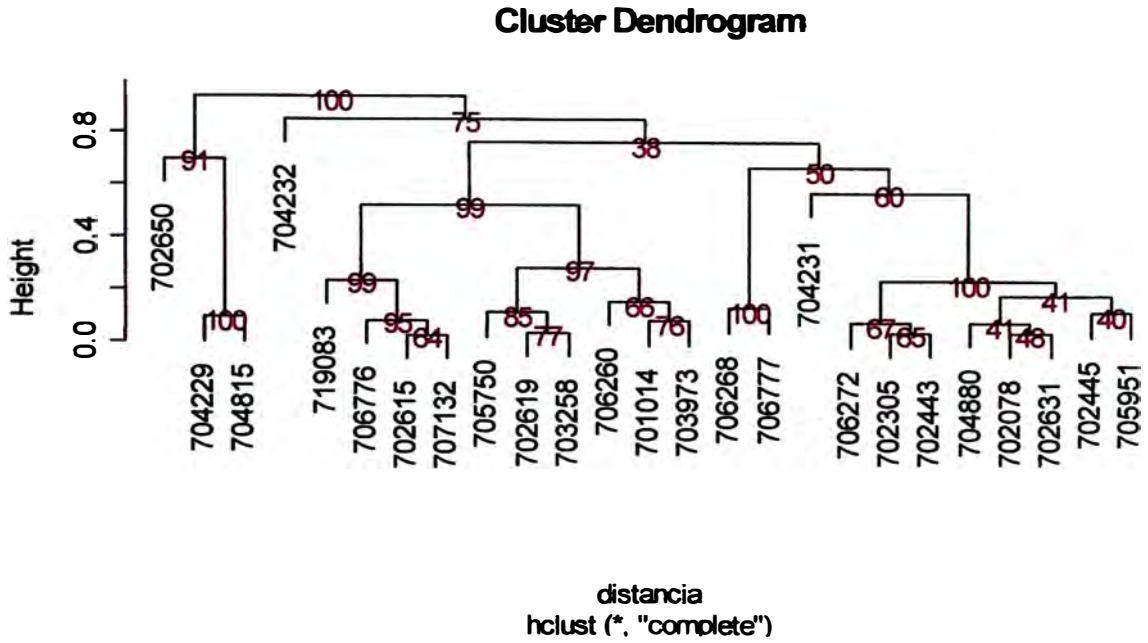


Figura 6.1.1. Dendrograma producion por consensus()

```
Duplicates: 18
New data  : 25 Records

Consensus hclust

Method distance: binary
Method cluster : complete
rows and cols  : 25 107
n-boostrap     : 500
Run time       : 20.469 secs
```

Cuando el dendrograma es complejo, es conveniente extraer parte de éste con la funcion hcut(), figuar 6.1.2.

```
hcut(output,h=0.4,group=8,type="t",edgePar = list(lty=1:2,
col=2:1),main="group 8"
,col.text="blue",cex.text=1)
```

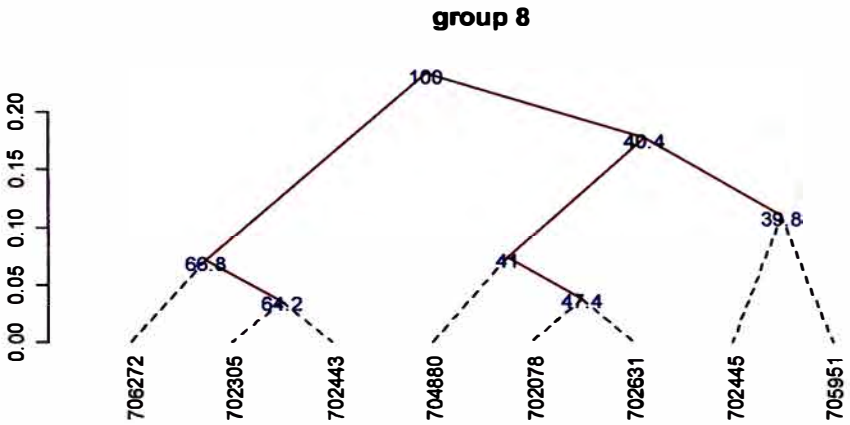


Figura 6.1.2. Dendrograma producion por hcut()

El objeto "output" obtenido contiene información del proceso:

```
names(output)

[1] "table.dend" "dendrogram" "duplicates"
```

Esto significa que se puede conocer los duplicados, volver a construir el diagrama de árbol y tener las relaciones

```
output$ table.dend

    X1  X2     xaxis     height percentage
groups
1   -6 -24  7.500000 0.02857143       64.0
6-24
2   -3  -4 19.500000 0.03571429       64.2
3-4

. . .

24  21  23  5.099609 0.93617021      100.0 1-2-3-4-5-6-7-8-9-10-11-
12-13-14-15-16-17-18-19-20-21-22-23-24-25
```

Reproducir el dendrograma:

```
dend<-output$dendrogram
data<-output$table.dend
plot(dend)
text(data[,3],data[,4],data[,5])
```

Construir un dendrograma clásico, figura 6.1.3

```
dend<-as.dendrogram(output$dendrogram)
plot(dend,type="r",edgePar = list(lty=1:2, col=2:1))
text(data[,3],data[,4],data[,5],col="blue",cex=1)
```

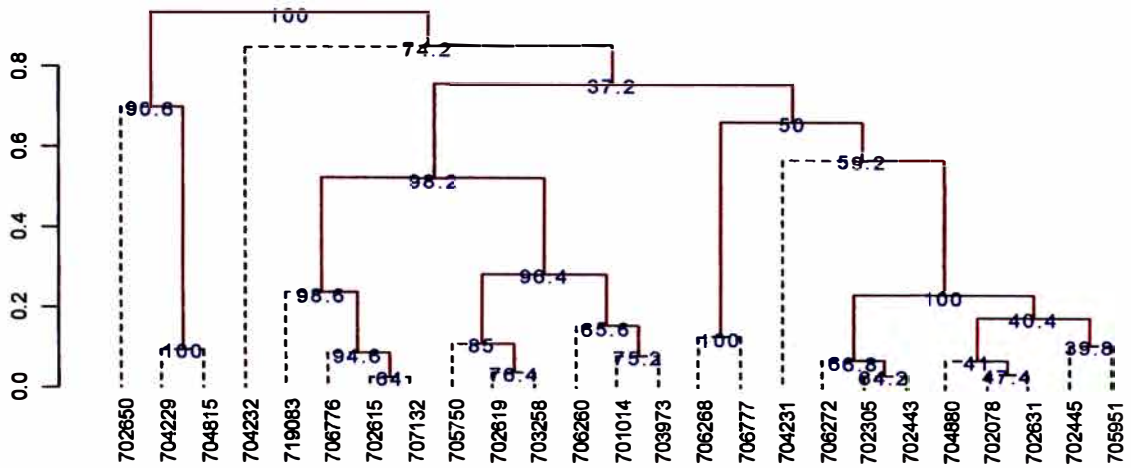


Figura 6.1.3. Dendrograma clásico

6.2 MONTECARLO

Es un método para generar números aleatorios de una distribución desconocida y utilizarse para algun proceso de simulación.

La densidad de probabilidad de los datos riginales y simulados pueden ser comparados, figura 6.2.

```
data(soil)
set.seed(9473)
simulado <- montecarlo(soil$pH,1000)
par(mar=c(3,0,2,1))
plot(density(soil$pH),axes=F,main="Densidad de pH del suelo\ncon
Ralstonia",xlab="",lwd=4)
lines(density(simulado), col="blue", lty=4,lwd=4)
h<-graph.freq(simulado,plot=F)
axis(1,0:12)
legend("topright",c("Original","Simulado"),lty=c(1,4),col=c("black",
"blue"), lwd=4)
```

Figura 6.2. Distribución de los datos simulados y el original

Se han generado 1000 datos y la tabla de frecuencia es:

```
round(table.freq(h),2)

    Inf   Sup    MC  fi  fri  Fi  Fri
   1.60  2.45  2.03   4 0.00   4 0.00
   . . .
   2.45  3.31  2.88  42 0.04  46 0.05
  10.14 10.99 10.57   8 0.01 999 1.00
```

Véanse algunas estadisticas:

```
summary(soil$pH)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  3.800   4.700   6.100   6.154   7.600   8.400

summary(simulado)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
  1.443   4.698   6.022   6.209   7.762  10.950
```

6.3 RE-MUESTREO EN MODELO LINEAL

Utiliza el método de permutación para el cálculo de las probabilidades de las fuentes de variación del ANOVA según el modelo lineal de regresión o diseño.

Los datos deben ser preparados en forma similar para un análisis de variancia. La función que se utilizará es: resampling.model()

```
data(potato)
potato[,1]<-as.factor(potato[,1])
potato[,2]<-as.factor(potato[,2])
model<-"cutting~variety + date + variety:date"
analysis<-resampling.model(1000,potato,model)

Resampling of the experiments
- - - - - - - - - - - - - - - -
Proposed model: cutting~variety + date + variety:date
---
Resampling of the analysis of variancia for the proposed model
Determination of the P-Value by Resampling
Samples: 1000

             Df    Sum Sq   Mean Sq   F value     Pr(>F) Resampling
variety       1 25.086806 25.086806 7.2580377 0.01952218      0.025
date          2 13.891758  6.945879 2.0095604 0.17670768      0.200
variety:date  2  4.853025  2.426513 0.7020312 0.51483592      0.530
Residuals    12 41.477005  3.456417
---
```

La funcion resampling.model() puede ser utilizado cuando los errores no siguen una distribución normal.

6.4 SIMULACIÓN EN MODELO LINEAL

Bajo el supuesto de normalidad, la función genera errores seudo experimentales bajo el modelo propuesto y determina la proporción de resultados válidos según el análisis de variancia encontrado.

La función es: simulation.model(). Los datos son preparados en una tabla, en forma idéntica al de un análisis de variancia.

Para el ejemplo planteado en el procedimiento anterior:

```
model <- simulation.model(1000, potato, model)

Simulation of experiments
Under the normality assumption
- - - - - - - - - - - - - - - - - -
Proposed model: cutting~variety + date + variety:date
Analysis of Variance Table

Response: cutting
             Df Sum Sq Mean Sq F value  Pr(>F)
variety       1 25.087  25.087  7.2580 0.01952 *
date          2 13.892   6.946  2.0096 0.17671
variety:date  2  4.853   2.427  0.7020 0.51484
Residuals    12 41.477   3.456
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
---
Validation of the analysis of variancia for the proposed model
Simulations: 1000

             Df   F value % Acceptance % Rejection  Criterion
variety       1 7.2580377         51.5        48.5 acceptable
date          2 2.0095604         61.1        38.9 acceptable
variety:date  2 0.7020312         67.5        32.5 acceptable
---
```

La validación es referido al porcentaje de resultados de decisión iguales al resultado de decisión del ANOVA. Así, 67.5% de resultados simulados sobre la interaccion variety*date dio el mismo resultado del ANOVA.

6.5 ANALISIS DE CAMINOS

Corresponde al metodo "path análisis", los datos corresponden a matrices de correlacion de las independientes con la dependiente (XY) y entre las independientes (XX)

Es necesario asignar nombres a las filas y columnas para identificar los efectos directos e indirectos.

```
corr.x<- matrix(c(1,0.5,0.5,1),c(2,2))
corr.y<- rbind(0.6,0.7)
names<-c("X1","X2")
dimnames(corr.x)<-list(names,names)
dimnames(corr.y)<-list(names,"Y")
path.analysis(corr.x,corr.y)
```

```
Correlations
============
    X1  X2
X1 1.0 0.5
X2 0.5 1.0

============
     Y
X1 0.6
X2 0.7

Direct(Diagonal) and indirect effect path coefficients
======================================================
          X1        X2
X1 0.3333333 0.2666667
X2 0.1666667 0.5333333

Residual Effect^2 =  0.4266667
```

6.6 LINEA POR PROBADOR

Corresponde a un análisis de cruzas de un diseño genético. Los datos deben estar organizados en una tabla. Sólo se necesitan 4 columnas: Repetición, Hembras, Machos y la respuesta. En el caso que corresponda a progenitores, el campo de Hembras o Machos, sólo estara llenado el que corresponda. Ver los datos heterosis.

```
Ejemplo con los datos de heterosis, localidad 2.

    Replication   Female   Male   v2
109           1     LT-8  TS-15 2.65
110           1     LT-8 TPS-13 2.26
...
131           1 Achirana TPS-13 3.55
132           1 Achirana TPS-67 3.05
133           1     LT-8   <NA> 2.93
134           1    TPS-2   <NA> 2.91
...
140           1 Achirana   <NA> 3.35
141           1     <NA>  TS-15 2.60
142           1     <NA> TPS-13 2.71
143           1     <NA> TPS-67 2.85
145           2     LT-8  TS-15 2.63
...
215           3     <NA> TPS-67 2.91

<NA> es vacío.

Si es una progenie, ésta proviene de un "Female" y "Male".
Si es un progenitor, sólo tendrá "Female" o "Male".

En este ejemplo se tiene datos de la localidad 2, correspondientes
a:

24 progenies (cruzas)
8 hembras
3 machos
3 repeticiones.
```

Son 35 tratamientos (24, 8, 3) aplicados en 3 bloques.

```
data(heterosis)
site2<-subset(heterosis,heterosis[,1]==2)
site2<-subset(site2[,c(2,5,6,8)],site2[,4]!="Control")
attach(site2)
output1<-lineXtester(Replication, Female, Male, v2)
detach(site2)
```

```
ANALYSIS LINE x TESTER:  v2

ANOVA with parents and crosses
==============================
                     Df       Sum Sq     Mean Sq F value Pr(>F)
Replications          2  0.519190476 0.259595238   9.801 0.0002
Treatments           34 16.101605714 0.473576639  17.879 0.0000
Parents              10  7.731490909 0.773149091  29.189 0.0000
Parents vs. Crosses   1  0.005082861 0.005082861   0.192 0.6626
Crosses              23  8.365031944 0.363697041  13.731 0.0000
Error                68  1.801142857 0.026487395
Total               104 18.421939048

ANOVA for line X tester analysis
================================
                Df    Sum Sq    Mean Sq F value Pr(>F)
Lines            7 4.9755431 0.71079187   3.632 0.0191
Testers          2 0.6493861 0.32469306   1.659 0.2256
Lines X Testers 14 2.7401028 0.19572163   7.389 0.0000
Error           68 1.8011429 0.02648739

ANOVA for line X tester analysis including parents
==================================================
                     Df       Sum Sq     Mean Sq F value Pr(>F)
Replications          2  0.519190476 0.259595238   9.801 0.0002
Treatments           34 16.101605714 0.473576639  17.879 0.0000
Parents              10  7.731490909 0.773149091  29.189 0.0000
Parents vs. Crosses   1  0.005082861 0.005082861   0.192 0.6626
Crosses              23  8.365031944 0.363697041  13.731 0.0000
Lines                 7  4.975543056 0.710791865   3.632 0.0191
Testers               2  0.649386111 0.324693056   1.659 0.2256
Lines X Testers      14  2.740102778 0.195721627   7.389 0.0000
Error                68  1.801142857 0.026487395
Total               104 18.421939048

GCA Effects:
============
Lines Effects:
Achirana    LT-8    MF-I    MF-II  Serrana    TPS-2   TPS-25    TPS-7
 0.022  -0.338   0.199  -0.449   0.058  -0.047   0.414   0.141

Testers Effects:
TPS-13 TPS-67  TS-15
 0.087  0.046 -0.132
```

```
SCA Effects:
===========
          Testers
Lines      TPS-13 TPS-67  TS-15
  Achirana  0.061  0.059 -0.120
  LT-8     -0.435  0.519 -0.083
  MF-I     -0.122 -0.065  0.187
  MF-II    -0.194  0.047  0.148
  Serrana   0.032 -0.113  0.081
  TPS-2     0.197 -0.072 -0.124
  TPS-25    0.126 -0.200  0.074
  TPS-7     0.336 -0.173 -0.162

Standard Errors for Combining Ability Effects:
==============================================
S.E. (gca for line)    : 0.05424983
S.E. (gca for tester)  : 0.0332211
S.E. (sca effect)      : 0.09396346
S.E. (gi - gj)line     : 0.07672084
S.E. (gi - gj)tester   : 0.04698173
S.E. (sij - skl)tester: 0.1328844

Genetic Components:
===================
Cov H.S. (line)    : 0.05723003
Cov H.S. (tester)  : 0.00537381
Cov H.S. (average): 0.003867302
Cov F.S. (average): 0.1279716

F = 0, Aditive genetic variance : 0.06187683
F = 1, Aditive genetic variance : 0.01546921
F = 0, Variance due to Dominance: 0.2256456
F = 1, Variance due to Dominance: 0.05641141

Proportional contribution of lines, testers
 and their interactions to total variance
=========================================
Contributions of lines   : 59.48026
Contributions of testers: 7.763104
Contributions of lxt     : 32.75663
```

6.7 UNIFORMIDAD DEL SUELO

El índice de smith es un indicador de la uniformidad, utilizado para determinar el tamaño de la parcela para fines de investigación.
Los datos corresponden a una matriz o tabla que contenga la respuesta por unidad básica, un número de n filas por m columnas, y un total de n*m unidades básicas.

Para la prueba se utilizara el archivo de arroz. El gráfico es un resultado con el ajuste de un modelo para el tamaño de la parcela y el coeficiente de variación, figura 6.7.

```
data(rice)
table<-index.smith(rice,
 main="Relacion entre el CV y el tamaño de la parcela" ,col="red",
type="l",xlab="Tamaño")
uniformidad <- data.frame(table$uniformity)
```

```
uniformidad
   Size Width Length plots       Vx   CV
1     1     1      1   648 9044.539 13.0
2     2     1      2   324 7816.068 12.1
3     2     2      1   324 7831.232 12.1
4     3     1      3   216 7347.975 11.7
5     3     3      1   216 7355.216 11.7
...
39  108     9     12     6 4847.973  9.5
40  162     9     18     4 4009.765  8.6
```

El tamaño es el producto del ancho por el largo de la parcela, y la forma rectangular el tamaño del ancho y el largo.

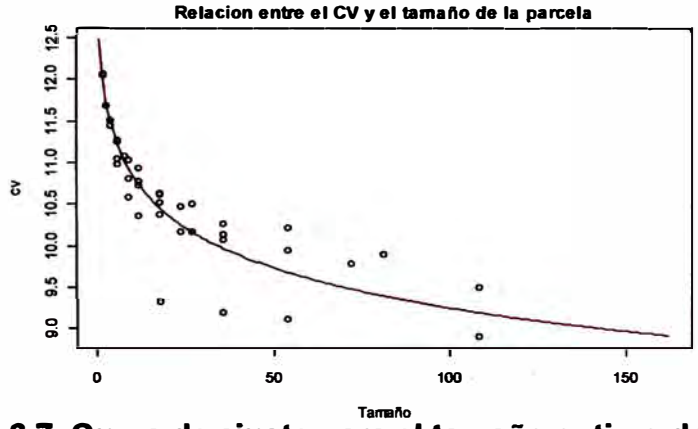


Figura 6.7. Curva de ajuste para el tamaño optimo de parcela

6.8 LÍMITES DE CONFIANZA EN ÍNDICES DE BIODIVERSIDAD

Los intervalos de confianza son determinados por boostrap. Los datos deben estar organizados en una tabla conteniendo en una columna las especies y en otra columna la cantidad. Los índices que se puede calcular con la función index.bio() son: "Margalef", "Simpson.Dom", "Simpson.Div", "Berger.Parker", "McIntosh", y "Shannon".

Para el ejemplo se utilizará los datos de la localidad de Paracsho, distrito de Huasahuasi, provincia de Tarma del departamento de Junin.

La evaluación se realizó el 17 de noviembre de 2005 en las parcelas sin aplicación de insecticidas. Se contabilizó los especímenes.

```
especies <- paracsho[79:87,4:6]
```

```
         Orden        Family Number.of.specimens
79     DIPTERA     TIPULIDAE                 3
80 LEPIDOPTERA     NOCTUIDAE                 1
81   NOCTUIDAE     PYRALIDAE                 3
82   HEMIPTERA  ANTHOCORIDAE                 1
83     DIPTERA    TACHINIDAE                16
84     DIPTERA  ANTHOCORIDAE                 3
85     DIPTERA SCATOPHAGIDAE                 5
86     DIPTERA     SYRPHIDAE                 1
87     DIPTERA      MUSCIDAE                 3
```

El indice de shanon es:

```
output <-
index.bio(especies[,3],method="Shannon",level=95,nboot=200)

Method: Shannon
The index: 3.52304

95 percent confidence interval:
 3.088775 ; 4.286088
```

6.9 CORRELACIÓN

La función correlation() de "agricolae" realiza las correlaciones mediante los métodos de pearson, spearman y kendall para vectores y/o matrices. Si son 2 vectores, realiza la prueba para una o dos colas; si es matricial, determina las probabilidades para una diferencia, mayor o menor.

Para su aplicación, considere los datos de suelo data(soil).

```
data(soil)
correlation(soil[,2:4],method="pearson")

Correlation Analysis

Method     : pearson
Alternative: two.sided

$correlation
        pH   EC CaCO3
pH    1.00 0.55  0.73
EC    0.55 1.00  0.32
CaCO3 0.73 0.32  1.00

$pvalue
                pH        EC       CaCO3
pH    1.000000000 0.0525330 0.004797027
EC    0.052532997 1.0000000 0.294159813
CaCO3 0.004797027 0.2941598 1.000000000

$n.obs
[1] 13

attach(soil)
correlation(pH,soil[,3:4],method="pearson")

Correlation Analysis

Method     : pearson
Alternative: two.sided

$correlation
     EC CaCO3
pH 0.55  0.73
```

```
$pvalue
       EC  CaCO3
pH 0.0525 0.0048

$n.obs
[1] 13
```

```
correlation(pH,CaCO3,method="pearson")
```

```
Pearson's product-moment correlation

data: pH and CaCO3
t = 3.520169 , df = 11 , p-value = 0.004797027
alternative hypothesis: true rho is not equal to 0
sample estimates:
cor
 0.7278362
```

6.10 OTRAS FUNCIONES

Funciones que facilitan el análisis y el manejo de datos:

tapply.stat() Cálculo de estadistas y operaciones matemáticas en columnas de una tabla en función de factores agrupados.

Tabla de factores y variables

Aplicación con datos de "agricolae":

```
data(RioChillon)
attach(RioChillon$babies)
tapply.stat(yield,farmer,function(x) max(x)-min(x))
detach(RioChillon$babies)
```

```
             farmer yield
1  AugustoZambrano   7.5
2         Caballero  13.4
3       ChocasAlto  14.1
4        FelixAndia  19.4
5      Huarangal-1   9.8
6      Huarangal-2   9.1
7      Huarangal-3   9.4
8         Huatocay  19.4
9 IgnacioPolinario  13.1
```

Corresponde al rango de variación en el rendimiento de los agricultores.

La función tapply puede ser utilizada en forma directa o con función.

Si A es una tabla con columnas 1,2 y 3 como factores y 5,6 y 7 como variables, los siguientes procedimientos son válidos:

```
tapply.stat(A[,5:7], A[,1:3],mean)
tapply.stat(A[,5:7], A[,1:3],function(x) mean(x,na.rm=TRUE))
tapply.stat(A[,c(7,6)], A[,1:2],function(x) sd(x)*100/mean(x))
```

Coeficiente de variación de un experimento

Si "modelo" es el objeto resultado de un análisis de variancia de la funcion aov() o lm() de R, entonces la funcion cv.model() calcula el coeficiente de variación

```
data(sweetpotato)
modelo <- modelo<-aov(yield ~ virus, data=sweetpotato)
cv.model(modelo)
[1] 17.16660
```

Asimetria y curtosis

Los resultados de asimetrica y curtosis obtenidos por SAS, Excel, son similares a las obtenidad por las funciones de agricolae:

Si x representa a un conjunto de datos:

x<-c(3,4,5,2,3,4,5,6,4,NA,7)

la asimetría se encuentra con:

```
skewness(x)
[1] 0.3595431
```

y la curtosis con:

kurtosis(x)

```
[1] -0.1517996
```

Valor tabular de waller-duncan

La función waller determina el valor, pero es necesario el valor de F calculado del ANOVA, los grados de libertad de la fuente de estudio y el error correspondiente. El valor K es la razón entre los dos errores. K=50, al nivel de 0.10 del error tipo I. K=100 a 0.05 y K=500 a 0.01, K puede tomar cualquier valor

La grafica de la función para valores diferentes de K, grados de libertad de 5 y 15 para el numerador y denominador respectivamente, y valores de Fc de 2, 4 y 8 se presenta en la figura 6.10

```
q<-5
f<-15
K<-seq(10,1000,100)
n<-length(K)
y<-rep(0,3*n)
dim(y)<-c(n,3)
for(i in 1:n) y[i,1]<-waller(K[i],q,f,Fc=2)
for(i in 1:n) y[i,2]<-waller(K[i],q,f,Fc=4)
for(i in 1:n) y[i,3]<-waller(K[i],q,f,Fc=8)
plot(K,y[,1],type="l",col="blue",ylab="waller")
lines(K,y[,2],type="l",col="red",lty=2,lwd=2)
lines(K,y[,3],type="l",col="green",lty=4,lwd=2)
legend("topleft",c("2","4","8"),col=c("blue","red","green"),lty=c(1,
8,20),lwd=2,title="Fc")
title(main="Waller en función de K")
```

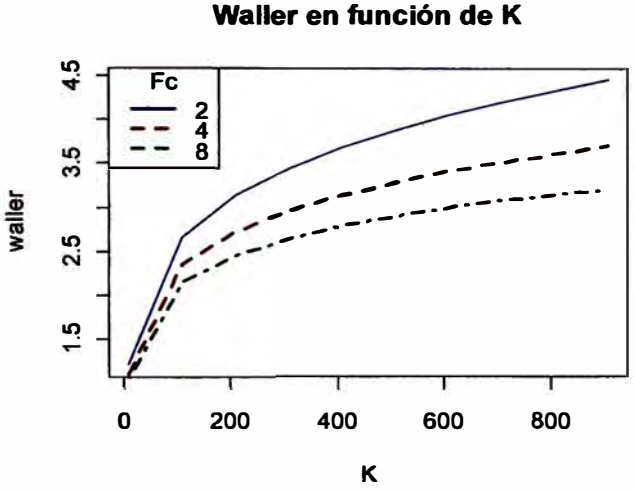


Figura 6.10.1. función de waller a diferente Valor del parámetro K y Fc.

AUDPC

Área bajo la curva del progreso de la enfermedad, figura 6.10.2. La función AUDPC calcula el absoluto y relativo del progreso de la enfermedad. Se requiere medir la enfermedad en porcentaje durante varias fechas de preferencia equidistantes.

```
dias<-c(7, 14,21,28,35,42)
evaluacion<-data.frame(E1=10,E2=40,E3=50,E4=70,E5=80, E6=90)
plot(dias,evaluacion,type="h",ylim=c(0,100),axes=F,col="red",xlab="D
ías", ylab="Evaluacón")
lines(dias,evaluacion,col="red")
axis(1,dias)
axis(2,seq(0,100,10))
abline(v=7,h=100,lty=4,lwd=2,col="blue")
abline(v=42,h=0,lty=4,lwd=2,col="blue")
audpc(evaluacion,dias)
audpc(evaluacion,dias,"relative")
text(15,80,"Audpc Absoluto = 2030")
text(15,70,"Audpc Relativo = 0.58")
```

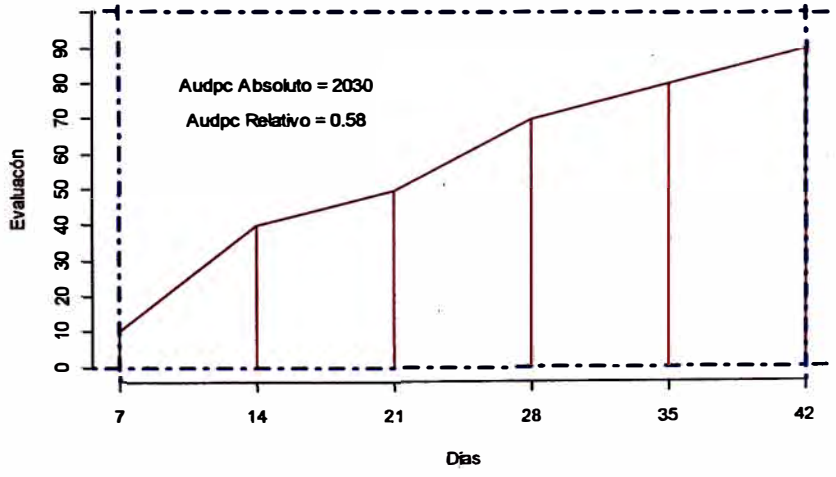


Figura 6.10.2. AUDPC: Área bajo la curva

NO ADITIVIDAD

La prueba de noaditividad de un modelo es utilizada cuando el supuesto de aditividad del modelo experimental no se cumple. Esta prueba verifica tal supuesto; se espera que la noaditividad no sea significativa.

Para utilizar esta función, es necesario tener datos experimentales, los cuales se pretende estudiar bajo un modelo lineal en los parámetros en un modelo de dos factores.

Para el ejemplo se utilizará los datos de un experimento en papa de la librería "agricolae". Un factor son los bloques, y otro, las variedades.

```
data(potato )
potato[,1]<-as.factor(potato[,1])
model<-lm(cutting ~ date + variety,potato)
df<-df.residual(model)
MSerror<-deviance(model)/df
attach(potato)
analysis<-nonadditivity(cutting, date, variety, df, MSerror)
detach(potato)

Tukey's test of nonadditivity
cutting

P : 15.37166
Q : 77.4444

Analysis of Variance Table

Response: residual
             Df Sum Sq Mean Sq F value Pr(>F)
Nonadditivity  1  3.051   3.051   0.922 0.3532
Residuals     14 46.330  3.309
```

Segun este resultado, el modelo es aditivo porque el p.value es 0.35.

Tabla 6.10. Referencia de codigos ascii para el uso de símbolos

Tabla de códigos ASCII utilizados en R					
Código	Símbolo	Código	Símbolo	Código	Símbolo
92	\	124	|	64	@
47	/	60	<	94	^
91	[	62	>	35	#
93	]	61	=	36	$
40	(	34	"	37	%
41	)	126	~	38	&
123	{	58	:	39	'
125	}	59	;		