

TRABAJO DE FIN DE MÁSTER

“GENERACIÓN DE UN PROGRAMA BIOINFORMÁTICO DE AYUDA DIAGNÓSTICA PARA ENFERMEDADES DEL SISTEMA INMUNITARIO DE BASE GENÉTICA”

Autor: **Iker Núñez Carpintero**

Directores: **Dr. Eduardo Arranz Sanz**
Dr. José Antonio Garrote Adrados

Titulación: Máster en Investigación Biomédica IBGM-UVa



Universidad de Valladolid

2018

ÍNDICE

| | |
|---|-----------|
| Glosario de términos..... | 1 |
| Resumen..... | 3 |
| Abstract..... | 3 |
| 1. Introducción..... | 4 |
| 1.1. Enfermedades autoinmunes y enfermedad celíaca..... | 4 |
| 1.2. Bioinformática aplicada a la medicina..... | 5 |
| 2. Materiales y métodos..... | 6 |
| 2.1. Software estadístico R..... | 6 |
| 2.1.1. Cálculo de riesgo de enfermedad celíaca..... | 6 |
| 2.1.1.1 Función <i>CeliacRisk</i> | 7 |
| 2.1.1.2 Función <i>CeliacRiskClasification</i> | 8 |
| 2.1.2 Cálculo de riesgos de enfermedad celíaca en la descendencia..... | 8 |
| 2.1.2.1 Función <i>InheritanceC</i> | 8 |
| 2.1.3. Librerías de apoyo..... | 10 |
| 2.1.3.1. <i>gWidgets2</i> | 10 |
| 2.1.3.2. <i>ReporteRs</i> | 10 |
| 2.1.3.3. <i>xlsx</i> | 10 |
| 2.1.3.4. <i>rJava</i> | 11 |
| 3. Resultados y funcionamiento del programa..... | 11 |
| 3.1. <i>HLADQCalculator</i> | 11 |
| 3.2. <i>CeliacRiskCalculator</i> | 12 |
| 3.3. <i>InheritanceDQCalculator</i> | 14 |
| 3.4. Estudio preliminar..... | 16 |
| 4. Discusión..... | 18 |
| 5. Conclusiones..... | 19 |
| 6. Bibliografía..... | 20 |
| Anexos: Código de las funciones del programa <i>CeliacDQ</i>..... | 22 |
| Anexo I. <i>CeliacRisk</i> | 22 |
| Anexo II. <i>CeliacRiskClasification</i> | 23 |
| Anexo III. <i>InheritanceC</i> | 25 |
| Anexo IV. <i>HLADQCalculator</i> | 27 |
| Anexo V. <i>CeliacRiskCalculator</i> | 27 |
| Anexo VI. <i>InheritanceDQCalculator</i> | 34 |

Glosario de términos

Argumento: Dentro de una función de un lenguaje informático, cada uno de las diferentes variables que se pueden configurar.

Biblioteca: Véase librería.

Bioinformática: Aplicación y desarrollo de de técnicas computacionales a la biología.

ESPGHAN: Sociedad europea de gastroenterología, hepatología y nutrición pediátrica.

GNOME: Entorno de escritorio para sistema operativo GNU/Linux, Unix y derivados, compuesto completamente por software libre.

GNU: Sistema operativo de tipo Unix, desarrollado por y para el proyecto GNU, un proyecto de software libre anunciado por Richard Stallman para producir dicho sistema operativo.

GNU/Linux: Sistema operativo de tipo Unix, que funciona como combinación del sistema GNU y el núcleo Linux, desarrollado por Linus Torvalds.

GO: Gene Ontology.

GTK+: Conjunto de bibliotecas para el desarrollo de interfaces gráficas de usuario en entornos gráficos GNOME.

GUI: Interfaz gráfica de usuario, un programa informático que actúa, mediante imágenes y objetos gráficos, como medio de comunicación entre usuario y sistema informático.

HLA: Antígenos leucocitarios humanos. El equivalente al complejo mayor de histocompatibilidad en *Homo sapiens*.

HLA-DQ: Molécula de presentación antigénica al receptor de las células T (TCR), que en ciertas situaciones es capaz de conducir a la existencia de riesgo de enfermedad celíaca, al presentar el antígeno inductor de la enfermedad, el gluten.

HLA-DQA: Gen codificante de la cadena α de la molécula HLA-DQ.

HLA-DQB: Gen codificante de la cadena β de la molécula HLA-DQ.

HPO: Human Phenotype Ontology.

Librería: En el contexto de un lenguaje de programación, conjunto de funciones codificadas en el lenguaje destinadas a ejercer una serie de acciones específicas.

MHC: Complejo mayor de histocompatibilidad. Conjunto de genes que codifican para la generación de proteínas implicadas en la presentación de antígeno al receptor de las células T (TCR).

OMIM: Online Mendelian Inheritance in Man.

Paquete: Véase librería.

R: Lenguaje de programación estadístico R.

Rg: Valor de riesgo de enfermedad celíaca del genotipo.

ρ : Valor de riesgo de enfermedad celíaca asociado a cada una de las diferentes moléculas HLA-DQ presentadoras del gluten al TCR.

tcltk: Lenguaje de programación libre dedicado a la creación de interfaces gráficas de usuario.

TCR: Receptor de células T.

Unix: Sistema operativo portable, multitarea y multiusuario, del cual es clon el sistema operativo libre GNU/Linux.

Resumen

Históricamente, la medicina clínica ha tendido a definir las enfermedades en base a sus signos clínicos, dejando de lado las características funcionales de los genes mutados causales, algo de lo que, hasta la explosión de la biología y la genética molecular se tenía un conocimiento muy reducido.

Recientemente, la biología de sistemas está reavivando el interés en estudiar la relación global entre la información clínica y molecular de las enfermedades genéticas, algo que es especialmente importante en varios campos de la medicina, como en el diagnóstico de enfermedades genéticas, o la clasificación de las mismas. Mediante herramientas bioinformáticas es posible procesar la enorme y creciente cantidad de información presente en las distintas bases de datos de enfermedades y genes causales de las mismas.

En este trabajo se presenta una herramienta bioinformática en el lenguaje de programación estadístico R para la ayuda diagnóstica de enfermedades del sistema inmune de base genética, más concretamente de la enfermedad celíaca. El programa posee una interfaz gráfica de utilización intuitiva para el usuario final y es capaz de procesar la información de la genética HLA del paciente para calcular el riesgo asociado a la genética de padecer la enfermedad.

Palabras clave: enfermedad autoinmune, bioinformática, HLA, interfaz gráfica de usuario, riesgo genético, librería, R.

Abstract

Historically, medicine tends to define diseases according to their clinical signs, putting aside the functional characteristics of the responsible mutated genes. As a result, knowledge in this field has been poor until the first molecular genetics studies with genetic diseases.

Nowadays, Systems biology is starting to study the global relationship between clinic and molecular information of genetic diseases, which is especially important in several medicine fields, such as genetic disease diagnosis. Using bioinformatic tools it is possible to process the large and always growing information quantities contained in disease databases.

In this dissertation, a bioinformatic tool for the diagnosis of genetic based immune-related diseases is presented, more accurately, for celiac disease. The program has an intuitive graphical user interface and is capable of processing HLA genetics information of a patient and calculates the genetic based risk of suffering the disease.

Keywords: autoimmune disease, bioinformatics, HLA, graphical user interface, genetic based risk, library, R.

1. Introducción

1.1. Enfermedades autoinmunes y enfermedad celíaca

Las enfermedades autoinmunes componen un amplio abanico de cuadros clínicos causados por la actuación inapropiada del sistema inmunitario del paciente contra antígenos del propio individuo.

Algunas de las enfermedades autoinmunes más reseñables son la esclerosis múltiple, el lupus eritematoso sistémico, la artritis reumatoide, la diabetes tipo I y la enfermedad celíaca.

En todas ellas existe una fuerte interacción tanto de componentes genéticos como de factores ambientales. A nivel molecular, la genética HLA de los pacientes es crítica para este tipo de cuadros clínicos, si bien en la mayoría de estas enfermedades existe un factor ambiental desencadenante de la enfermedad como tal.

HLA hace referencia a los genes de los antígenos leucocitarios humanos, los cuales son el equivalente en *Homo sapiens* al complejo mayor de histocompatibilidad (MHC), es decir, son los genes codificantes para moléculas de membrana especializadas en la presentación de antígenos al receptor de linfocitos T (TCR).

La enfermedad celíaca es un buen ejemplo de enfermedad autoinmune con alta influencia ambiental. Así, la ESPGHAN (Sociedad europea de gastroenterología, hepatología y nutrición pediátrica) la define como una enfermedad autoinmune sistémica, desencadenada por el gluten en pacientes genéticamente susceptibles (Husby et al., 2012), entendiéndose como susceptibilidad la presencia de moléculas HLA-DQ2 (tanto DQ2.5 como DQ2.2) y/o moléculas HLA-DQ8.

Cabe destacar también que, aunque la presencia de moléculas HLA-DQ7.5 casi no conlleva riesgo a desarrollar la enfermedad *per se* (debido a su baja prevalencia), no es correcto descartar la enfermedad celíaca en casos donde sólo aparezca dicha molécula de riesgo.

El diagnóstico de la enfermedad celíaca puede llegar a ser muy complejo debido a la enorme variedad de signos clínicos inespecíficos que presenta. Por ello, la ESPGHAN propone dos grupos de pacientes donde debe considerarse realizar pruebas de enfermedad celíaca (Husby et al., 2012):

- **Niños y adolescentes con síntomas de cuadros intestinales sin explicación clara:** diarrea intermitente, pérdida de peso, pubertad retardada, dolor abdominal crónico, anemia por deficiencia en hierro, náuseas, vómitos, etc.

- **Niños y adolescentes asintomáticos**, pero con riesgo incrementado de sufrir la enfermedad como resultado de otras dolencias **autoinmunes**, como diabetes tipo I, síndrome de Down, síndrome de Williams, síndrome de Turner, enfermedad autoinmune tiroidea, etc.

Además de estos dos grupos de riesgo también cabe valorar la realización de pruebas a pacientes que presentan familiares con la enfermedad.

Como herramientas diagnósticas incluidas en esa misma disertación encontramos tests de anticuerpos específicos de la enfermedad celíaca, biopsias de intestino delgado y pruebas genéticas.

En la actualidad, la prueba genética principal que se lleva a cabo en la enfermedad celíaca consiste en analizar mediante métodos de biología molecular los alelos HLA del paciente, ya que la no aparición de alelos de riesgo constituye una muy buena manera de descartar la enfermedad celíaca.

1.2. Bioinformática aplicada a la medicina

Clásicamente, la medicina tiende a describir las enfermedades únicamente en función de los signos clínicos (fenotipo) que éstas presentan, hasta el auge de la biología y la genética molecular, que han permitido aumentar enormemente el conocimiento de la funcionalidad de los genes causales de gran cantidad de enfermedades.

La información existente acerca de la relación entre ambos “mundos” (el mundo de la medicina clínica, dedicada a la descripción por signos clínicos de las enfermedades y el mundo de la biología y genética molecular, enfocado a describir las causas moleculares y genéticas de las enfermedades), generalmente estaba centrada en las enfermedades individuales. Sin embargo ya existen trabajos recientes (Chagoyen y Pazos, 2016) que tratan de llevar a cabo este tipo de estudios globalmente ahora que la información fenotípica y molecular de las enfermedades es mucho mayor.

Por otro lado, el enorme y creciente tamaño de información biológica actual en muchas bases de datos requiere de herramientas capaces de tratar tal acumulación de forma efectiva y en menos tiempo del que llevaría su tratamiento manual.

Con dicho objetivo, aparece la bioinformática (aplicación de técnicas computacionales a la biología). La bioinformática tiene una aplicación clave en muchas ramas de la biología (biomedicina, bioquímica, genética...) y su importancia crece según aumenta la complejidad en el conocimiento biológico: sólo mediante el uso de herramientas informáticas puede analizarse e integrar globalmente la información de bases de datos de gran tamaño, como es el caso de OMIM (Online Mendelian Inheritance in Man) (McKusick-Nathans Institute of Genetic Medicine, 2017), HPO (Human Phenotype Ontology) (Köhler et al., 2017), GO (Gene Ontology) (The Gene Ontology Consortium, 2015) o como es el caso de otros muchos proyectos (como el de Menche et al., 2015) que tratan de catalogar el incompleto interactoma humano, el conjunto de las interacciones entre los productos génicos del genoma humano.

Algunas de estas herramientas se dedican al diagnóstico clínico de enfermedades, como The Phenomizer (Köhler et al., 2009), un programa web que utiliza información de signos clínicos de HPO para el diagnóstico de enfermedades genéticas incluidas en OMIM.

Siguiendo la idea de este tipo de herramientas, el objetivo de este trabajo de fin de máster es el desarrollo y presentación de *CeliacDQ*, un programa generado en el lenguaje de programación R para la ayuda diagnóstica de la

enfermedad celíaca utilizando la información genética HLA obtenida de los pacientes a analizar (**Figura 1**):

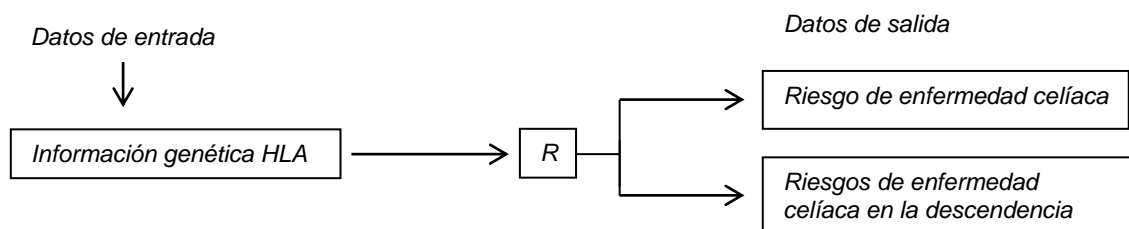


Figura 1. Esquema del programa objetivo del trabajo

2. Materiales, métodos y funcionamiento del programa

2.1. Software Estadístico R

Este trabajo de fin de máster se ha llevado a cabo mediante la herramienta de programación de carácter libre para cálculos estadísticos “R” (R Core Team, 2018).

R es un lenguaje de programación concebido para el tratamiento estadístico de información. Utilizado ampliamente en Biología, junto a otros lenguajes de programación como Python (Python Software Foundation, 2018), R permite la instalación de las llamadas “librerías” o “paquetes”, conjuntos de nuevos datos y funciones, producidas bien por los desarrolladores o bien por los propios usuarios de R, que proveen al lenguaje de nuevas posibilidades para realizar procedimientos estadísticos y matemáticos de muy diferente índole.

En este caso, tres son los paquetes específicos utilizados para la generación del programa: “gWidgets2” (Verzani, 2016), “xlsx” (Dragulescu, 2014), “ReporteRs” (Gohel, 2018) y “rJava” (Urbanek, 2017) junto con sus respectivas dependencias (paquetes necesarios para el funcionamiento de los ya mencionados, y que se instalan de forma automática al descargarlos).

Estas librerías se utilizan como apoyo en la construcción del programa *CeliacDQ*, mientras que los métodos que la herramienta utiliza para llevar a cabo el cálculo del riesgo de enfermedad celíaca en base a la genética HLA del individuo y los riesgos en la descendencia fueron programados *de novo*. Más adelante se detalla el uso de estos paquetes y la naturaleza de los métodos de cálculo de riesgo.

De esta manera, las funciones de estos paquetes se interrelacionan con muchas de las que están predefinidas en los paquetes básicos del propio lenguaje, como la librería “base” (contiene las estructuras de control del lenguaje R y las funciones básicas del mismo -“if”, “else”, “for”, etcétera- que permiten la realización de tareas simples); dando con ello lugar al código fuente del programa.

2.1.1 Cálculo de riesgo de enfermedad celíaca

Siguiendo el objetivo de este trabajo, la primera funcionalidad de *CeliacDQ* consiste en calcular el riesgo de enfermedad celíaca asociada al genotipo de un individuo en base a las posibles moléculas HLA-DQ que el paciente puede generar como resultado de sus alelos.

El programa toma los alelos de 2 genes en particular: HLA-DQA y HLA-DQB, ya que la combinación de un alelo del HLA-DQA (codificante para la cadena α de la molécula) con otro del HLA-DQB (codificante para la cadena β) es lo que genera la molécula HLA-DQ. De esta forma, el programa utiliza como datos de partida los dos alelos del individuo para cada gen, y computa las posibles combinaciones de los mismos.

En la **tabla 1** se muestran las cuatro combinaciones de riesgo que utiliza el programa, en base a la información obtenida de Núñez et al., (2018), junto al valor de riesgo (p) asociado a cada combinación.

Dichos valores de riesgo p se obtuvieron utilizando como referencia los datos de prevalencia de la enfermedad celíaca en base a diferentes genotipos HLA-DQ obtenidos por Fernández-Cavada-Pollo et al., (2013), y teniendo en cuenta la clasificación de riesgo descrita por Núñez et al., en el ya mencionado artículo, que se muestra posteriormente en esta disertación (**tabla 2**).

| Alelo HLA-DQA | Alelo HLA-DQB | Proteína DQ | Riesgo (p) |
|---------------|---------------|-------------|----------------|
| A1*05 | B1*02 | DQ2.5 | 1 |
| A1*03 | B1*03:02 | DQ8 | 0.1 |
| A1*02 | B1*02 | DQ2.2 | 0.064 |
| A1*05 | B1*03:01 | DQ7.5 | 0.01 |

Tabla 1. Combinaciones de riesgo de enfermedad celíaca y riesgo asociado a las proteínas DQ que generan. Modificado de Núñez et al., (2018). Riesgos (p) obtenidos a partir de datos de prevalencia de Fernández-Cavada-Pollo et al., (2013)

Cualquier otro tipo de combinación de alelos es considerada como sin riesgo por el programa. Para aquellos alelos que tienen algunas variantes (por ejemplo el A1*05) se considera que las fluctuaciones en cuanto a riesgo son despreciables, por lo que se consideran todos como parte de una única opción, siguiéndose de esta manera las directrices marcadas por Núñez et al., (2018).

Este valor de riesgo (p) asociado a cada proteína DQ es la referencia, como veremos ahora, para calcular el valor de riesgo del genotipo HLA del individuo (R_g) por parte del programa.

El programa utiliza dos funciones para calcular el valor de riesgo y clasificarlo en forma de escala de riesgo: “*CeliacRisk*” y “*CeliacRiskClasification*”.

2.1.1.1 Función *CeliacRisk*

La función *CeliacRisk* es la primera que se generó para el programa y constituye la base principal del mismo. Dados los cuatro alelos de un paciente para los genes HLA-DQA y HLA-DQB, *CeliacRisk* calcula las cuatro posibles combinaciones de los mismos para dar lugar a moléculas HLA-DQ, y calcula el valor de riesgo de enfermedad celíaca del genotipo (R_g).

El valor R_g se define como el sumatorio de los valores de riesgo (p) de cada uno de las posibles moléculas que el individuo puede generar en base a sus alelos y varía desde 0 (sin riesgo) hasta 4 (riesgo máximo).

En el **anexo I** se incluye el código de la función *CeliacRisk*, cuya extensión alcanza un total de 80 líneas.

2.1.1.2 Función *CeliacRiskClasification*

Esta segunda función lleva a cabo el mismo cálculo que la anterior, sin embargo en vez de devolver el valor R_g , *CeliacRiskClasification* toma ese valor para determinar la clasificación del riesgo del genotipo, en función de lo indicado en la tabla 2:

| Valor de riesgo (R_g) | Clasificación del riesgo |
|---------------------------|--------------------------|
| $R_g > 1.074$ | Muy alto |
| $0.2 \leq R_g \leq 1.074$ | Alto |
| $0.064 \leq R_g < 0.2$ | Moderado |
| $0 < R_g < 0.064$ | Bajo |
| $R_g = 0$ | Sin riesgo |

Tabla 2. Intervalos de riesgo utilizados por *CeliacRiskClasification*.

Ambas funciones descritas incluyen además una excepción, pues la combinación DQ2.5/7.5, de acuerdo a los valores de p descritos en la tabla 1 daría lugar a un riesgo muy alto debido a la existencia de un segundo DQ2.5 en trans, cuando dicho combinación corresponde a riesgo alto (Núñez et al, 2018).

Para las dos implementaciones, existen 5 argumentos:

- **a1**: Primero de los alelos del gen HLA-DQA
- **a2**: Segundo alelo del gen HLA-DQA
- **b1**: Primero de los alelos del gen HLA-DQB
- **b2**: Segundo alelo del gen HLA-DQB
- **verbose**: Un argumento lógico o booleano (sólo toma los valores TRUE/FALSE) para indicar si recibir mensajes de información mientras dura el proceso de cálculo de riesgo.

El **anexo II** del presente trabajo corresponde al código de la función *CeliacRiskClasification*.

2.1.2 Cálculo de riesgos de enfermedad celíaca en la descendencia

2.1.2.1 Función *InheritanceC*

La tercera función del programa trata de cumplir con el segundo objetivo del mismo: calcular los riesgos de enfermedad celíaca asociados a los posibles genotipos de la descendencia entre dos individuos en función de sus genotipos HLA-DQ.

Para ello, la función *InheritanceC* calcula el tablero de Punnett correspondiente al cruce de ambos individuos.

Con la idea de ilustrar esto más correctamente, supongamos que tenemos dos padres, cuyos genotipos son, para el padre, los alelos “1” y “2” para el gen A y los alelos “3”y “4” para el gen B; y para el caso de la madre, los alelos “5” y “6” para el gen A y los alelos “7” y “8” para el gen B. El tablero de Punnett correspondiente a ese caso sería el siguiente (tabla 3):

| Gametos | 5,7 | 5,8 | 6,7 | 6,8 |
|----------------|---------|---------|---------|---------|
| 1,3 | 1,3,5,7 | 1,3,5,8 | 1,3,6,7 | 1,3,6,8 |
| 1,4 | 1,4,5,7 | 1,4,5,8 | 1,4,6,7 | 1,4,6,8 |
| 2,3 | 2,3,5,7 | 2,3,5,8 | 2,3,6,7 | 2,3,6,8 |
| 2,4 | 2,4,5,7 | 2,4,5,8 | 2,4,6,7 | 2,4,6,7 |

Tabla 3. Tablero de Punnett calculado por la función *InheritanceC*.

InheritanceC calcula el tablero de Punnett y lo rellena llamando a las funciones *CeliacRisk* y *CeliacRiskClasification*. Cuando elegimos que los datos sean los valores Rg de cada posible genotipo, la función llama a *CeliacRisk*, mientras que si elegimos que sean los grupos de riesgo al que pertenecen, siguiendo la información de la tabla 2, la función llama a *CeliacRiskClasification*.

De esta forma, la función *InheritanceC* incluye hasta un total de 10 argumentos diferentes:

- **ap1**: Primer alelo del gen HLA-DQA del padre.
- **ap2**: Segundo alelo del gen HLA-DQA del padre.
- **am1**: Primer alelo del gen HLA-DQA de la madre.
- **am2**: Segundo alelo del gen HLA-DQA de la madre.
- **bp1**: Primer alelo del gen HLA-DQB del padre.
- **bp2**: Segundo alelo del gen HLA-DQB del padre
- **bm1**: Primer alelo del gen HLA-DQB de la madre
- **bm2**: Segundo alelo del gen HLA-DQB de la madre
- **mode**: Permite elegir entre las funciones *CeliacRisk* y *CeliacRiskClasification* para rellenar el tablero de Punnett. Para elegir *CeliacRisk*, el valor del argumento “mode” será “Math”, mientras que para escoger *CeliacRiskClasification* el valor será “Class”.
- **verbose**: De nuevo un argumento lógico (sólo toma los valores TRUE/FALSE) para indicar si recibir mensajes de información mientras dura el proceso de cálculo del tablero de Punnett.

En el **anexo III** del trabajo se incluye el código fuente de la función *InheritanceC*.

2.1.3 Librerías de apoyo

2.1.3.1 *gWidgets2*

Mediante la librería de R “*gWidgets2*” se generó la interfaz de usuario del programa (**GUI**), con el objetivo de hacer más fácil e intuitivo el manejo del mismo por parte del usuario final, ya que éste podría no tener experiencia en el manejo de R.

gWidgets2 permite el uso de dos herramientas informáticas diferentes para la generación de interfaces de usuario en R:

- 1. **GTK+**, un conjunto de librerías para la producción de GUIs en GNOME (The GNOME Project, 2018). GNOME constituye uno de los entornos de escritorio para sistemas GNU/Linux más utilizados, perteneciendo así al proyecto GNU de software libre.
- 2. **tcltk** (Tcl Core Team, 2018) es un lenguaje de programación libre dedicado del mismo modo a la creación de estas interfaces. Ambas herramientas son independientes de lenguaje de programación R, de modo que el paquete actúa como intermediario permitiendo el uso de las dos herramientas mediante código del lenguaje R.

Ambas herramientas se pueden utilizar instalando dos paquetes adicionales del mismo autor a *gWidgets2*, los paquetes *gWidgets2RGtk2* -el cual requiere también la instalación del paquete *RGtk2* (Lawrence y Lang, 2010)- y *gWidgets2tcltk*, que hace uso de la librería *tcltk* que forma parte de las librerías base del lenguaje R, siendo

Para el uso de este programa más fácilmente recomendamos trabajar con **GTK+**, ya que fue el utilizado durante la generación del programa, aunque el uso de *tcltk* es también posible.

2.1.3.2 *ReporteRs*

La librería “*ReporteRs*” está orientada a la generación y manejo de archivos de Microsoft Word ® en R. Con la ayuda de las funciones incluidas en *ReporteRs*, *CeliacDQ* permite al usuario generar un informe de los riesgos de enfermedad celíaca del individuo analizado, incluyendo sus datos, los datos del facultativo responsable, las posibles moléculas HLA-DQ que puede generar el paciente, el valor de riesgo del genotipo (Rg) y la clasificación del mismo a la que pertenece dicho valor.

2.1.3.3 *xlsx*

“*xlsx*” es un paquete dedicado a la producción y uso de documentos de Microsoft Excel ® en R. Utilizando esta librería, se añadió al programa la posibilidad de guardar la información obtenida al calcular los riesgos de herencia en un archivo de formato Excel (.xlsx), el cual contiene, en una hoja el tablero de Punnett calculado por la función *InheritanceC* y en una segunda las frecuencias y porcentajes de cada uno de los resultados obtenidos, para su tratamiento posterior por parte del usuario.

2.1.3.4 rJava

“rJava” convierte código del lenguaje de R al lenguaje de programación Java, siendo necesaria su instalación para el uso de los ya mencionados *ReporteRs* y *xlsx*, ya que ambos utilizan implementaciones del paquete para poder llevar a cabo sus funciones, lo que hace a *CeliacDQ* dependiente de la instalación de la versión más reciente del lenguaje Java.

3. Resultados y funcionamiento del programa

Haciendo uso de las 4 librerías ya mencionadas y de las funciones *CeliacRisk*, *CeliacRiskClasification* e *InheritanceC*, se procedió a generar la interfaz de usuario del programa, para lo cual, se crearon otras tres funciones: *CeliacRiskCalculator*, *InheritanceDQCalculator* y *HLADQCalculator*.

De forma general, el esquema de la **figura 2** resume el funcionamiento del programa, que se desglosa en los siguientes apartados:

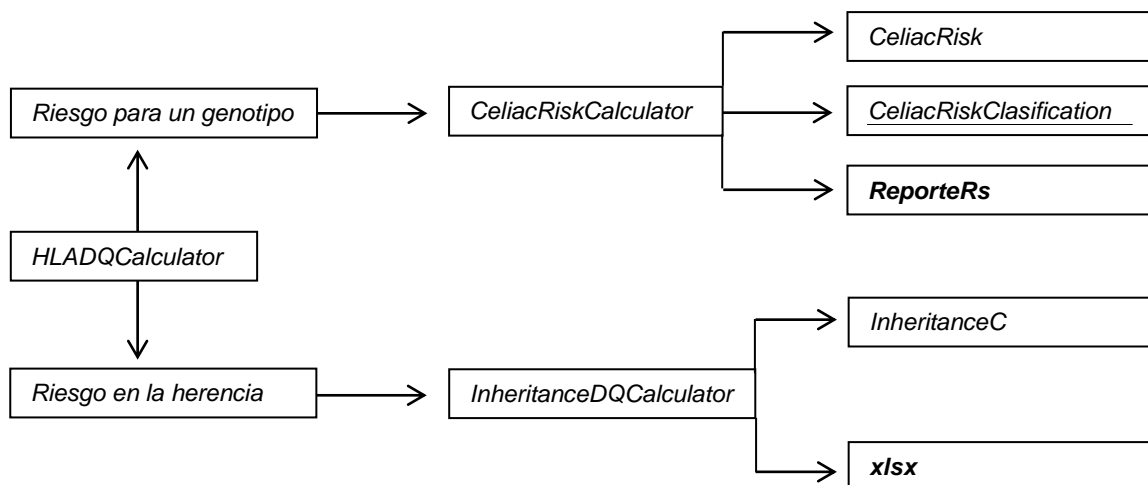


Figura 2. Esquema del funcionamiento de las llamadas entre funciones dentro del programa *CeliacDQ*.

3.1 HLADQCalculator

HLADQCalculator actúa como función de inicialización del programa *CeliacDQ*. Esta función, tras iniciar los diferentes paquetes necesarios para el funcionamiento del programa, inicia una ventana que incluye dos botones mediante los cuales es posible entre elegir los dos modos de funcionamiento del mismo: calcular el riesgo de un solo genotipo (llamando a *CeliacRiskCalculator*) o los riesgos de la posible descendencia entre dos individuos (llamando a *InheritanceDQCalculator*) (Figura 3):

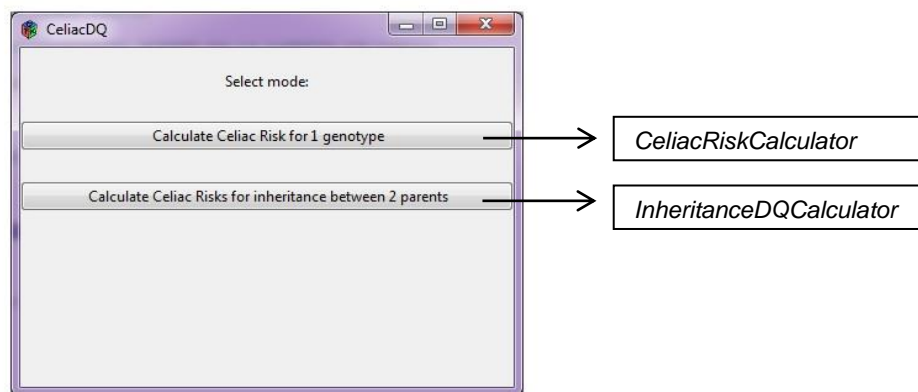


Figura 3. Ventana de *HLADQCalculator*.

El código de la función *HLADQCalculator* se encuentra en el **anexo IV** del trabajo.

3.2 *CeliacRiskCalculator*

Una vez lanzado el programa, si escogemos la opción de calcular un único genotipo, se inicia la función *CeliacRiskCalculator*. Al ser activada, esta función abre una segunda ventana como la de la **figura 4**, donde se podrán elegir los alelos HLA-DQ del paciente a analizar:

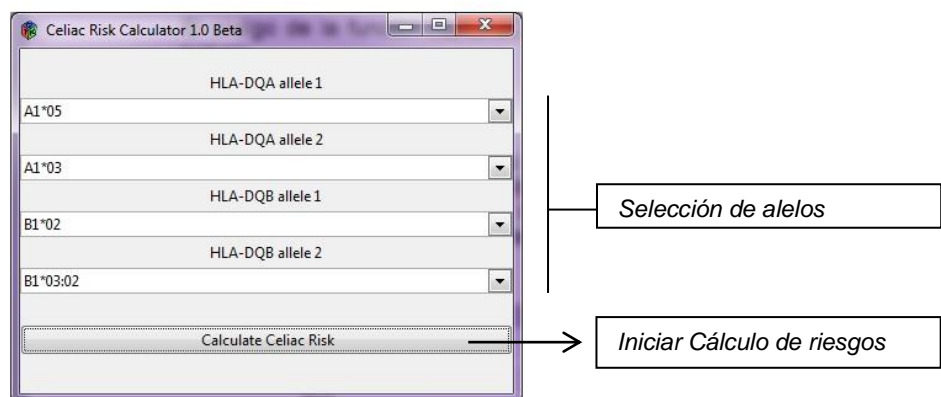


Figura 4. Ventana de *CeliacRiskCalculator*.

Tras proceder al cálculo de riesgos, *CeliacRiskCalculator* nos conducirá primero a una ventana con la información general de la computación realizada por las funciones *CeliacRisk* y *CeliacRiskClasification*, incluyendo los riesgos, la clasificación correspondiente de los mismos, los alelos introducidos y las posibles moléculas HLA-DQ que se podrían generar, junto a la opción de producir un informe de dichos resultados (**Figura 5**):

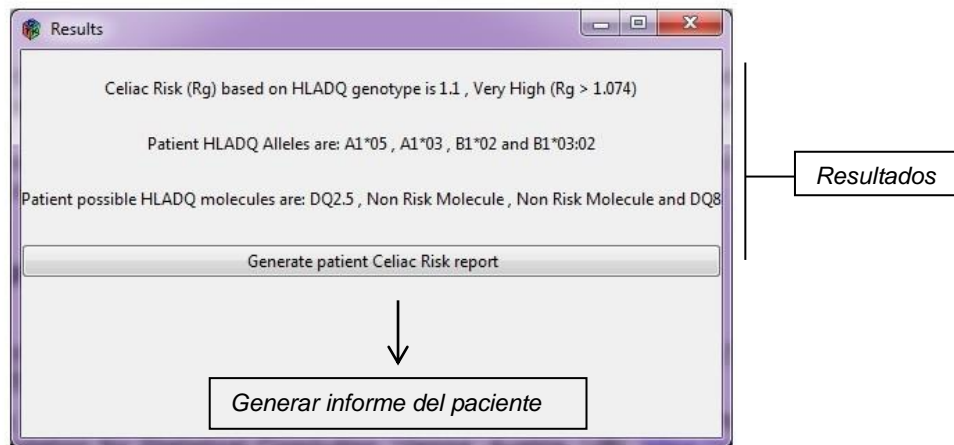


Figura 5. Cuadro de resultados generales de *CeliacRiskCalculator*.

De este modo, podemos pedirle al programa que genere un informe del análisis, para lo cual nos abrirá una nueva ventana donde introducir los datos del paciente y del usuario del mismo, así como elegir el directorio donde guardar el dicho informe. Es importante recalcar que el mismo debe guardarse con formato .docx para asegurar su perfecto funcionamiento (**Figura 6**):

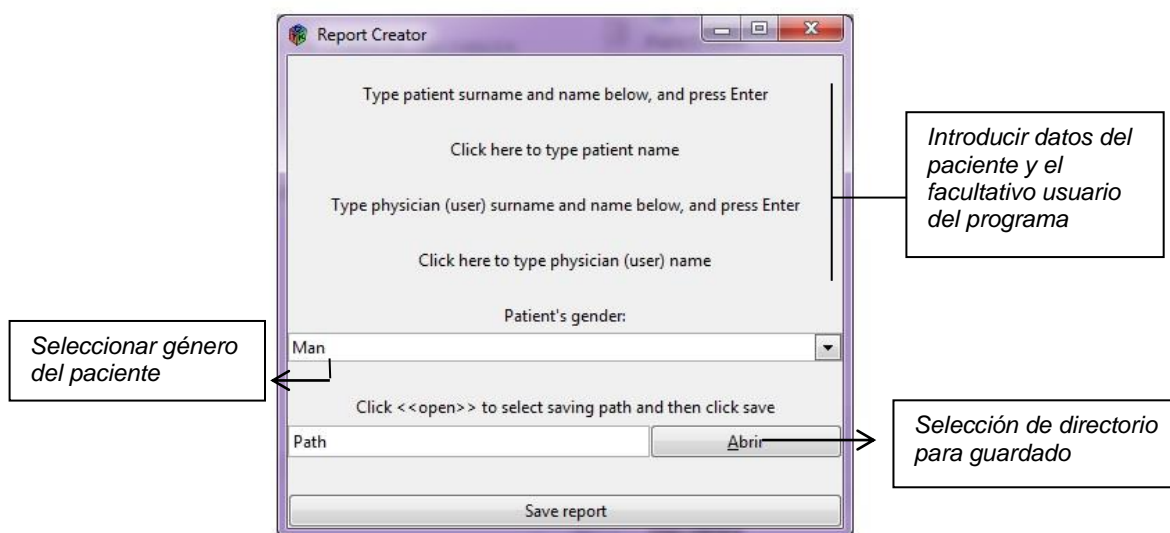


Figura 6. Ventana para la generación del informe de resultados.

Así, *CeliacRiskCalculator* llama a funciones del paquete **ReporteRs** y genera un informe con los datos de interés en formato Microsoft Word® (**Figura 7**):

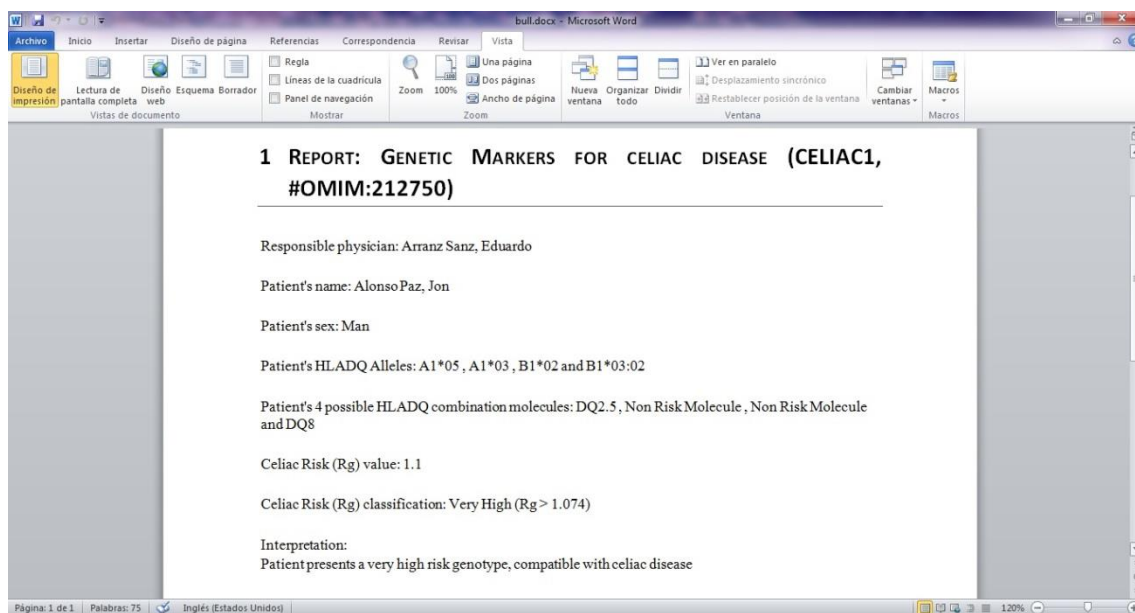


Figura 7. Imagen del informe generado por *CeliacRiskCalculator* en el programa Microsoft Word 2010 ®.

El **anexo V** describe el código de la función *CeliacRiskCalculator*.

3.3 *InheritanceDQCalculator*

Como se comentó previamente, escogiendo la opción para calcular los riesgos de enfermedad celíaca en la descendencia de dos individuos, *HLADQCalculator* llama a la última de las funciones del programa: *InheritanceDQCalculator* (**Figura 3**).

Esta función es la más larga y compleja de todo el programa (alcanzando 791 líneas, siendo la segunda más larga *CeliacRiskCalculator*, con 289) (**ver anexo VI**). Como veremos ahora, dicha complejidad radica en la cantidad de información y posibilidades que ofrece, pero que a la vez tiene que manejar.

Al iniciarse, *InheritanceDQCalculator* nos abre una nueva ventana como la de la **figura 8**, donde ahora aparecen hasta un total de ocho menús desplegables con los que escoger los alelos de ambos padres.

Se incluyen además, tres botones para tres modos de funcionamiento:

- 1. **“Math Mode”**: Llama a la función *InheritanceC*, para calcular el tablero de Punnett correspondiente a los genotipos dados, lanzando la función con el argumento **mode= “Math”**. Así, nos devuelve como resultado una ventana con el tablero, cuya información será los valores **Rg** de cada combinación (**Figura 9**).
- 2. **“Class Mode”**: Llama a la función *InheritanceC*, para computar el tablero de Punnett correspondiente a los genotipos introducidos por el usuario, iniciando la función con el argumento **mode= “Class”**.

Con ello, se nos abre una ventana con el tablero, cuya información será la clasificación de riesgo correspondiente al valor Rg de cada combinación (**Figura 10**).

- 3. **“Math & Class Mode”**: Llama a la función *InheritanceC*, para calcular el tablero de Punnett correspondiente a los genotipos dados, lanzando la función con el argumento mode= “Math”. Posteriormente, vuelve a hacer lo mismo pero con el argumento mode= “Class”.

Así, nos devuelve como resultado dos ventanas, una con el resultado que correspondería a haber seleccionado “Math mode” (**figura 9**) y una segunda con el resultado correspondiente a “Class mode” (**figura 10**).

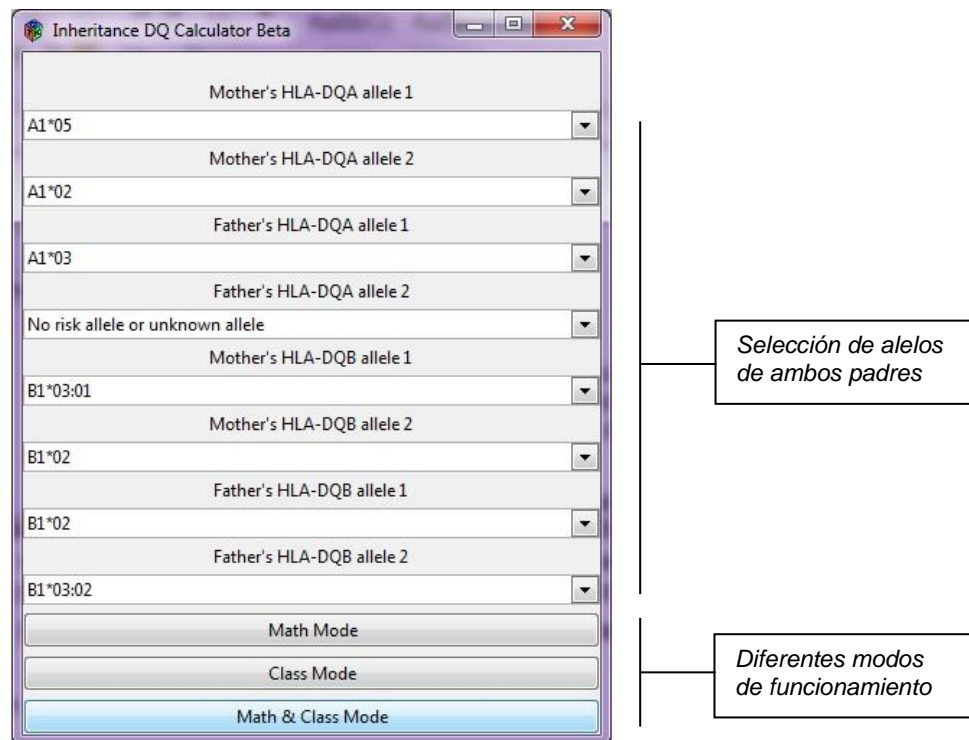


Figura 8. Ventana inicial de *InheritanceDQCalculator*

Como se puede observar en las **figuras 9 y 10**, se incluye, además del tablero de Punnett, las probabilidades de cada uno de los resultados y el porcentaje de los mismos. Dichos cálculos son realizados directamente por la función *InheritanceDQCalculator*, tomando el resultado de la función *InheritanceC*, es decir, el tablero de Punnett.

Por último, en ambos casos se añade la posibilidad de guardar las tablas en un libro Excel (en formato .xlsx), donde en una hoja aparecerá el tablero de Punnett, mientras que en la segunda aparecerán los datos de probabilidad de cada resultado. Para ello, *InheritanceDQCalculator* llama a funciones del paquete *xlsx* mencionado previamente en el apartado **2.1.3.3**.

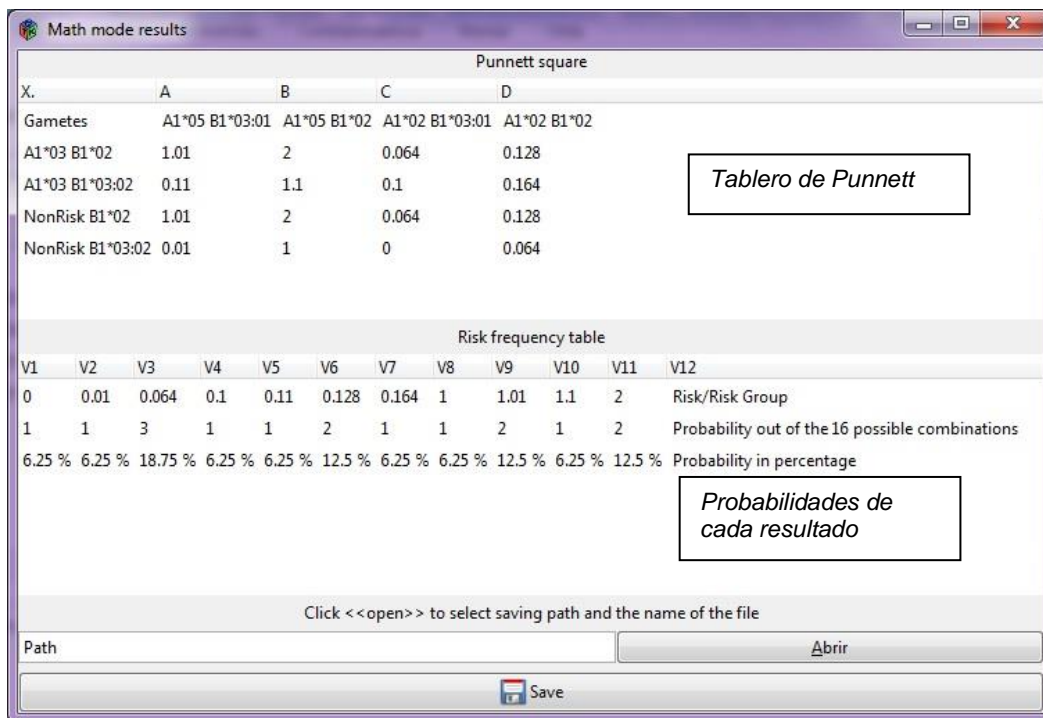


Figura 9. Cuadro de resultados correspondiente a seleccionar "Math Mode".

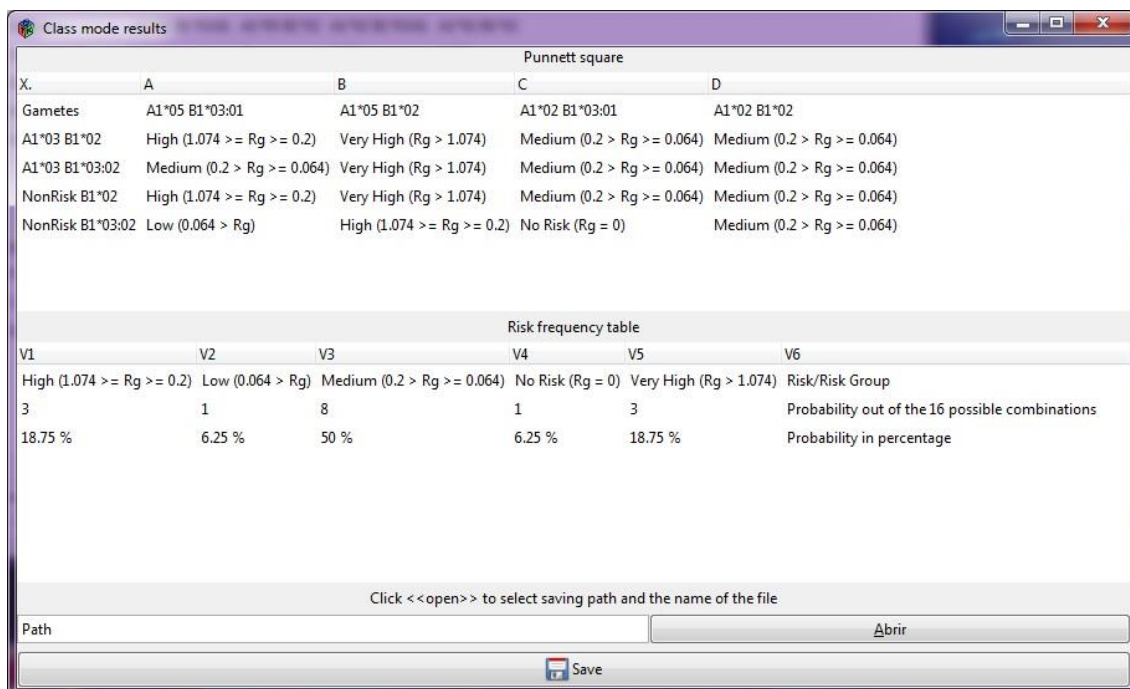


Figura 10. Cuadro de resultados correspondiente a seleccionar "Class Mode".

3.4. Estudio preliminar

Una vez generado el programa, decidimos probar el correcto funcionamiento de *CeliacDQ*. Para ello, se planteó llevar a cabo un estudio preliminar, es decir, una prueba de concepto en la que, utilizando datos de genética HLA de 30 pacientes recogidos en el Laboratorio de Genética Molecular del Servicio de Análisis Clínicos del Hospital Universitario Río-Hortega de Valladolid, observamos cómo el programa clasificaba correctamente los riesgos de estos pacientes siguiendo el ya citado sistema de Núñez et al., (2018).

En la **tabla 4** se **resumen** los resultados del estudio preliminar:

| HLA-DQA 1 | HLA-DQB 1 | HLA-DQA 2 | HLA-DQB 2 | Rg | Riesgo |
|-----------|-----------|-----------|-----------|-------|------------|
| A1*01 | B1*05:01 | A1*05 | B1*03:01 | 0.01 | Bajo |
| A1*01 | B1*05:01 | A1*01 | B1*05:01 | 0 | Sin riesgo |
| A1*03 | B1*03:02 | A1*03 | B1*03:03 | 0.2 | Alto |
| A1*01:03 | B1*03:03 | A1*02 | B1*06:03 | 0 | Sin riesgo |
| A1*01:04 | B1*02 | A1*05 | B1*05:03 | 1 | Alto |
| A1*01:04 | B1*02 | A1*02 | B1*05:03 | 0.064 | Medio |
| A1*01:04 | B1*03:01 | A1*03 | B1*05:03 | 0 | Sin riesgo |
| A1*01 | B1*02 | A1*02 | B1*05:01 | 0.064 | Medio |
| A1*01 | B1*03:01 | A1*03 | B1*06:02 | 0 | Sin riesgo |
| A1*03 | B1*02 | A1*05 | B1*03:02 | 1.1 | Muy alto |
| A1*01:02 | B1*03:03 | A1*03 | B1*05:01 | 0 | Sin riesgo |
| A1*01:02 | B1*05:05 | A1*01:04 | B1*06:02 | 0 | Sin riesgo |
| A1*01:03 | B1*06:03 | A1*01:03 | B1*06:03 | 0 | Sin riesgo |
| A1*01:02 | B1*03:02 | A1*03 | B1*06:02 | 0.1 | Medio |
| A1*02 | B1*02 | A1*05 | B1*03:01 | 1.074 | Alto |
| A1*05 | B1*02 | A1*05 | B1*02 | 4 | Muy alto |
| A1*01:02 | B1*03:03 | A1*03 | B1*06:02 | 0 | Sin riesgo |
| A1*01:02 | B1*02 | A1*02 | B1*06:02 | 0.064 | Medio |
| A1*01:03 | B1*05:01 | A1*01:01 | B1*06:03 | 0 | Sin riesgo |
| A1*01:04 | B1*03:03 | A1*02 | B1*05:03 | 0 | Sin riesgo |
| A1*02 | B1*02 | A1*02 | B1*02 | 0.256 | Alto |
| A1*02 | B1*03:01 | A1*05 | B1*03:03 | 0.01 | Bajo |
| A1*01:05 | B1*02 | A1*02 | B1*05:01 | 0.064 | Medio |
| A1*01:02 | B1*03:02 | A1*03 | B1*06:02 | 0.1 | Medio |
| A1*01:01 | B1*05:01 | A1*01:03 | B1*06:03 | 0 | Sin riesgo |
| A1*01:03 | B1*03:01 | A1*05 | B1*06:03 | 0.01 | Bajo |
| A1*02 | B1*02 | A1*03 | B1*03:01 | 0.064 | Medio |
| A1*01:02 | B1*03:01 | A1*05 | B1*05:02 | 0.01 | Bajo |
| A1*03 | B1*03:01 | A1*05 | B1*03:01 | 0.02 | Bajo |
| A1*01:01 | B1*03:02 | A1*03 | B1*05:01 | 0.1 | Medio |

Tabla 4. Genética HLA de los pacientes analizados en la prueba de concepto y riesgos de enfermedad celíaca asociados a los mismos. Cada fila corresponde a un nuevo paciente. Los datos de genética HLA fueron recogidos por el Laboratorio de Genética Molecular del Servicio de Análisis Clínicos del Hospital Universitario Río-Hortega, Valladolid.

4. Discusión

Observando los resultados presentados en la **tabla 4**, podemos ver como el programa es capaz de llevar a cabo correctamente la clasificación del riesgo asociado a los genotipos siguiendo la información y directrices dadas por Núñez et al., (2018), permitiéndole excluir aquellos genotipos compatibles con la enfermedad de los que no presentan este riesgo.

La aplicación nos da además la posibilidad de guardar nuestro trabajo en archivos de fácil manejo para el usuario para el futuro tratamiento de la información obtenida, tanto en consultas de atención primaria como en consultas de atención especializada en aparato digestivo.

Con ello podemos confirmar el buen funcionamiento tanto de los cálculos de riesgo de enfermedad celíaca para un único genotipo (*CeliacRisk* y *CeliacRiskCalculator*) como de los cálculos de riesgo para la descendencia entre dos parentales (*InheritanceC*), además del correcto comportamiento de la interfaz de usuario de la aplicación.

La creación de este programa responde a la necesidad de agilizar y facilitar en análisis de la genética HLA en el diagnóstico de la enfermedad celíaca ya que los archivos producto del uso del programa pueden ayudar acelerando y simplificando en gran medida la realización del estudio genético. Como se comentaba en la introducción del trabajo, el diagnóstico de la enfermedad celíaca puede tornarse en ocasiones muy complejo, y en estos casos el estudio genético puede hacerse muy útil en ciertas situaciones (Grupo de trabajo del Protocolo para el diagnóstico precoz de la enfermedad celíaca, 2018):

- En pacientes de diagnóstico incierto con sospecha clínica fundada, ya que la existencia de riesgo genético puede llevar a considerar la necesidad de realizar una biopsia de intestino delgado.
- Para facilitar y agilizar la identificación de individuos de riesgo considerable entre familiares de primer grado y pacientes con enfermedades asociadas con biopsias normales. En aquellos pacientes con genética positiva, es necesario llevar a cabo un seguimiento clínico y analítico periódico, al existir la posibilidad de que desarrollen la enfermedad celíaca con posterioridad.
- En aquellos casos sin respuesta a la retirada del gluten, para descartar diagnósticos erróneos y plantear nuevas hipótesis.

El estudio genético puede llegar a ser muy útil también en aquellos casos marcados por la ESPGHAN (Husby et al., 2012):

- Para realizar el diagnóstico sin necesidad de biopsias intestinales en niños con sospecha clínica fundada y altos niveles de anticuerpos IgA anti-transglutaminasa 2.
- Como prueba inicial de exclusión en niños asintomáticos. En caso de presentar haplotipos de riesgo, podría solicitarse el estudio serológico, y si este fuera positivo, la biopsia duodenal, pero en caso de no presentar haplotipos de riesgo, se puede descartar el diagnóstico de la enfermedad.

De cara al futuro de esta aplicación, el objetivo principal sería la adición continuada de nuevas enfermedades autoinmunes de base genética para su análisis en posteriores versiones de la misma, ya que con ello aumentaríamos

progresivamente el número de usuarios (ya que especialistas en diferentes afecciones podrían verse interesados) y la variedad de información presentada por el programa.

Otra posible mejora, al nivel de la enfermedad celíaca, podría ser la inclusión de nuevos alelos y moléculas de riesgo a medida que se vaya conociendo más información acerca del riesgo de padecer enfermedad celíaca asociado a la genética, tanto fuera como dentro de la región genómica HLA.

Por otro lado, es importante también comentar aquellas características que no han sido añadidas por cuestiones de tiempo, pero que constituirían algunas de las principales mejoras a llevar a cabo en el programa.

La primera de ellas consistiría en hacer al programa capaz de analizar simultáneamente una lista dada de genotipos. El hecho de poder utilizar como datos de entrada un archivo de Excel (con una estructura parecida a la de la **tabla 4**) con los genotipos de un gran número de pacientes para su computación secuencial no llevaría largos períodos de tiempo de espera y permitiría llevar a cabo investigaciones más amplias.

En segundo lugar, sería posible incluir para la clasificación de riesgos, una escala que variase sólo en el rango de 0 a 1 (es decir, hacer una clasificación normalizada) ya que esto podría ser muy útil para establecer un método único de clasificación para todas las enfermedades que se fueran añadiendo.

En tercer lugar, estaría añadir la posibilidad de guardar los archivos de resultados en diferentes formatos aparte de los propios de Microsoft Excel® y Microsoft Word®. Esto permitiría trabajar más fácilmente a usuarios de sistemas operativos diferentes de Windows.

Finalmente, el objetivo final del desarrollo del programa sería convertirlo a una aplicación independiente de iniciar R previo a su uso, de nuevo con la idea de hacerlo más fácil para el manejo por parte del usuario final.

5. Conclusiones

Con este trabajo hemos generado una herramienta de ayuda diagnóstica en el lenguaje de programación R, capaz de darnos información acerca del riesgo de enfermedad celíaca asociado al genotipo de un paciente o en la hipotética descendencia entre dos parentales.

Haciendo uso de dicha información, es posible conocer si el genotipo es o no compatible con sufrir la afección, lo que podría ahorrar costes y agilizar trabajos de análisis genético dirigidos a la diagnosis de enfermedad celíaca, tanto en aquellos pacientes cuyo genotipo no sea compatible con este cuadro clínico (en los cuales se descartaría la enfermedad), como en los pacientes con genotipos de riesgo, en los que el análisis genético puede servir para identificar familiares de riesgo, o para plantear nuevos análisis en pacientes de diagnóstico incierto o erróneo.

Esta aplicación, además, pretende ser un modelo para el desarrollo de otros programas para ayudar en el diagnóstico de otras enfermedades autoinmunes de base genética, como la diabetes tipo I, la artritis reumatoide o el lupus eritematoso sistémico, donde la genética HLA constituye un punto clave a analizar.

6. Bibliografía

Chagoyen M., Pazos F. (2016). "Characterization of clinical signs in the human interactome". *Bioinformatics*, 32 (12), pp 1761-1765.

Dragulescu. A.A., (2014). "xlsx: Read, write, format Excel 2007 and Excel 97/2000/XP/2003 files." <https://CRAN.R-project.org/package=xlsx>

Fernández-Cavada-Pollo, M.J., Alcalá-Peña, M.I., Vargas-Pérez, M.L, Vergara-Prieto, E., Vallcorba-Gómez-Del-Valle, I., Melero-Ruiz, J., ... González-Roiz, C., (2013). "Celiac Disease and HLA-DQ genotype: Diagnosis of different genetic risk profiles related to age in Badajoz, southwestern Spain". *Revista Española de Enfermedades Digestivas*, 105 (8), pp 469-476.

Gohel, D. (2018). "ReporteRs: Microsoft Word and PowerPoint Documents Generation". R package version 0.8.10.
<https://CRAN.R-project.org/package=ReporteRs>

Grupo de trabajo del Protocolo para el diagnóstico precoz de la enfermedad celíaca (2018). "Protocolo para el diagnóstico precoz de la enfermedad celíaca". Ministerio de Sanidad, Servicios Sociales e Igualdad. Servicio de Evaluación del Servicio Canario de la Salud (SESCS).

Husby, S., Koletzko, S., Korponay-Szabó, I.R., Mearin, M.L, Phillips, A., Shamir, R et al. European Society for Pediatric Gastroenterology, Hepatology and Nutrition (2012), "Guidelines for the diagnosis of coeliac disease". *Journal of Pediatric Gastroenterology and Nutrition*, 54:136-60

Köhler, S., Vasilevsky N., Engelstad, M., Foster, E., McMurry, J., Aymé, S., ... Robinson, P. N. (2017). "The Human Phenotype Ontology in 2017". *Nucleic Acids Research*, 45 (D1), pp 865-876.

Köhler, S., Schulz M.H., Krawitz, P., Bauer S., Dölken, S., Ott, C.E., Robinson, P.N. (2009). "Clinical diagnostics in human genetics with semantic similarity searches in ontologies". *American journal of human genetics*, 85, pp 457-464

Lawrence, M., Lang., D.T. (2010). "RGtk2: A Graphical User Interface Toolkit for R". *Journal of Statistical Software*, 37(8), pp 1-52.
URL: <http://www.jstatsoft.org/v37/i08>

McKusick-Nathans Institute of Genetic Medicine, Johns Hopkins University (2017). "Online Mendelian Inheritance in Man, OMIM[®]". <https://omim.org/>

Menche, J., Sharma, A., Kitsak, M., Ghiassian, S.D., Vidal, M., Loscalzo, J., Barabási, A.L. (2015). "Disease networks. Uncovering disease-disease relationships through the incomplete interactome". *Science*, 347(6224), 1257601.

Núñez, C., Garrote, J.A., Arranz, E., Bilbao, J.R., Fernández-Bañares, F., Jiménez, J., Perucho, T., Ruiz-Casares, E., Sánchez-Valverde, F., Serrano, N. (2018). "Recommendations to report and interpret HLA findings in coeliac disease". *Revista Española de Enfermedades Digestivas*. doi: 10.17235/reed.2018.5269/2017.

Python Software Foundation (2018). Python Language Reference, version 3.6.5. Disponible en <http://www.python.org>

R Core Team (2018). R: A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria. URL <https://www.R-project.org/>.

Tcl Core Team, (2017). Tcl/tk versión 8.6.8. Disponible en <https://www.tcl.tk/software/tcltk/8.6.html>

The Gene Ontology Consortium (2015). "Gene Ontology Consortium: going forward". *Nucleic Acids Research*, 43(D1), pp 1049-1056.

The GNOME Project (2018). GNOME versión 3.28. Disponible en <https://help.gnome.org/misc/release-notes/3.28/>

Urbanek, S. (2017). "rJava: Low-Level R to Java Interface." R package version 0.9-9. <https://CRAN.R-project.org/package=rJava>

Verzani, J. (2016). "gWidgets2: Rewrite of gWidgets API for Simplified GUI Construction". R package version 1.0-7. <https://CRAN.Rproject.org/package=gWidgets2>

Anexos. Código de las funciones del programa *CeliacDQ*.

Anexo I. *CeliacRisk*

```
1. CeliacRisk <- function(a1,a2,b1,b2,verbose=T) {
2.   h1adq1 <- paste(a1,b1,sep="")
3.   h1adq2 <- paste(a1,b2,sep="")
4.   h1adq3 <- paste(a2,b1,sep="")
5.   h1adq4 <- paste(a2,b2,sep="")
6.   DQrisks <- read.table(file="C:/Users/iker/Desktop/Cosas
R/CELIAQUÍA/DQrisks.csv", sep=";")
7.   DQrisks[,1] <- as.character(DQrisks[,1])
8.   DQrisks[,3] <- as.character(DQrisks[,3])
9.   #DQrisks <- get("DQrisks",envir="CeliacEnv")
10.  rownames(DQrisks) <- DQrisks[,1]
11.  us1 <- DQrisks[h1adq1,3]
12.  if(is.na(us1)){
13.    us1 <- "Non Risk Molecule"
14.  }
15.  us2 <- DQrisks[h1adq2,3]
16.  if(is.na(us2)){
17.    us2 <- "Non Risk Molecule"
18.  }
19.  us3 <- DQrisks[h1adq3,3]
20.  if(is.na(us3)){
21.    us3 <- "Non Risk Molecule"
22.  }
23.  us4 <- DQrisks[h1adq4,3]
24.  if(is.na(us4)){
25.    us4 <- "Non Risk Molecule"
26.  }
27.  if(verbose==T) {
28.    message(paste0("Possible HLADQs are: HLA",us1,"
HLA",us2," HLA",us3," HLA",us4))
29.    Sys.sleep(0.5)
30.    message(paste0("Calculating risk for HLADQs..."))
31.  }
32.
33.  h1adqs <- c(h1adq1,h1adq2,h1adq3,h1adq4)
34.  values <- matrix(data=0, nrow = 4, ncol= 1)
35.  x= 0
36.  repeat{
37.    x= x+1
38.    if((x==5)==TRUE){
39.      break()
40.    }
41.    cosa <- h1adqs[x]
42.    if((cosa %in% rownames(DQrisks))==FALSE){
43.      values[x,1] <- 0
44.    }
45.    if((cosa %in% rownames(DQrisks))==TRUE){
46.      values[x,1] <- DQrisks[cosa,2]
47.    }
48.  }
49.  ker <- sum(values[1:4,1])
50.  kir <- NA
51.  if(ker > 1.074){
52.    kir<- "Very High (Rg > 1.074):"
53.  }
54.  if(ker <= 1.074){
55.    if(ker >= 0.2){
```



```

56.         kir<- "High (1.074 >= Rg >= 0.2):"
57.     }
58.     if(ker < 0.2){
59.         if(ker >= 0.064){
60.             kir<- "Medium (0.2 > Rg >= 0.064):"
61.         }
62.         if(ker < 0.064){
63.             kir <- "Low (0.064 > Rg):"
64.         }
65.     }
66. }
67. if(ker==0){
68.     kir <- "No Risk, Rg = 0"
69. }
70. #####This if bucle is for the DQ2.5/DQ7.5 case, which has
a trans DQ2.5 that lead to Very
71. ##### high group, while being part of high group.
72. if(ker==2.02){
73.     message("DQ2.5/DQ7.5 is high risk, converting...")
74.     Sys.sleep(0.3)
75.     ker <- 1.07
76.     kir<- "High (1.074 >= Rg > 0.2):"
77. }
78. message(paste("Celiac Risk (Rg) based on HLA DQ genotype
is",kir,sep = " "))
79. return(ker)
80. }

```

Anexo II. CeliacRiskClasification

```

1. CeliacRiskclasification <- function(a1,a2,b1,b2,verbose=T) {
2.   h1adq1 <- paste(a1,b1,sep="")
3.   h1adq2 <- paste(a1,b2,sep="")
4.   h1adq3 <- paste(a2,b1,sep="")
5.   h1adq4 <- paste(a2,b2,sep="")
6.   DQrisks <- read.table(file="C:/Users/iker/Desktop/Cosas
R/CELIAQUA/DQrisks.csv",sep=";")
7.   DQrisks[,1] <- as.character(DQrisks[,1])
8.   DQrisks[,3] <- as.character(DQrisks[,3])
9.   rownames(DQrisks) <- DQrisks[,1]
10.  #DQrisks <- get("DQrisks",envir="CeliacEnv")
11.  us1 <- DQrisks[h1adq1,3]
12.  if(is.na(us1)){
13.      us1 <- "Non Risk Molecule"
14.  }
15.  us2 <- DQrisks[h1adq2,3]
16.  if(is.na(us2)){
17.      us2 <- "Non Risk Molecule"
18.  }
19.  us3 <- DQrisks[h1adq3,3]
20.  if(is.na(us3)){
21.      us3 <- "Non Risk Molecule"
22.  }
23.  us4 <- DQrisks[h1adq4,3]
24.  if(is.na(us4)){
25.      us4 <- "Non Risk Molecule"
26.  }
27.  if(verbose==T) {
28.      message(paste0("Possible HLA DQs are: HLA",us1,"
HLA",us2," HLA",us3," HLA",us4))

```

```

29.     Sys.sleep(0.5)
30.     message(paste0("Calculating risk for HLA DQs..."))
31.   }
32.
33.   hladqs <- c(hladq1,hladq2,hladq3,hladq4)
34.   values <- matrix(data=0, nrow = 4, ncol= 1)
35.   x= 0
36.   repeat{
37.     x= x+1
38.     if((x==5)==TRUE){
39.       break()
40.     }
41.     cosa <- hladqs[x]
42.     if((cosa %in% rownames(DQrisks))==FALSE){
43.       values[x,1] <- 0
44.     }
45.     if((cosa %in% rownames(DQrisks))==TRUE){
46.       values[x,1] <- DQrisks[cosa,2]
47.     }
48.   }
49.   ker <- sum(values[1:4,1])
50.   kir <- NA
51.   if(ker > 1.074){
52.     kir<- "Very High (Rg > 1.074):"
53.   }
54.   if(ker <= 1.074){
55.     if(ker >= 0.2){
56.       kir<- "High (1.074 >= Rg >= 0.2):"
57.     }
58.     if(ker < 0.2){
59.       if(ker >= 0.064){
60.         kir<- "Medium (0.2 > Rg >= 0.064):"
61.       }
62.       if(ker < 0.064){
63.         kir <- "Low (0.064 > Rg):"
64.       }
65.     }
66.   }
67.   if(ker==0){
68.     kir <- "No Risk (Rg = 0):"
69.   }
70.   #####This if bucle is for the DQ2.5/DQ7.5 case, which has
a trans DQ2.5 that lead to Very
71.   ##### high group, while being part of high group.
72.   if(ker==2.02){
73.     message("DQ2.5/DQ7.5 is high risk, converting...")
74.     Sys.sleep(0.3)
75.     ker <- 1.07
76.     kir<- "High (1.074 >= Rg > 0.2):"
77.   }
78.   message(paste("Celiac Risk (Rg) based on HLA DQ genotype
is",kir,sep = " "))
79.   print(ker)
80.   kir <- gsub(":", "", kir)
81.   return(kir)
82. }

```

Anexo III. InheritanceC

```
1. InheritanceC <-
  function (ap1, ap2, am1, am2, bp1, bp2, bm1, bm2, mode= "Math", verbose) {
2.   uno <- paste (ap1, bp1)
3.   dos <- paste (ap1, bp2)
4.   tres <- paste (ap2, bp1)
5.   cuatro <- paste (ap2, bp2)
6.   cinco <- paste (am1, bm1)
7.   seis <- paste (am1, bm2)
8.   siete <- paste (am2, bm1)
9.   ocho <- paste (am2, bm2)
10.  rowan <- c (uno, dos, tres, cuatro)
11.  rowan <- as.list (rowan)
12.  harper <- c (cinco, seis, siete, ocho)
13.  harper <- as.list (harper)
14.  tablavalores <- matrix (data = 0, ncol= 4, nrow= 4)
15.  rownames (tablavalores) <- rowan
16.  colnames (tablavalores) <- harper
17.  if (mode=="Math"){ #####Calcular el valor numérico
18.    (tablavalores[1,1] <-
      CeliacRisk (a1=ap1, a2=am1, b1=bp1, b2=bm1, verbose)) #1357
19.    print (tablavalores[1,1])
20.    (tablavalores[1,2] <-
      CeliacRisk (a1=ap1, a2=am1, b1=bp1, b2=bm2, verbose)) #1358
21.    print (tablavalores[1,2])
22.    (tablavalores[1,3] <-
      CeliacRisk (a1=ap1, a2=am2, b1=bp1, b2=bm1, verbose)) #1367
23.    print (tablavalores[1,3])
24.    (tablavalores[1,4] <-
      CeliacRisk (a1=ap1, a2=am2, b1=bp1, b2=bm2, verbose)) #1368
25.    print (tablavalores[1,4])
26.    (tablavalores[2,1] <-
      CeliacRisk (a1=ap1, a2=am1, b1=bp2, b2=bm1, verbose)) #1457
27.    print (tablavalores[2,1])
28.    (tablavalores[2,2] <-
      CeliacRisk (a1=ap1, a2=am1, b1=bp2, b2=bm2, verbose)) #1458
29.    print (tablavalores[2,2])
30.    (tablavalores[2,3] <-
      CeliacRisk (a1=ap1, a2=am2, b1=bp2, b2=bm1, verbose)) #1467
31.    print (tablavalores[2,3])
32.    (tablavalores[2,4] <-
      CeliacRisk (a1=ap1, a2=am2, b1=bp2, b2=bm2, verbose)) #1468
33.    print (tablavalores[2,4])
34.    (tablavalores[3,1] <-
      CeliacRisk (a1=ap2, a2=am1, b1=bp1, b2=bm1, verbose)) #2357
35.    print (tablavalores[3,1])
36.    (tablavalores[3,2] <-
      CeliacRisk (a1=ap2, a2=am1, b1=bp1, b2=bm2, verbose)) #2358
37.    print (tablavalores[3,2])
38.    (tablavalores[3,3] <-
      CeliacRisk (a1=ap2, a2=am2, b1=bp1, b2=bm1, verbose)) #2367
39.    print (tablavalores[3,3])
40.    (tablavalores[3,4] <-
      CeliacRisk (a1=ap2, a2=am2, b1=bp1, b2=bm2, verbose)) #2368
41.    print (tablavalores[3,4])
42.    (tablavalores[4,1] <-
      CeliacRisk (a1=ap2, a2=am1, b1=bp2, b2=bm1, verbose)) #2457
43.    print (tablavalores[4,1])
44.    (tablavalores[4,2] <-
      CeliacRisk (a1=ap2, a2=am1, b1=bp2, b2=bm2, verbose)) #2458
```

```

45.     print(tablavalores[4,2])
46.     (tablavalores[4,3] <-
CeliacRisk(a1=ap2,a2=am2,b1=bp2,b2=bm1,verbose)) #2467
47.     print(tablavalores[4,3])
48.     (tablavalores[4,4] <-
CeliacRisk(a1=ap2,a2=am2,b1=bp2,b2=bm2,verbose)) #2468
49.     print(tablavalores[4,4])
50.     print(tablavalores)
51.     return(tablavalores)
52. }
53.     else if(mode=="Class"){ #####Calcular solo el grupo al
que pertenece
54.     (tablavalores[1,1] <-
CeliacRiskclasificacion(a1=ap1,a2=am1,b1=bp1,b2=bm1,verbose)) #
1357
55.     print(tablavalores[1,1])
56.     (tablavalores[1,2] <-
CeliacRiskclasificacion(a1=ap1,a2=am1,b1=bp1,b2=bm2,verbose)) #
1358
57.     print(tablavalores[1,2])
58.     (tablavalores[1,3] <-
CeliacRiskclasificacion(a1=ap1,a2=am2,b1=bp1,b2=bm1,verbose)) #
1367
59.     print(tablavalores[1,3])
60.     (tablavalores[1,4] <-
CeliacRiskclasificacion(a1=ap1,a2=am2,b1=bp1,b2=bm2,verbose)) #
1368
61.     print(tablavalores[1,4])
62.     (tablavalores[2,1] <-
CeliacRiskclasificacion(a1=ap1,a2=am1,b1=bp2,b2=bm1,verbose)) #
1457
63.     print(tablavalores[2,1])
64.     (tablavalores[2,2] <-
CeliacRiskclasificacion(a1=ap1,a2=am1,b1=bp2,b2=bm2,verbose)) #
1458
65.     print(tablavalores[2,2])
66.     (tablavalores[2,3] <-
CeliacRiskclasificacion(a1=ap1,a2=am2,b1=bp2,b2=bm1,verbose)) #
1467
67.     print(tablavalores[2,3])
68.     (tablavalores[2,4] <-
CeliacRiskclasificacion(a1=ap1,a2=am2,b1=bp2,b2=bm2,verbose)) #
1468
69.     print(tablavalores[2,4])
70.     (tablavalores[3,1] <-
CeliacRiskclasificacion(a1=ap2,a2=am1,b1=bp1,b2=bm1,verbose)) #
2357
71.     print(tablavalores[3,1])
72.     (tablavalores[3,2] <-
CeliacRiskclasificacion(a1=ap2,a2=am1,b1=bp1,b2=bm2,verbose)) #
2358
73.     print(tablavalores[3,2])
74.     (tablavalores[3,3] <-
CeliacRiskclasificacion(a1=ap2,a2=am2,b1=bp1,b2=bm1,verbose)) #
2367
75.     print(tablavalores[3,3])
76.     (tablavalores[3,4] <-
CeliacRiskclasificacion(a1=ap2,a2=am2,b1=bp1,b2=bm2,verbose)) #
2368
77.     print(tablavalores[3,4])

```

```

78.         (tablavalores[4,1] <-
  CeliacRiskclasificacion(a1=ap2,a2=am1,b1=bp2,b2=bm1,verbose)) #
2457
79.         print(tablavalores[4,1])
80.         (tablavalores[4,2] <-
  CeliacRiskclasificacion(a1=ap2,a2=am1,b1=bp2,b2=bm2,verbose)) #
2458
81.         print(tablavalores[4,2])
82.         (tablavalores[4,3] <-
  CeliacRiskclasificacion(a1=ap2,a2=am2,b1=bp2,b2=bm1,verbose)) #
2467
83.         print(tablavalores[4,3])
84.         (tablavalores[4,4] <-
  CeliacRiskclasificacion(a1=ap2,a2=am2,b1=bp2,b2=bm2,verbose)) #
2468
85.         print(tablavalores[4,4])
86.         print(tablavalores)
87.         return(tablavalores)
88.     }
89.     else stop(paste("Unknown mode",mode)) ###Modo
desconocido?
90.     }

```

Anexo IV. HLADQCalculator

```

1. HLADQCalculator <- function() {
2.   library ("gWidgets2", lib.loc=~R/win-library/3.5")
3.   library ("gWidgets2RGtk2", lib.loc=~R/win-library/3.5")
4.   library ("gWidgets2tcltk", lib.loc=~R/win-library/3.5")
5.   library ("xlsx", lib.loc=~R/win-library/3.5")
6.   win <- gwindow(title="CeliacDQ")
7.   group <- ggroup(horizontal = FALSE,container= win)
8.   glados <- glabel(text= " ",container= group,editable = FALSE)
9.   labelillo <- glabel(text= "Select mode:",container= group)
10.  glados1 <- glabel(text= "
",container= group,editable = FALSE)
11.  P1button <- gbutton("Calculate Celiac Risk for 1
genotype",handler = function(h,...){
12.    CeliacRiskCalculator()},container= group)
13.  glados2 <- glabel(text= "
",container= group,editable = FALSE)
14.  P2button <- gbutton("Calculate Celiac Risks for
inheritance between 2 parents",handler = function(h,...){
15.    InheritanceDQCalculator()
16.  },container=group)
17.
18.  }

```

Anexo V. CeliacRiskCalculator

```

1. CeliacRiskCalculator <- function() {
2.   library ("gWidgets2", lib.loc=~R/win-library/3.5")
3.   library ("gWidgets2RGtk2", lib.loc=~R/win-library/3.5")
4.   library ("gWidgets2tcltk", lib.loc=~R/win-library/3.5")
5.   library ("xlsx", lib.loc=~R/win-library/3.5")
6.   library ("ReporteRs", lib.loc=~R/win-library/3.5")
7.   win <- gwindow(title="Celiac Risk Calculator 1.0 Beta")
8.   group <- ggroup(horizontal=FALSE, container=win)
9.   alelosa <- c("A1*05", "A1*02", "A1*03", "No risk allele or
unknown allele")

```

```

10.     alelosb <- c("B1*02", "B1*03:01", "B1*03:02", "No risk
allele or unknown allele")
11.     glados5 <- glabel(text= "
", container= group, editable = FALSE)
12.     labelGenA1 <- glabel(text="HLA-DQA allele
1", container = group)
13.     comboboxGenA1 <-
gcombobox(items= alelosa, selected=1, editable=FALSE, container=gr
oup,
14.                                     handler=function(h, ...){
15.     svalue(comboboxGenA1, index= TRUE)
dox <-
16.     dox <- as.numeric(dox)
17.     if(dox==1){
18.     cu<- "A1*05"}
19.     else if(dox>1){
20.     if(dox==2){
21.     cu<- "A1*02"}
22.     else if(dox>2){
23.     if(dox==3){
24.     cu<- "A1*03"
25.     }
26.     else if(dox==4){
27.     cu<- "NonRisk"
28.     }
29.     }
30.     }
31.     message("a1 set to ", cu)
32.     }
33.     )
34.     labelGenA2 <- glabel(text="HLA-DQA allele
2", container = group)
35.     comboboxGenA2 <-
gcombobox(items= alelosa, selected=1, editable=FALSE, container=gr
oup,
36.                                     handler=function(h, ...){
37.     svalue(comboboxGenA2, index= TRUE)
dox2 <-
38.     dox2 <- as.numeric(dox2)
39.     if(dox2==1){
40.     cu2<- "A1*05"}
41.     else if(dox2>1){
42.     if(dox2==2){
43.     cu2<- "A1*02"}
44.     else if(dox2>2){
45.     if(dox2==3){
46.     cu2<- "A1*03"
47.     }
48.     else if(dox2==4){
49.     cu2<- "NonRisk"
50.     }
51.     }
52.     }
53.     message("a2 set to ", cu2)
54.     }
55.     )
56.     labelGenB1 <- glabel(text="HLA-DQB allele
1", container = group)
57.     comboboxGenB1 <-
gcombobox(items= alelosb, selected=1, editable=FALSE, container=gr
oup,

```

```

58.                                     handler=function(h,...){
59.                                     dox5 <-
svalue (comboboxGenB1,index= TRUE)
60.                                     dox5 <- as.numeric(dox5)
61.                                     if(dox5==1){
62.                                     cu5<- "B1*02"}
63.                                     else if(dox5>1){
64.                                     if(dox5==2){
65.                                     cu5<- "B1*03:01"}
66.                                     else if(dox5>2){
67.                                     if(dox5==3){
68.                                     cu5<- "B1*03:02"
69.                                     }
70.                                     else if(dox5==4){
71.                                     cu5<- "NonRisk"
72.                                     }
73.                                     }
74.                                     }
75.                                     message("b1 set to ",cu5)
76.                                     }
77.                                     )
78.                                     labelGenB2 <- glabel(text="HLA-DQB allele
2",container = group)
79.                                     comboboxGenB2 <-
gcombobox(items= alelosb,selected=1,editable=FALSE,container=gr
oup,
80.                                     handler=function(h,...){
81.                                     dox6 <-
svalue (comboboxGenB2,index= TRUE)
82.                                     dox6 <- as.numeric(dox6)
83.                                     if(dox6==1){
84.                                     cu6<- "B1*02"}
85.                                     else if(dox6>1){
86.                                     if(dox6==2){
87.                                     cu6<- "B1*03:01"}
88.                                     else if(dox6>2){
89.                                     if(dox6==3){
90.                                     cu6<- "B1*03:02"
91.                                     }
92.                                     else if(dox6==4){
93.                                     cu6<- "NonRisk"
94.                                     }
95.                                     }
96.                                     }
97.                                     message("b2 set to ",cu6)
98.                                     }
99.                                     )
100.                                    glados6 <- glabel(text= "
",container= group,editable = FALSE)
101.                                    MathButton <- gbutton("Calculate Celiac Risk",
container=group,
102.                                    handler=function(h, ...) {
103.                                    dox <-
svalue (comboboxGenA1,index= TRUE)
104.                                    dox <- as.numeric(dox)
105.                                    if(dox==1){
106.                                    cu<- "A1*05"}
107.                                    else if(dox>1){
108.                                    if(dox==2){
109.                                    cu<- "A1*02"}
110.                                    else if(dox>2){

```

```

111.         if (dox==3) {
112.             cu<- "A1*03"
113.         }
114.         else if (dox==4) {
115.             cu<- "Non Risk allele"
116.         }
117.     }
118. }
119.
120.     dox2 <-
svalue (comboboxGenA2, index= TRUE)
121.     dox2 <- as.numeric (dox2)
122.     if (dox2==1) {
123.         cu2<- "A1*05" }
124.     else if (dox2>1) {
125.         if (dox2==2) {
126.             cu2<- "A1*02" }
127.         else if (dox2>2) {
128.             if (dox2==3) {
129.                 cu2<- "A1*03"
130.             }
131.             else if (dox2==4) {
132.                 cu2<- "Non Risk allele"
133.             }
134.         }
135.     }
136.     dox5 <-
svalue (comboboxGenB1, index= TRUE)
137.     dox5 <- as.numeric (dox5)
138.     if (dox5==1) {
139.         cu5<- "B1*02" }
140.     else if (dox5>1) {
141.         if (dox5==2) {
142.             cu5<- "B1*03:01" }
143.         else if (dox5>2) {
144.             if (dox5==3) {
145.                 cu5<- "B1*03:02"
146.             }
147.             else if (dox5==4) {
148.                 cu5<- "Non Risk allele"
149.             }
150.         }
151.     }
152.     dox6 <-
svalue (comboboxGenB2, index= TRUE)
153.     dox6 <- as.numeric (dox6)
154.     if (dox6==1) {
155.         cu6<- "B1*02" }
156.     else if (dox6>1) {
157.         if (dox6==2) {
158.             cu6<- "B1*03:01" }
159.         else if (dox6>2) {
160.             if (dox6==3) {
161.                 cu6<- "B1*03:02"
162.             }
163.             else if (dox6==4) {
164.                 cu6<- "Non Risk allele"
165.             }
166.         }
167.     }
168.     hldq1 <- paste (cu, cu5, sep="")

```



```

169.             hladq2 <- paste(cu, cu6, sep="")
170.             hladq3 <- paste(cu2, cu5, sep="")
171.             hladq4 <- paste(cu2, cu6, sep="")
172.             data3 <-
CeliacRisk(a1= cu, a2 = cu2, b1= cu5, b2= cu6)
173.             data4 <-
CeliacRiskclasification(cu, a2 = cu2, b1= cu5, b2= cu6)
174.             DQrisks <-
read.table(file="C:/Users/iker/Desktop/Cosas
R/CELIAQUÍA/DQrisks.csv", sep=";")
175.             DQrisks[,1] <-
as.character(DQrisks[,1])
176.             DQrisks[,3] <-
as.character(DQrisks[,3])
177.             #DQrisks <-
get("DQrisks",envir="CeliacEnv")
178.             rownames(DQrisks) <- DQrisks[,1]
179.             us1 <- DQrisks[hladq1,3]
180.             if(is.na(us1)){
181.                 us1 <- "Non Risk Molecule"
182.             }
183.             us2 <- DQrisks[hladq2,3]
184.             if(is.na(us2)){
185.                 us2 <- "Non Risk Molecule"
186.             }
187.             us3 <- DQrisks[hladq3,3]
188.             if(is.na(us3)){
189.                 us3 <- "Non Risk Molecule"
190.             }
191.             us4 <- DQrisks[hladq4,3]
192.             if(is.na(us4)){
193.                 us4 <- "Non Risk Molecule"
194.             }
195.             Results <-
gwindow(title="Results")
196.             grupie <-
ggroup(horizontal=FALSE, container=Results)
197.             space7 <- spacel <- glabel(text="
", container = grupie, editable = FALSE)
198.             Resultlabel <-
glabel(text=paste("Celiac Risk (Rg) based on HLAQ genotype
is", data3, ", ", data4), container = grupie, editable = FALSE)
199.             patientgenotype2 <-
paste("Patient HLAQ Alleles
are:", cu, ", ", cu2, ", ", cu5, "and", cu6)
200.             spacel <- glabel(text="
", container = grupie, editable = FALSE)
201.             Resultlabel2 <-
glabel(text=patientgenotype2, container = grupie, editable = FALSE)
202.             space2 <- glabel(text="
", container = grupie, editable = FALSE)
203.             possiblehla2 <- paste("Patient
possible HLAQ molecules are:", us1, ", ", us2, ", ", us3, "and", us4)
204.             Resultlabel3 <-
glabel(text=possiblehla2, container = grupie, editable = FALSE)
205.             space3 <- glabel(text="
", container = grupie, editable = FALSE)
206.             botoninforme <- gbutton("Generate
patient Celiac Risk
report", container=grupie, handler = function(h, ...){

```

```

207. informewindow <-
  gwindow(title = "Report Creator")
208. grupo <-
  ggroup(horizontal = FALSE,container=informewindow)
209. space8 <- glabel(text="
",container = grupo,editable = FALSE)
210. nombrenoedit <-
  glabel(text = "Type patient surname and name below, and press
Enter",editable = FALSE,container = grupo)
211. space4 <- glabel(text="
",container = grupo,editable = FALSE)
212. nombreedit <-
  glabel(text = "Click here to type patient
name",editable = TRUE,container = grupo)
213. space5 <- glabel(text="
",container = grupo,editable = FALSE)
214. nombrenoedit2 <-
  glabel(text = "Type physician (user) surname and name below,
and press Enter",editable = FALSE,container = grupo)
215. space71 <- glabel(text="
",container = grupo,editable = FALSE)
216. nombreedit2 <-
  glabel(text = "Click here to type physician (user)
name",editable = TRUE,container = grupo)
217. space82<- glabel(text="
",container = grupo,editable = FALSE)
218. cuniha <- c("Man", "Woman")
219. sexelect <-
  glabel(text = "Patient's
sex:",editable = TRUE,container = grupo)
220. selectsex <-
  gcombobox(items= cuniha,selected = 1,editable = FALSE,container
= grupo,handler=function(h,...){
221. tsubasa <- svalue(selectsex)
222. message("sex set to
",tsubasa)
223. })
224. space821<- glabel(text="
",container = grupo,editable = FALSE)
225. space825<- glabel(text="Click
<<open>> to select saving path and then click
save",container = grupo,editable = FALSE)
226. saverino <-
  gfilebrowse(text = "Path",type = "save",initial.filename = "Unt
itled1.xlsx", container = grupo)
227. space6 <- glabel(text="
",container = grupo,editable = FALSE)
228. botonguardado <- gbutton("Save
report",container = grupo, handler= function(h,...){
229. tsubasa <- svalue(selectsex)
230. report <- docx()
231. title <- paste("Report:
Genetic Markers for celiac disease (CELIAC1, #OMIM:212750) ")
232. report <- addTitle(report,
value = title, level=1)
233. report<-
  addParagraph(report,value= " " )
234. paste("Responsible physician:")
  namedovbold <-
235. paste(svalue(nombreedit2))
  namedov <-

```

```

236.         report<-
      addParagraph(report,value= paste(namedovbold,namedov),level=2)
237.         report<-
      addParagraph(report,value= " " )
238.         namebold <- paste("Patient's
      name:")
239.         name <-
      paste(svalue(nombreedit))
240.         report <-
      addParagraph(report, value= paste(namebold,name),level= 2)
241.         report<-
      addParagraph(report,value= " " )
242.         patientsex <-
      paste("Patient's sex:",tsubasa)
243.         report<-
      addParagraph(report,value= patientsex)
244.         report<-
      addParagraph(report,value= " " )
245.         patientgenotypebold <-
      paste("Patient's HLADQ Alleles:")
246.         patientgenotype <-
      paste(cu, ",", cu2, ",", cu5, "and", cu6)
247.         report <-
      addParagraph(report,
      value= paste(patientgenotypebold,patientgenotype),level=2)
248.         report<-
      addParagraph(report,value= " " )
249.         possiblehlabold <-
      paste("Patient's 4 possible HLADQ combination molecules:")
250.         possiblehla <-
      paste(us1, ",", us2, ",", us3, "and", us4)
251.         report <-
      addParagraph(report, value= paste(possiblehlabold,possiblehla),
      level=2)
252.         report<-
      addParagraph(report,value= " " )
253.         riskybold <- paste("Celiac
      Risk (Rg) value:")
254.         risky <-
      paste(riskybold,data3)
255.         riskyboldtype <-
      paste("Celiac Risk (Rg) classification:")
256.         riskytype <-
      paste(riskyboldtype,data4)
257.         report <-
      addParagraph(report, value= risky,level=2)
258.         report<-
      addParagraph(report,value= " " )
259.         report <-
      addParagraph(report, value= riskytype,level=2)
260.         report<-
      addParagraph(report,value= " " )
261.         report <-
      addParagraph(report, value= "Interpretation:",level=2)
262.         if(data3 > 1.074){
263.           report <-
      addParagraph(report, value= "Patient presents a very high risk
      genotype, compatible with celiac disease",level=3)
264.         }
265.         if(data3 <= 1.074){
266.           if(data3 > 0.2){

```

```

267.         report <-
      addParagraph(report, value= "Patient presents a high risk
      genotype, compatible with celiac disease",level=3)
268.     }
269.     if(data3 <= 0.2){
270.         if(data3 >= 0.064){
271.             report <-
      addParagraph(report, value= "Patient presents a medium risk
      genotype, compatible with celiac disease",level=3)
272.         }
273.         if(data3 < 0.064){
274.             report <-
      addParagraph(report, value= "Patient presents a low risk
      genotype, celiac disease can not be discarded",level=3)
275.         }
276.     }
277. }
278. if(data3== 0){
279.     report <-
      addParagraph(report, value= "Patient does not present a risk
      genotype, celiac disease can be discarded",level=3)
280. }
281. writeDoc(report, file=svalue(
saverino))
282. hlconfirmed <-
      gwindow(title = "Saved")
283. rebels <-
      ggroup(horizontal = FALSE,container=hlconfirmed)
284. combine <-
      glabel(text = paste("Word document report saved
      on",svalue(saverino)),container=hlconfirmed)
285.     })
286.     }
287.     )
288.     })
289. }

```

Anexo VI. InheritanceDQCalculator

```

1. InheritanceDQCalculator <- function(){
2.   library("gWidgets2", lib.loc=~R/win-library/3.5")
3.   library("gWidgets2RGtk2", lib.loc=~R/win-library/3.5")
4.   library("gWidgets2tcltk", lib.loc=~R/win-library/3.5")
5.   library("xlsx", lib.loc=~R/win-library/3.5")
6.   win <- gwindow(title="Inheritance DQ Calculator Beta")
7.   group <- ggroup(horizontal=FALSE, container=win)
8.   alelosa <- c("A1*05", "A1*02", "A1*03", "No risk allele or
      unknown allele")
9.   alelosb <- c("B1*02", "B1*03:01", "B1*03:02", "No risk allele or
      unknown allele")
10.    glados4 <- glabel(text= "
      ",container= group,editable = FALSE)
11.    labelmadreGenA1 <- glabel(text="Mother's HLA-DQA allele
      1",container = group)
12.    comboboxmadreGenA1 <-
      gcombobox(items= alelosa,selected=1,editable=FALSE,container=gr
      oup,
13.              handler=function(h,...){
14.                dox <-
      svalue(comboboxmadreGenA1,index= TRUE)
15.                dox <- as.numeric(dox)

```

```

16.         if(dox==1) {
17.             cu<- "A1*05"}
18.         else if(dox>1) {
19.             if(dox==2) {
20.                 cu<- "A1*02"}
21.             else if(dox>2) {
22.                 if(dox==3) {
23.                     cu<- "A1*03"
24.                 }
25.                 else if(dox==4) {
26.                     cu<- "NonRisk"
27.                 }
28.             }
29.         }
30.         message("am1 set to
", cu)
31.     }
32. )
33.     labelmadreGenA2 <- glabel(text="Mother's HLA-DQA allele
2",container = group)
34.     comboboxmadreGenA2 <-
gcombobox(items= alelosa,selected=1,editable=FALSE,container=gr
oup,
35.         handler=function(h,...) {
36.             dox2 <-
svalue(comboboxmadreGenA2,index= TRUE)
37.             dox2 <-
as.numeric(dox2)
38.             if(dox2==1) {
39.                 cu2<- "A1*05"}
40.             else if(dox2>1) {
41.                 if(dox2==2) {
42.                     cu2<- "A1*02"}
43.                 else if(dox2>2) {
44.                     if(dox2==3) {
45.                         cu2<- "A1*03"
46.                     }
47.                     else if(dox2==4) {
48.                         cu2<- "NonRisk"
49.                     }
50.                 }
51.             }
52.             message("am2 set to
", cu2)
53.         }
54.     )
55.     labelpadreGenA1 <- glabel(text="Father's HLA-DQA allele
1",container = group)
56.     comboboxpadreGenA1 <-
gcombobox(items= alelosa,selected=1,editable=FALSE,container=gr
oup,
57.         handler=function(h,...) {
58.             dox3 <-
svalue(comboboxpadreGenA1,index= TRUE)
59.             dox3 <-
as.numeric(dox3)
60.             if(dox3==1) {
61.                 cu3<- "A1*05"}
62.             else if(dox3>1) {
63.                 if(dox3==2) {
64.                     cu3<- "A1*02"}

```

```

65.                                     else if(dox3>2) {
66.                                       if(dox3==3) {
67.                                         cu3<- "A1*03"
68.                                       }
69.                                       else if(dox3==4) {
70.                                         cu3<- "NonRisk"
71.                                       }
72.                                     }
73.                                   }
74.                                   message("ap1 set to
",cu3)
75.                                   }
76.                                 )
77.                                 labelpadreGenA2 <- glabel(text="Father's HLA-DQA allele
2",container = group)
78.                                 comboboxpadreGenA2 <-
gcombobox(items= alelosa,selected=1,editable=FALSE,container=gr
oup,
79.                                               handler=function(h,...) {
80.                                                 dox4 <-
svalue(comboboxpadreGenA2,index= TRUE)
81.                                                 dox4 <-
as.numeric(dox4)
82.                                                 if(dox4==1) {
83.                                                   cu4<- "A1*05"
84.                                                 else if(dox4>1) {
85.                                                   if(dox4==2) {
86.                                                     cu4<- "A1*02"
87.                                                   else if(dox4>2) {
88.                                                     if(dox4==3) {
89.                                                       cu4<- "A1*03"
90.                                                     }
91.                                                     else if(dox4==4) {
92.                                                       cu4<- "NonRisk"
93.                                                     }
94.                                                   }
95.                                                 }
96.                                                 message("ap2 set to
",cu4)
97.                                               }
98.                                             )
99.                                             labelmadreGenB1 <- glabel(text="Mother's HLA-DQB allele
1",container = group)
100.                                             comboboxmadreGenB1 <-
gcombobox(items= alelosb,selected=1,editable=FALSE,container=gr
oup,
101.                                               handler=function(h,...) {
102.                                                 dox5 <-
svalue(comboboxmadreGenB1,index= TRUE)
103.                                                 dox5 <-
as.numeric(dox5)
104.                                                 if(dox5==1) {
105.                                                   cu5<- "B1*02"
106.                                                 else if(dox5>1) {
107.                                                   if(dox5==2) {
108.                                                     cu5<- "B1*03:01"
109.                                                   else if(dox5>2) {
110.                                                     if(dox5==3) {
111.                                                       cu5<- "B1*03:02"
112.                                                     }
113.                                                     else if(dox5==4) {

```

```

114.                                     cu5<- "NonRisk"
115.                                     }
116.                                     }
117.                                     }
118.                                     message ("bm1 set to
", cu5)
119.                                     }
120.     )
121.     labelmadreGenB2 <- glabel(text="Mother's HLA-DQB allele
2", container = group)
122.     comboboxmadreGenB2 <-
gcombobox(items= alelosb, selected=1, editable=FALSE, container=gr
oup,
123.                                     handler=function(h, ...) {
124.                                     dox6 <-
svalue(comboboxmadreGenB2, index= TRUE)
125.                                     dox6 <-
as.numeric(dox6)
126.                                     if(dox6==1) {
127.                                     cu6<- "B1*02"}
128.                                     else if(dox6>1) {
129.                                     if(dox6==2) {
130.                                     cu6<- "B1*03:01"}
131.                                     else if(dox6>2) {
132.                                     if(dox6==3) {
133.                                     cu6<- "B1*03:02"
134.                                     }
135.                                     else if(dox6==4) {
136.                                     cu6<- "NonRisk"
137.                                     }
138.                                     }
139.                                     }
140.                                     message ("bm2 set to
", cu6)
141.                                     }
142.     )
143.     labelpadreGenB1 <- glabel(text="Father's HLA-DQB allele
1", container = group)
144.     comboboxpadreGenB1 <-
gcombobox(items= alelosb, selected=1, editable=FALSE, container=gr
oup,
145.                                     handler=function(h, ...) {
146.                                     dox7 <-
svalue(comboboxpadreGenB1, index= TRUE)
147.                                     dox7 <-
as.numeric(dox7)
148.                                     if(dox7==1) {
149.                                     cu7<- "B1*02"}
150.                                     else if(dox7>1) {
151.                                     if(dox7==2) {
152.                                     cu7<- "B1*03:01"}
153.                                     else if(dox7>2) {
154.                                     if(dox7==3) {
155.                                     cu7<- "B1*03:02"
156.                                     }
157.                                     else if(dox7==4) {
158.                                     cu7<- "NonRisk"
159.                                     }
160.                                     }
161.                                     }

```

```

162.                                     message("bp1 set to
    ",cu7)
163.                                     }
164.     )
165.     labelpadreGenB2 <- glabel(text="Father's HLA-DQB allele
    2",container = group)
166.     comboboxpadreGenB2 <-
    gcombobox(items= alelosb,selected=1,editable=FALSE,container=gr
    oup,
167.                                     handler=function(h,...) {
168.                                     dox8 <-
    svalue(comboboxpadreGenB2,index= TRUE)
169.                                     dox8 <-
    as.numeric(dox8)
170.                                     if(dox8==1) {
171.                                     cu8<- "B1*02"}
172.                                     else if(dox8>1) {
173.                                     if(dox8==2) {
174.                                     cu8<- "B1*03:01"}
175.                                     else if(dox8>2) {
176.                                     if(dox8==3) {
177.                                     cu8<- "B1*03:02"
178.                                     }
179.                                     else if(dox8==4) {
180.                                     cu8<- "NonRisk"
181.                                     }
182.                                     }
183.                                     }
184.                                     message("bp2 set to
    ",cu8)
185.                                     }
186.     )
187.     MathButton <- gbutton("Math Mode", container=group,
188.                             handler=function(h, ...) {
189.                             dox <-
    svalue(comboboxmadreGenA1,index= TRUE)
190.                             dox <- as.numeric(dox)
191.                             if(dox==1) {
192.                             cu<- "A1*05"}
193.                             else if(dox>1) {
194.                             if(dox==2) {
195.                             cu<- "A1*02"}
196.                             else if(dox>2) {
197.                             if(dox==3) {
198.                             cu<- "A1*03"
199.                             }
200.                             else if(dox==4) {
201.                             cu<- "NonRisk"
202.                             }
203.                             }
204.                             }
205.                             dox2 <-
    svalue(comboboxmadreGenA2,index= TRUE)
206.                             dox2 <- as.numeric(dox2)
207.                             if(dox2==1) {
208.                             cu2<- "A1*05"}
209.                             else if(dox2>1) {
210.                             if(dox2==2) {
211.                             cu2<- "A1*02"}
212.                             else if(dox2>2) {
213.                             if(dox2==3) {

```



```

214.         cu2<- "A1*03"
215.     }
216.     else if(dox2==4) {
217.         cu2<- "NonRisk"
218.     }
219. }
220. }
221. dox3 <-
svalue (comboboxpadreGenA1,index= TRUE)
222. dox3 <- as.numeric (dox3)
223. if (dox3==1) {
224.     cu3<- "A1*05" }
225. else if (dox3>1) {
226.     if (dox3==2) {
227.         cu3<- "A1*02" }
228.     else if (dox3>2) {
229.         if (dox3==3) {
230.             cu3<- "A1*03"
231.         }
232.         else if (dox3==4) {
233.             cu3<- "NonRisk"
234.         }
235.     }
236. }
237. dox4 <-
svalue (comboboxpadreGenA2,index= TRUE)
238. dox4 <- as.numeric (dox4)
239. if (dox4==1) {
240.     cu4<- "A1*05" }
241. else if (dox4>1) {
242.     if (dox4==2) {
243.         cu4<- "A1*02" }
244.     else if (dox4>2) {
245.         if (dox4==3) {
246.             cu4<- "A1*03"
247.         }
248.         else if (dox4==4) {
249.             cu4<- "NonRisk"
250.         }
251.     }
252. }
253. dox5 <-
svalue (comboboxmadreGenB1,index= TRUE)
254. dox5 <- as.numeric (dox5)
255. if (dox5==1) {
256.     cu5<- "B1*02" }
257. else if (dox5>1) {
258.     if (dox5==2) {
259.         cu5<- "B1*03:01" }
260.     else if (dox5>2) {
261.         if (dox5==3) {
262.             cu5<- "B1*03:02"
263.         }
264.         else if (dox5==4) {
265.             cu5<- "NonRisk"
266.         }
267.     }
268. }
269. dox6 <-
svalue (comboboxmadreGenB2,index= TRUE)
270. dox6 <- as.numeric (dox6)

```

```

271.         if (dox6==1) {
272.             cu6<- "B1*02"
273.         else if (dox6>1) {
274.             if (dox6==2) {
275.                 cu6<- "B1*03:01"
276.             else if (dox6>2) {
277.                 if (dox6==3) {
278.                     cu6<- "B1*03:02"
279.                 }
280.                 else if (dox6==4) {
281.                     cu6<- "NonRisk"
282.                 }
283.             }
284.         }
285.         dox7 <-
svalue (comboboxpadreGenB1, index= TRUE)
286.         dox7 <- as.numeric (dox7)
287.         if (dox7==1) {
288.             cu7<- "B1*02"
289.         else if (dox7>1) {
290.             if (dox7==2) {
291.                 cu7<- "B1*03:01"
292.             else if (dox7>2) {
293.                 if (dox7==3) {
294.                     cu7<- "B1*03:02"
295.                 }
296.                 else if (dox7==4) {
297.                     cu7<- "NonRisk"
298.                 }
299.             }
300.         }
301.         dox8 <-
svalue (comboboxpadreGenB2, index= TRUE)
302.         dox8 <- as.numeric (dox8)
303.         if (dox8==1) {
304.             cu8<- "B1*02"
305.         else if (dox8>1) {
306.             if (dox8==2) {
307.                 cu8<- "B1*03:01"
308.             else if (dox8>2) {
309.                 if (dox8==3) {
310.                     cu8<- "B1*03:02"
311.                 }
312.                 else if (dox8==4) {
313.                     cu8<- "NonRisk"
314.                 }
315.             }
316.         }
317.         data3 <-
InheritanceC (ap1=cu3, ap2=cu4, bp1 = cu7, bp2=cu8, bm1=cu5, bm2=cu6,
am1=cu, am2=cu2, mode="Math", verbose=T)
318.         data3 <- as.matrix (data3)
319.         colie <- c (colnames (data3))
320.         rowie <- c (rownames (data3))
321.         data3 <- cbind (data3, 0)
322.         data3 <- rbind (data3, 0)
323.         data3 [2:5, 2:5] <- data3 [1:4, 1:4]
324.         data3 [2:5, 1] <-
rownames (data3) [1:4]
325.         data3 [1, 2:5] <-
colnames (data3) [1:4]

```

```

326. colnames(data3)[1:4] colnames(data3)[2:5]<-
327. colnames(data3)[1] <- "-"
328. colnames(data3)[2:5] <-
c("A", "B", "C", "D")
329. data3[1,1] <- "Gametes"
330. data3 <- as.data.frame(data3)
331. aws <- as.matrix(data3)
332. aws <- aws[2:5,2:5]
333. aws <- as.matrix(aws)
334. rownames(aws) <- NULL
335. colnames(aws) <- NULL
336. awstabila <- table(aws)
337. awstabila <- as.matrix(awstabila)
338. nombres <- c(rownames(awstabila))
339. matrix(0, nrow = 3, ncol=length(awstabila[,1]))
340. tableto[1,] <- nombres
341. qu <- awstabila[,1]
342. tableto[2,] <- qu
343. quqi <- as.numeric(qu)
344. quqi <- quqi/16
345. quqi <- quqi * 100
346. quqi <- paste(quqi, "%")
347. tableto[3,] <- quqi
348. cosilla <- c("Risk/Risk
Group", "Probability out of the 16 possible
combinations", "Probability in percentage")
349. tableto <- cbind(tableto, 0)
350. tableto[,length(tableto[1,])] <-
cosilla
351. tableto <- as.data.frame(tableto)
352. Results <-
gwindow(title="Results",width = 1000,height = 600)
353. groupie <-
ggroup(horizontal=FALSE, container=Results)
354. gumi <- glabel("Punnett
square",container=groupie)
355. tabla <-
gtable(items=data3,multiple = TRUE,expand=TRUE,chosen.col = 1,i
ndex=TRUE, fill=TRUE,container = groupie)
356. gomi <- glabel("Risk frequency
table",container=groupie)
357. tablafrec <-
gtable(items=tableto,multiple = TRUE,expand=TRUE,chosen.col = 1
,index=TRUE, fill=TRUE,container = groupie)
358. guuu <- glabel("Click <<open>> to
select saving path and the name of the
file",editable = FALSE,container=groupie)
359. tus <-
gfilebrowse(text = "Path",type = "save",initial.filename = "Unt
itled1.xlsx"
,container = g
roupie)
361. savebutton <- gbutton("Save",
362. handler= fu
nction(h, ...) {
363. write.xls
x(x = data3, file = svalue(tus),sheetName = "Punnett
square",col.names = FALSE, row.names = FALSE)

```

```

364.                                                                 write.xls
x(x = tableto, file = svalue(tus), sheetName = "Risk Frecuency
Table", append = T, col.names = FALSE, row.names = FALSE)
365.                                                                 mensaje <-
- gwindow(title = "Results")
366.                                                                 groupiess
<- ggroup(horizontal = FALSE, container= mensaje)
367.                                                                 labus <-
glabel(text = paste ("Saved on", svalue(tus)), container= mensaje)
368.                                                                 }, container
=groupie
369.                                                                 )
370.                                                                 })
371.     ClassButton <- gbutton("Class Mode", container=group,
372.                            handler=function(h, ...) {
373.                                dox <-
svalue(comboboxmadreGenA1, index= TRUE)
374.                                dox <- as.numeric(dox)
375.                                if(dox==1) {
376.                                    cu<- "A1*05"}
377.                                else if(dox>1) {
378.                                    if(dox==2) {
379.                                        cu<- "A1*02"}
380.                                    else if(dox>2) {
381.                                        if(dox==3) {
382.                                            cu<- "A1*03"
383.                                        }
384.                                        else if(dox==4) {
385.                                            cu<- "NonRisk"
386.                                        }
387.                                    }
388.                                }
389.                                dox2 <-
svalue(comboboxmadreGenA2, index= TRUE)
390.                                dox2 <- as.numeric(dox2)
391.                                if(dox2==1) {
392.                                    cu2<- "A1*05"}
393.                                else if(dox2>1) {
394.                                    if(dox2==2) {
395.                                        cu2<- "A1*02"}
396.                                    else if(dox2>2) {
397.                                        if(dox2==3) {
398.                                            cu2<- "A1*03"
399.                                        }
400.                                        else if(dox2==4) {
401.                                            cu2<- "NonRisk"
402.                                        }
403.                                    }
404.                                }
405.                                dox3 <-
svalue(comboboxpadreGenA1, index= TRUE)
406.                                dox3 <- as.numeric(dox3)
407.                                if(dox3==1) {
408.                                    cu3<- "A1*05"}
409.                                else if(dox3>1) {
410.                                    if(dox3==2) {
411.                                        cu3<- "A1*02"}
412.                                    else if(dox3>2) {
413.                                        if(dox3==3) {
414.                                            cu3<- "A1*03"
415.                                        }

```

```

416.         else if (dox3==4) {
417.             cu3<- "NonRisk"
418.         }
419.     }
420. }
421. dox4 <-
svalue (comboboxpadreGenA2,index= TRUE)
422. dox4 <- as.numeric (dox4)
423. if (dox4==1) {
424.     cu4<- "A1*05"
425. else if (dox4>1) {
426.     if (dox4==2) {
427.         cu4<- "A1*02"
428.     else if (dox4>2) {
429.         if (dox4==3) {
430.             cu4<- "A1*03"
431.         }
432.         else if (dox4==4) {
433.             cu4<- "NonRisk"
434.         }
435.     }
436. }
437. dox5 <-
svalue (comboboxmadreGenB1,index= TRUE)
438. dox5 <- as.numeric (dox5)
439. if (dox5==1) {
440.     cu5<- "B1*02"
441. else if (dox5>1) {
442.     if (dox5==2) {
443.         cu5<- "B1*03:01"
444.     else if (dox5>2) {
445.         if (dox5==3) {
446.             cu5<- "B1*03:02"
447.         }
448.         else if (dox5==4) {
449.             cu5<- "NonRisk"
450.         }
451.     }
452. }
453. dox6 <-
svalue (comboboxmadreGenB2,index= TRUE)
454. dox6 <- as.numeric (dox6)
455. if (dox6==1) {
456.     cu6<- "B1*02"
457. else if (dox6>1) {
458.     if (dox6==2) {
459.         cu6<- "B1*03:01"
460.     else if (dox6>2) {
461.         if (dox6==3) {
462.             cu6<- "B1*03:02"
463.         }
464.         else if (dox6==4) {
465.             cu6<- "NonRisk"
466.         }
467.     }
468. }
469. dox7 <-
svalue (comboboxpadreGenB1,index= TRUE)
470. dox7 <- as.numeric (dox7)
471. if (dox7==1) {
472.     cu7<- "B1*02"

```

```

473.         else if(dox7>1) {
474.             if(dox7==2) {
475.                 cu7<- "B1*03:01"
476.             } else if(dox7>2) {
477.                 if(dox7==3) {
478.                     cu7<- "B1*03:02"
479.                 }
480.             } else if(dox7==4) {
481.                 cu7<- "NonRisk"
482.             }
483.         }
484.     }
485.     dox8 <-
svalue (comboboxpadreGenB2, index= TRUE)
486.     dox8 <- as.numeric(dox8)
487.     if(dox8==1) {
488.         cu8<- "B1*02"
489.     } else if(dox8>1) {
490.         if(dox8==2) {
491.             cu8<- "B1*03:01"
492.         } else if(dox8>2) {
493.             if(dox8==3) {
494.                 cu8<- "B1*03:02"
495.             }
496.         } else if(dox8==4) {
497.             cu8<- "NonRisk"
498.         }
499.     }
500. }
501. data3 <-
InheritanceC (ap1=cu3, ap2=cu4, bp1 = cu7, bp2=cu8, bm1=cu5, bm2=cu6,
aml=cu, am2=cu2, mode="Class", verbose=T)
502. data3 <- as.matrix(data3)
503. colie <- c(colnames(data3))
504. rowie <- c(rownames(data3))
505. data3 <- cbind(data3,0)
506. data3 <- rbind(data3,0)
507. data3[2:5,2:5] <- data3[1:4,1:4]
508. data3[2:5,1] <-
rownames(data3)[1:4]
509. data3[1,2:5] <-
colnames(data3)[1:4]
510. colnames(data3)[2:5]<-
colnames(data3)[1:4]
511. colnames(data3)[1] <- "-"
512. colnames(data3)[2:5] <-
c("A", "B", "C", "D")
513. data3[1,1] <- "Gametes"
514. data3 <- as.data.frame(data3)
515. aws <- as.matrix(data3)
516. aws <- aws[2:5,2:5]
517. aws <- as.matrix(aws)
518. rownames(aws) <- NULL
519. colnames(aws) <- NULL
520. awstabla <- table(aws)
521. awstabla <- as.matrix(awstabla)
522. nombres <- c(rownames(awstabla))
523. tableto <-
matrix(0, nrow = 3, ncol=length(awstabla[,1]))
524. tableto[1,] <- nombres
525. qu <- awstabla[,1]

```

```

526.         tableto[2,] <- qu
527.         quqi <- as.numeric(qu)
528.         quqi <- quqi/16
529.         quqi <- quqi * 100
530.         quqi <- paste(quqi, "%")
531.         tableto[3,] <- quqi
532.         cosilla <- c("Risk/Risk
Group", "Probability out of the 16 possible
combinations", "Probability in percentage")
533.         tableto <- cbind(tableto, 0)
534.         tableto[,length(tableto[1,])] <-
cosilla
535.         tableto <-
as.data.frame(tableto)
536.         Results <-
gwindow(title="Results",width = 1000,height = 600)
537.         groupie <-
ggroup(horizontal=FALSE, container=Results)
538.         gumi <- glabel("Punnett
square",container=groupie)
539.         tabla <-
gtable(items=data3,multiple = TRUE,expand=TRUE,chosen.col = 1,i
ndex=TRUE, fill=TRUE,container = groupie)
540.         gomi <- glabel("Risk frequency
table",container=groupie)
541.         tablafrec <-
gtable(items=tableto,multiple = TRUE,expand=TRUE,chosen.col = 1
,index=TRUE, fill=TRUE,container = groupie)
542.         guuu <- glabel("Click <<open>>
to select saving path and the name of the
file",editable = FALSE,container=groupie)
543.         tus <-
gfilebrowse(text = "Path",type = "save",initial.filename = "Unt
itled1.xlsx"
,container =
groupie)
544.         savebutton <- gbutton("Save",
handler=f
unction(h,...){
545.         write.xls
546.         sx(x = data3,file = svalue(tus),sheetName = "DQResults",col.name
s = FALSE,row.names = FALSE)
547.         write.xls
548.         sx(x = tableto,file = svalue(tus),sheetName = "Risk Frequency
Table",append = T,col.names = FALSE,row.names = FALSE)
549.         mensaje
<- gwindow(title = "Results")
550.         groupies
s <- ggroup(horizontal = FALSE, container=mensaje)
551.         labus <-
glabel(text = paste("Saved on",svalue(tus)),container=mensaje)
552.         },containe
r=groupie
553.         )
554.         })
555.         BothModesButton <- gbutton("Math & Class Mode",
container=group,
556.         handler=function(h, ...) {
557.         dox <-
svalue(comboboxmadreGenAl,index= TRUE)
558.         dox <- as.numeric(dox)

```

```

559.         if(dox==1){
560.             cu<- "A1*05"}
561.         else if(dox>1){
562.             if(dox==2){
563.                 cu<- "A1*02"}
564.             else if(dox>2){
565.                 if(dox==3){
566.                     cu<- "A1*03"
567.                 }
568.                 else if(dox==4){
569.                     cu<- "NonRisk"
570.                 }
571.             }
572.         }
573.         dox2 <-
svalue(comboboxmadreGenA2,index= TRUE)
574.         dox2 <- as.numeric(dox2)
575.         if(dox2==1){
576.             cu2<- "A1*05"}
577.         else if(dox2>1){
578.             if(dox2==2){
579.                 cu2<- "A1*02"}
580.             else if(dox2>2){
581.                 if(dox2==3){
582.                     cu2<- "A1*03"
583.                 }
584.                 else if(dox2==4){
585.                     cu2<- "NonRisk"
586.                 }
587.             }
588.         }
589.         dox3 <-
svalue(comboboxpadreGenA1,index= TRUE)
590.         dox3 <- as.numeric(dox3)
591.         if(dox3==1){
592.             cu3<- "A1*05"}
593.         else if(dox3>1){
594.             if(dox3==2){
595.                 cu3<- "A1*02"}
596.             else if(dox3>2){
597.                 if(dox3==3){
598.                     cu3<- "A1*03"
599.                 }
600.                 else if(dox3==4){
601.                     cu3<- "NonRisk"
602.                 }
603.             }
604.         }
605.         dox4 <-
svalue(comboboxpadreGenA2,index= TRUE)
606.         dox4 <- as.numeric(dox4)
607.         if(dox4==1){
608.             cu4<- "A1*05"}
609.         else if(dox4>1){
610.             if(dox4==2){
611.                 cu4<- "A1*02"}
612.             else if(dox4>2){
613.                 if(dox4==3){
614.                     cu4<- "A1*03"
615.                 }
616.                 else if(dox4==4){

```



```

617.             cu4<- "NonRisk"
618.         }
619.     }
620. }
621.     dox5 <-
svalue (comboboxmadreGenB1,index= TRUE)
622.     dox5 <- as.numeric (dox5)
623.     if (dox5==1) {
624.         cu5<- "B1*02"
625.     else if (dox5>1) {
626.         if (dox5==2) {
627.             cu5<- "B1*03:01"
628.         else if (dox5>2) {
629.             if (dox5==3) {
630.                 cu5<- "B1*03:02"
631.             }
632.             else if (dox5==4) {
633.                 cu5<- "NonRisk"
634.             }
635.         }
636.     }
637.     dox6 <-
svalue (comboboxmadreGenB2,index= TRUE)
638.     dox6 <- as.numeric (dox6)
639.     if (dox6==1) {
640.         cu6<- "B1*02"
641.     else if (dox6>1) {
642.         if (dox6==2) {
643.             cu6<- "B1*03:01"
644.         else if (dox6>2) {
645.             if (dox6==3) {
646.                 cu6<- "B1*03:02"
647.             }
648.             else if (dox6==4) {
649.                 cu6<- "NonRisk"
650.             }
651.         }
652.     }
653.     dox7 <-
svalue (comboboxpadreGenB1,index= TRUE)
654.     dox7 <- as.numeric (dox7)
655.     if (dox7==1) {
656.         cu7<- "B1*02"
657.     else if (dox7>1) {
658.         if (dox7==2) {
659.             cu7<- "B1*03:01"
660.         else if (dox7>2) {
661.             if (dox7==3) {
662.                 cu7<- "B1*03:02"
663.             }
664.             else if (dox7==4) {
665.                 cu7<- "NonRisk"
666.             }
667.         }
668.     }
669.     dox8 <-
svalue (comboboxpadreGenB2,index= TRUE)
670.     dox8 <- as.numeric (dox8)
671.     if (dox8==1) {
672.         cu8<- "B1*02"
673.     else if (dox8>1) {

```

```

674.         if(dox8==2) {
675.             cu8<- "B1*03:01"}
676.         else if(dox8>2) {
677.             if(dox8==3) {
678.                 cu8<- "B1*03:02"
679.             }
680.             else if(dox8==4) {
681.                 cu8<- "NonRisk"
682.             }
683.         }
684.     }
685.     data3 <-
        InheritanceC (ap1=cu3,ap2=cu4,bp1 = cu7,bp2=cu8,bm1=cu5,bm2=cu6,
        am1=cu, am2=cu2, mode="Class", verbose=T)
686.     data4 <-
        InheritanceC (ap1=cu3,ap2=cu4,bp1 = cu7,bp2=cu8,bm1=cu5,bm2=cu6,
        am1=cu, am2=cu2, mode="Math", verbose=T)
687.     data3 <- as.matrix(data3)
688.     data4 <- as.matrix(data4)
689.     colie <- c(colnames(data3))
690.     colie4 <- c(colnames(data4))
691.     rowie <- c(rownames(data3))
692.     rowie4 <- c(rownames(data4))
693.     data3 <- cbind(data3,0)
694.     data3 <- rbind(data3,0)
695.     data3[2:5,2:5] <-
        data3[1:4,1:4]
696.     data3[2:5,1] <-
        rownames(data3)[1:4]
697.     data3[1,2:5] <-
        colnames(data3)[1:4]
698.     colnames(data3)[2:5]<-
        colnames(data3)[1:4]
699.     colnames(data3)[1] <- "-"
700.     colnames(data3)[2:5] <-
        c("A", "B", "C", "D")
701.     data4 <- cbind(data4,0)
702.     data4 <- rbind(data4,0)
703.     data4[2:5,2:5] <-
        data4[1:4,1:4]
704.     data4[2:5,1] <-
        rownames(data4)[1:4]
705.     data4[1,2:5] <-
        colnames(data4)[1:4]
706.     colnames(data4)[2:5]<-
        colnames(data4)[1:4]
707.     colnames(data4)[1] <- "-"
708.     colnames(data4)[2:5] <-
        c("A", "B", "C", "D")
709.     data3[1,1] <- "Gametes"
710.     data4[1,1] <- "Gametes"
711.     aws <- as.matrix(data3)
712.     aws <- aws[2:5,2:5]
713.     aws <- as.matrix(aws)
714.     rownames(aws) <- NULL
715.     colnames(aws) <- NULL
716.     awstabla <- table(aws)
717.     awstabla <-
        as.matrix(awstabla)
718.     nombres <-
        c(rownames(awstabla))

```

```

719.      tableto <-
      matrix(0, nrow = 3, ncol=length(awstabila[,1]))
720.      tableto[1,] <- nombres
721.      qu <- awstabila[,1]
722.      tableto[2,] <- qu
723.      quqi <- as.numeric(qu)
724.      quqi <- quqi/16
725.      quqi <- quqi * 100
726.      quqi <- paste(quqi, "%")
727.      tableto[3,] <- quqi
728.      cosilla <- c("Risk/Risk
      Group", "Probability out of the 16 possible
      combinations", "Probability in percentage")
729.      tableto <- cbind(tableto, 0)
730.      tableto[,length(tableto[1,])
      ] <- cosilla
731.      tableto <-
      as.data.frame(tableto)
732.      aws4 <- as.matrix(data4)
733.      aws4 <- aws4[2:5,2:5]
734.      aws4 <- as.matrix(aws4)
735.      rownames(aws4) <- NULL
736.      colnames(aws4) <- NULL
737.      awstabila4 <- table(aws4)
738.      awstabila4 <-
      as.matrix(awstabila4)
739.      nombres4 <-
      c(rownames(awstabila4))
740.      tableto4 <-
      matrix(0, nrow = 3, ncol=length(awstabila4[,1]))
741.      tableto4[1,] <- nombres4
742.      qu4 <- awstabila4[,1]
743.      tableto4[2,] <- qu4
744.      quqi4 <- as.numeric(qu4)
745.      quqi4 <- quqi4/16
746.      quqi4 <- quqi4 * 100
747.      quqi4 <- paste(quqi4, "%")
748.      tableto4[3,] <- quqi4
749.      cosilla4 <- c("Risk/Risk
      Group", "Probability out of the 16 possible
      combinations", "Probability in percentage")
750.      tableto4 <-
      cbind(tableto4, 0)
751.      tableto4[,length(tableto4[1,
      ])] <- cosilla4
752.      tableto4 <-
      as.data.frame(tableto4)
753.      Results <-
      gwindow(title="Class mode results",width = 1000,height = 600)
754.      Results4 <-
      gwindow(title="Math mode results",width = 1000,height = 600)
755.      groupie <-
      ggroup(horizontal=FALSE, container=Results)
756.      groupie4 <-
      ggroup(horizontal=FALSE, container=Results4)
757.      gumi <- glabel("Punnett
      square",container=groupie)
758.      tabla <-
      gtable(items=data3,multiple = TRUE,expand=TRUE,chosen.col = 1,i
      ndex=TRUE, fill=TRUE,container = groupie)

```

```

759.         gomi <- glabel("Risk
frequency table",container=groupie)
760.         tablafrec <-
gtable(items=tableto,multiple = TRUE,expand=TRUE,chosen.col = 1
,index=TRUE, fill=TRUE,container = groupie)
761.         gumi4 <- glabel("Punnett
square",container=groupie4)
762.         tabla4 <-
gtable(items=data4,multiple = TRUE,expand=TRUE,chosen.col = 1,i
ndex=TRUE, fill=TRUE,container = groupie4)
763.         gomi4 <- glabel("Risk
frequency table",container=groupie4)
764.         tablafrec4 <-
gtable(items=tableto4,multiple = TRUE,expand=TRUE,chosen.col =
1,index=TRUE, fill=TRUE,container = groupie4)
765.         guuu <- glabel("Click
<<open>> to select saving path and the name of the
file",editable = FALSE,container=groupie)
766.         tus <-
gfilebrowse(text = "Path",type = "save",initial.filename = "Unt
itled1.xlsx"
767.         ,containe
r = groupie)
768.         savebutton <-
gbutton("Save",
769.         handle
r= function(h,...){
770.         writ
e.xlsx(x = data3,file = svalue(tus),sheetName = "DQResults",col.
names = FALSE,row.names = FALSE)
771.         writ
e.xlsx(x = tableto,file = svalue(tus),sheetName = "Risk
Frecuency Table",append = T,col.names = FALSE,row.names = FALSE)
772.         mens
aje <- gwindow(title = "Results")
773.         grou
piess <- ggroup(horizontal = FALSE, container= mensaje)
774.         labu
s <- glabel(text = paste("Saved
on",svalue(tus)),container= mensaje)
775.         },cont
ainer=groupie
776.         )
777.         guuu4 <- glabel("Click
<<open>> to select saving path and the name of the
file",editable = FALSE,container=groupie4)
778.         tus4 <-
gfilebrowse(text = "Path",type = "save",initial.filename = "Unt
itled1.xlsx"
779.         ,contain
er = groupie4)
780.         savebutton4 <-
gbutton("Save",
781.         handl
er= function(h,...){
782.         wri
te.xlsx(x = data4,file = svalue(tus4),sheetName = "DQResults",co
l.names = FALSE,row.names = FALSE)
783.         wri
te.xlsx(x = tableto4,file = svalue(tus4),sheetName = "Risk
Frecuency Table",append = T,col.names = FALSE,row.names = FALSE)

```

```

784.                                     men
      saje4 <- gwindow(title = "Results")
785.                                     gro
      upiess4 <- ggroup(horizontal = FALSE, container= mensaje4)
786.                                     lab
      us4 <- glabel(text = paste("Saved
      on", svalue(tus4)), container= mensaje4)
787.                                     }, con
      tainer=groupie4
788.                                     )
789.                                     })
790.
791.   }

```