GUILLERMO AYALA GALLEGO

BIOINFORMÁTICA ESTADÍSTICA
ESTADÍSTICA DE DATOS ÓMICOS

UNIVERSITAT DE VALÈNCIA

Guillermo Ayala
Guillermo.Ayala@uv.es

Índice general

Prólogo

Este texto se ocupa de la aplicación de técnicas estadística a datos ómicos. Tiene una orientación aplicada aunque sin olvidar la teoría en la que se basan los métodos que utilizamos. La Probabilidad y la Estadística es un edificio no una colección de herramientas y por ello es imprescindible la comprensión de los conceptos que están implementados. Con frecuencia, el usuario de software lo uso sin saber mucho de lo que está usando. Aunque el código funcione.

Un investigador que publique un artículo al mes durante un año tendrá al final de ese año un total de 12 publicaciones. Después de ese año maravilloso de publicaciones, el siguiente es tan bueno como el anterior. Y sigue publicando un trabajo al mes durante este segundo año. Ya reune 24 publicaciones en dos años agotadores. Y esto mismo lo hace durante 10 años.[1] Al final de esta decada prodigiosa tendrá 120 publicaciones. Ya ha justificado su vida como investigador. Pero no, incansable sigue otros diez años y alcanza al cabo de 20 años los 240 publicaciones. Y diez años más. Exhausto él o ella (y cuantos hayan intentado leer todas estas publicaciones) ya ha llegado a las 360 publicaciones. Hay una enorme cantidad de investigadores que superan esta cantidad. La superan ampliamente. Por amor de Dios, no más. Es seguro (con probabilidad uno) que han participado activamente en todas estas publicaciones. Pero el resto de los humanos no tenemos la culpa. Podríamos llamarlo un 12/30. De hecho hay 12/40 y mucho más. Hay numerosos estudios sobre el crecimiento del número de publicaciones. Sin entrar en precisiones innecesarias, en el momento actual, se tarda unos nueve años en doblar el número de publicaciones.[2] Evidentemente el planeta no tiene un problema de superpoblación de personas. Es de publicaciones científicas. Con frecuencia leemos cómo muchas publicaciones han de retirarse[1] Nunca es culpa de nadie. Nunca es cierto que los autores habían apostado por lo rápido. Por lo fácil. Por no controlar el trabajo. Por no limitarse a copiar el tratamiento de datos de otros sin preocuparse si era adecuado en su caso. O si lo entendían. O controlar el trabajo del precario, mal pagado y sufrido becario/a que echa demasiadas horas y todo es nuevo (y bueno) para él/ella. Necesitaban algunas decenas (centenares) de publicaciones más para que las generaciones futuras los/las recuerden. Y mucho que los recordarán. Sé que los recordarán. Sobre la evolución de las revistas científicas es interesante https://www.eldiario.es/sociedad/neoliberalismo-burocracia-robert-maxwell-revistas-cientificas-primaron-negocio_1_9952229.html.

Y debiera de ser posiblemente candidato a algún premio importante.

[1] Asumimos que no es un superhéroe con algún superpoder producido por una pequeña araña radioactiva o algún rayo cósmico por salir sin paraguas a la calle.

[2] Un optimista diría que doblamos el conocimiento que la humanidad ha logrado atesorar cada nueve años.

[1] https://www.abc.es/sociedad/abci-retirada-ultima-investigacion-reciente-premio-nobel-202001050137_noticia.html, https://elpais.com/elpais/2019/01/27/ciencia/1548629779_450088.html, https://elpais.com/elpais/2019/10/14/ciencia/1571052466_871787.html, https://www.theguardian.com/world/2020/jun/03/covid-19-surgisphere-who-world-health-organization-hydroxychloroquine?.

En campos como las Matemáticas o la Estadística **teórica** el producto está a la vista. Un teorema lleva su prueba y si hay un fallo puede ser visto y corregido por la comunidad de expertos en ese tema en algún momento posterior. Sin embargo, en campos como la Biología o la Medicina las cosas no son así. No es tan fácil comprobar la reproducibilidad de los resultados. En [12] se analizan los resultados de una encuesta a 1576 investigadores. La proporción de trabajos que no se pueden reproducir es enorme. Los factores que se comentan como importantes son los siguientes de mayor a menor frecuencia.

1. No incluir una descripción completa del trabajo.
2. Presión por publicar.
3. Trabajos sin un estudio de potencia previo y un análisis pobre de los resultados.
4. Insuficiente replicación en el laboratorio original.
5. Insuficiente supervisión.
6. La metodología y el código informático utilizado no disponible.
7. Un diseño experimental pobre o inadecuado.
8. Los datos originales no disponibles.
9. Fraude.
10. Una revisión por parte de la revista insuficiente.

Con el análisis estadístico de los datos tienen que ver los puntos 3, 6, 7 y 8. Hay que hacer bien las cosas desde el principio.

En muchas ocasiones biólogos o médicos o biotecnólogos o bioquímicos o ... me han indicado que querían aprender más Probabilidad y Estadística. Normalmente después de que les has analizado unos datos para una tesis o un artículo. Al principio, con interés, les recomiendas algún texto esperando que les ayude. El final de la historia suele ser el mismo. Que no leen nunca el libro. Aprender Probabilidad y Estadística es muy fácil. Se coge un libro sencillo de Probabilidad y luego otro de Estadística. Cada libro lleva una página que pone el número uno. Se lee esa página. Se le da la vuelta y se lee la siguiente numerada con el 2. Y así sucesivamente. Y no hay otra.

Estas notas tratan de la aplicación de procedimientos estadísticos al **análisis de datos de alto rendimiento**. Bonito el nombre pero: ¿qué son datos de alto rendimiento? Datos que rompen lo que tradicionalmente era un prerrequisito en Estadística (multivariante). Muchos procedimientos estadísticos empiezan indicando que el número de observaciones, n, ha de ser mayor que el número de variables por observación, p. Actualmente es frecuente (mejor habitual) que los datos no los recoja un experimentador con un lápiz y un papel y luego los introduzca con mucho trabajo en una hoja de cálculo. Lo hacen disposivos electrónicos conectados con ordenadores. Por eso los datos tienen dimensiones p que marean: miles de variables frente a decenas (con suerte algo más de un centenar) de observaciones o muestras. ¿Y qué hacemos para analizar esto? Lo que se pueda.[3] De esto van estas notas, **de lo que se pueda**. Son unas notas en progreso. Se van añadiendo ideas, técnicas, software y se ve cómo incorporar estos

[3] Uno no está obligado más que a hacer las cosas lo mejor que pueda y no más.

análisis en nuestro trabajo. También[4] voy incorporando nuevos tipos de información. De momento, analizamos datos de expresión de gen (o expresión génica) sabiendo que (la mayor parte de) lo que hacemos es aplicable en otros contextos. El tema § 2 está dedicado a este tipo de datos, sus características y técnicas de preprocesado. Es el tema específico de esta información. Si se sustituye este capítulo por otro dedicado a otra técnica de adquisición de información casi todo lo que sigue es aplicable.

[4] En la medida en que me voy encontrando con nuevos tipos de datos que entienda.

Seguir el material de este manual supone unos conocimientos biológicos fundamentalmente de genética molecular. Conocimientos muy básicos para un especialista en campos como Biología, Biotecnología, Bioquímica o Medicina pero no tan presentes en personas con una formación en otros estudios no cercanos a la Biología. Para este segundo de grupo es recomendable leer alguna introducción breve y simple. En particular, recomiendo [58, chapter 1].

El objetivo es que estas notas ayuden a quien lee y no que demuestren que sabe el que las escribe.[5] En lo biológico hay imprecisión por ignorancia, en lo probabilístico y estadístico hay imprecisión porque pretenden ser unas notas para un público interesado en estos temas pero con formación en Biología, Biotecnología, Bioquímica o Medicina fundamentalmente. Hay que tener algún interés en Probabilidad, Estadística e Informática para poder seguirlas en su totalidad. Todo en el curso está basado en R y Bioconductor.

[5] Sobre esta cuestión yo diría que que el autor se apaña como puede (que no es mucho) con los conceptos biológicos.

Se han de leer artículos de revistas de diversos ámbitos científicos. Como matemático/estadístico es decepcionante intentar entender la metodología que se ha aplicado cuando lees trabajos de revistas de gran prestigio en Biología. Muchos trabajos tienden a ser una pura ilustración de los métodos propuestos. El cómo se hace realmente se relega al final en una sección suplementaria o directamente a un archivo de material suplementario. Dónde se coloca da lo mismo: No da lo mismo que no se incluya o que se incluya sin el detalle necesario. Hay una auténtica aversión, miedo, odio a la formulación matemática de las cosas. El lenguaje matemático se ha desarrollado para ser preciso. No se puede expresar todo simplemente con palabras. De hecho, a veces, es imposible saber qué se está haciendo porque no quieren poner una simple formula. Si la revista de Biología publica un trabajo metodológico ha de asumir que tiene que dar una formulación precisa y eso pasa por incluir una presentación con lenguaje matemático. De lo contrario, el artículo es inútil aunque el índice de impacto de la revista sea mayor.

En los datos que vamos a analizar lo más frecuente en que se tenga una muestra de tejido y se aplique alguna técnica ómica. Sin embargo, y cada vez más, se puede analizar cada célula individualmente. Mientras que en el caso en que la muestra se refiere al tejido el interés es comparar los datos observados entre condiciones biológicas distintas, cuando la información llega al detalle de la célula, además de poder seguir comparando los valores entre condiciones, se abre la posibilidad de clasificar la célula en distintos tipos de celulas (un problema de clasificación). Con datos a nivel de célula también permite la comparación entre condiciones biológicas. Obviamente es una información desagregada.

Estas notas utilizan sistemáticamente R/Bioconductor y hacemos algún uso de Bash y Python. Un buen punto de inicio para ver las posibilidades de R/Bioconductor en análisis estadístico de datos ómicos

es https://cran.r-project.org/web/views/Omics.html.

Cuando empecé con esto de la Probabilidad y la Estadística uno leía libros y artículos, entendías aquello y luego intentabas descifrar cómo aplicaba (mejor implementaba) estas técnicas el software comercial (en mi caso **SPSS**). Lo primero era bonito. Los autores intentan que les entiendas porque tratan de transmitir ideas. Sin embargo, el software comercial no piensa así (y esto no es necesariamente malo). El software comercial intenta dar un producto bueno y fácilmente utilizable para llegar a un máximo de usuarios. Solamente suelen considerar temas sobre los que hay mucho interés y muchos (potenciales) usuarios. Y esto es correcto. No es la opción elegida en estas notas. Hemos elegido trabajar con software libre. Tanto R como Bioconductor son el resultado de un gran trabajo coordinado de muchas personas. Algunos son proceden del mundo académico. En otros ocasiones proceden de empresas para las cuales les resulta interesante que se disponga de software que permita utilizar su hardware.

¿ES R/Bioconductor la única opción? Por supuesto que no. Y tampoco tengo muy claro que sea la mejor. Indudablemente es buena. En estas notas nos centramos en el análisis estadístico de datos ómicos. Y desde el punto de vista de la Estadística sí que podemos afirmar que, en el momento de escribir, R es la mejor opción para análisis estadístico de datos y Bioconductor es su opción natural cuando nuestro interés es en datos ómicos.[6]

Es muy recomendable consultar http://www.bioconductor.org/help/workflows/ en donde podemos encontrar análisis completos de datos.

Es ingente la cantidad de publicaciones científicas que llevan tratamientos estadísticos. Los investigadores suelen saber qué quieren estudiar. Diseñan un experimento y observan datos. Lo que se hace después no lo suelen conocer. No suelen conocer las técnicas que han utilizado.[7] Citan los procedimientos estadísticos y no indican el software utilizado si está disponible en la red o, en el caso de que sea propio de los autores, dónde se puede conseguir. Sin duda alguna el control de la calidad de los tratamientos estadísticos descansa en que cada lector de una publicación científica tenga a su disposición el artículo (que básicamente es la explicación de lo que se ha hecho) así como los datos [8] y **todo** el código necesario para reproducir **todo** el tratamiento estadístico realizado con los datos. Se repite todo porque en muchas ocasiones lo que se ha descartado puede ser tan interesante como lo que se ha publicado. Este es el conocido sesgo a publicar los resultados significativos descartando los *no* significativos. Sin esto, no se puede realizar un control adecuado de un tratamiento estadístico de datos de alto rendimiento (de hecho, de ningún tipo de datos). Esto nos lleva a los conceptos de programación literaria o comentada (literate programming) propuesto por Donald K. Knuth[9] y, de un modo más genérico, a la investigación reproducible [10]. R/Bioconductor incorpora muchas herramientas para realizar investigación reproducible.[11] En particular, este texto está realizado utilizando [97, knitr]. Todos los datos que se utilizan están disponibles en bases de datos públicas como (sobre todo) GEO o ArrayExpress. No de todos los datos que se utilizados tenemos los datos sin procesado previo. Los usamos por haber sido analizados en otros textos o en ejemplos de file:///home/gag/ownCloud/quindingTFM/qindingTFM-20240201T084237Z-001.zip R/Bioconductor o, simplemente, porque son bonitos de estu-

[6] Sin embargo, en http://neurolex.org/wiki/Category:Resource:Gene_Ontology_Tools (en concreto buscamos dentro de la página la expresión "Statistical analysis") tenemos una buena muestra de qué podemos hacer y sobre todo con qué hacerlo.

[7] Y eso es un punto que no tiene solución.

[8] Los datos sin ningún tipo de preprocesamiento. Por ejemplo, si tenemos datos de expresión con Affymetric GeneChip se debiera de disponer de los ficheros .CEL.

[9] http://en.wikipedia.org/wiki/Literate_programming

[10] http://en.wikipedia.org/wiki/Reproducibility

[11] http://cran.r-project.org/web/views/ReproducibleResearch.html

diar.

Es de destacar que recientemente Nature Genetics ha refrendado el uso de Bioconductor.

Hemos de poder realizar un análisis de datos y generar un informe de un modo sencillo. Esto excluye el uso de herramientas como Word, Excel o similares. En este texto utilizaremos la opción R/Markdown/Quarto.

Se intenta explicar Estadística aplicada a la Bioinformática. Por ello es un texto que combina ambas cosas. En ocasiones se utiliza R/Bioconductor como herramienta pedagógica para ilustrar un concepto. En otras casi hacemos de manual técnico para usar un paquete de R. En principio, la idea es que vamos leyendo y ejecutando el código que se inserta. Lo lógico es tener R funcionando y, con un copiar y pegar, podemos ir ejecutando el código.[12]

En http://www.uv.es/ayala/docencia/tami/AllTami.R tenemos un script para instalar todos los paquetes que se utilizan en estas notas.[13] En https://www.uv.es/ayala/docencia/tami/#instalación se muestra cómo instalar los paquetes de R y Bioconductor y de algunos paquetes en Debian/Ubuntu necesarios para seguir el manual.

A lo largo de las notas utilizamos muchas funciones que corresponden a paquetes distintos. Indicaremos el nombre de la función y su paquete conjuntamente. Por ejemplo, con `annotate::annotation()` estamos refiriéndonos a la función `annotation` que se encuentra en el paquete [42, annotate].

Empezamos en § 1 indicándo qué se entiende por Estadística de datos ómicos.

En la parte I mostramos los datos con los que trabajamos posteriormente. Se comenta el tipo de dato, dónde se puede conseguir en bases de datos públicas y cómo preprocesarlo (normalizarlo). Finalmente se ve con cierto detalle los bancos de datos que utilizamos en el resto del libro. Los datos procesados y preparados para hacer análisis estadístico los tenemos en los paquetes [9, 10, 11, tamidata].[14] En el momento actual consideramos datos de expresión obtenidos con microarrays de DNA y con la técnica RNASeq.[15]

La parte **??** hablamos de Probabilidad y Estadística. Empezamos por un introducción muy básica para luego ir subiendo el nivel para poder tratar los problemas que se abordan posteriormente.

En la parte IV se estudian los problemas de comparaciones múltiples así como técnicas de expresión diferencial en microarrays y RNA-Seq.

En la parte IV nos ocupamos de la expresión diferencial gen a gen (o en lenguaje más estadístico, marginal).

En la parte V se estudia análisis de grupos de genes: cómo obtener las colecciones de grupos de genes, análisis de sobre representación y análisis de grupo de genes (o enriquecimiento).

En VI se habla de investigación reproducible.

Los aspectos relativos específicamente a R o Bioconductor son tratados en la parte VII.

Finalmente se incluyen apéndices sobre matrices, Probabilidad más avanzada, métodos numéricos y código que se referencia en el resto del texto.

Asociado a este manual tenemos la página https://www.uv.es/ayala/docencia/tami/ en donde se indica cómo instalar los paquetes necesarios. La versión más actualizada de este manual está en https:

[12] Solamente tener en cuenta que los caminos hay que modificarlos adaptándolos a donde tengamos los datos en nuestro ordenador.

[13] Quizás no es necesario instalarlo de una vez. Lo más conveniente es instalar los paquetes en la medida en que los necesitemos en los distintos capítulos.

[14] Son paquetes propios alojados en la Universidad de Valencia.

[15] Aunque se pretende ampliar a otros datos ómicos.

//www.uv.es/ayala/docencia/tami/tami13.pdf. He desarrollado algunos paquetes de apoyo: el paquete [7, tami], https://www.uv.es/ayala/software/tami_1.0.tar.gz, pretende que se pueda realizar un análisis de expresión diferencial marginal con dos grupos a comparar con un mínimo de aprendizaje. Los paquetes [9, 10, 11] contienen los datos que utilizamos en los ejemplos.

Es un material que se utiliza en tres cursos claramente diferenciados.

TAMI Uno es una breve introducción en 20 horas lectivas en tercer curso del grado de Bioctecnología de la Universidad de Valencia. En este curso se utilizan las herramientas más básicas, la opción simple y rápida. Se pretende ver cosas interesantes y mantener (en lo que se pueda) la programación con R/Bioconductor simple. Fue el origen de estas notas.

Estadística de datos ómicos Es una asignatura obligatoria de tercer curso del grado de Ciencia de Datos. Se utiliza un nivel básico con énfasis en los aspectos más computacionales y estadísticos.

Bioinformática Estadística El segundo curso es un módulo en Bioinformática Estadística en el master de Bioinformática de la Universidad de Valencia. Utilizamos todo el material.

La secuencia de lectura para **Bioinformática Estadística** así como las sesiones en las que lo tratamos es la siguiente:

1. El tema § 1 hace comentarios genéricos sobre este tipo de dato y en lo esencial se trata en la primera sesión.

2. El tema § 2 constituye el contenido de la segunda y parte de la tercera sesión.

Hacer y leer este manual es un auténtico tostón por lo que un poco de humor ayuda a digerirlo.

Parte I

Datos

Capítulo 1

Estadística y datos ómicos

1.1 Introducción

A diferentes procedimientos de obtención de información en Biología se les ha dado en llamar ómicas. El propio nombre como algo que lo engloba todo, que lo observa todo, es ya de por sí algo pretencioso. Las personas que vivan en los siglos que han de venir (y gracias a Dios no veremos) si son compasivos sonreirán ante esta denominación. Dato ómico es un tipo particular de dato de alta dimensión. Estas notas se ocupan de revisar y aplicar técnicas estadísticas a datos ómicos. No estamos interesados en el detalle exhaustivo del tratamiento de cada tipo de dato. Nos centraremos en lo que tienen en común y comentaremos lo que diferencia su tratamiento.

El énfasis de este texto es sobre **métodos estadísticos**. Evitaremos en lo posible métodos que no utilicen modelos probabilísticos. Lo que se ha dado en llamar métodos de **machine learning** o **Big Data**.[16] En este sentido es interesante recordar una frase (que subscribo) de un gran estadístico inglés Brian D. Ripley.

[16] He puesto a propósito los términos en inglés. Queda moderno. El autor no es un gran seguidor de esas técnicas nuevas. Soy un desfasado que solo cree en la Probabilidad. Sin modelo probabilístico no hay mucho. No hay nada.

```
fortunes::fortune(50)
```

```
To paraphrase provocatively, 'machine learning is
statistics minus any checking of models and
assumptions'.
   -- Brian D. Ripley (about the difference
      between machine learning and statistics)
      useR! 2004, Vienna (May 2004)
```

1.2 Estructura de los datos

En lo que sigue tendremos distintos tipos de datos con una estructura similar. Observaremos un gran número de características sobre un pequeño número de muestras. Por tanto tendremos muestras de dimensión alta.

Las características que observamos son de distintos tipo: fluorescencia cuando trabajemos con distintos tipos de microarrays como microarrays de DNA, metilación o microarrays de proteínas; número

de lecturas alineadas cuando hablamos de procedimientos de secuenciación; presencia o ausencia de una mutación cuando estudiemos asociación.[17] Esta característica puede estar asociada a una sonda o a un grupo de sondas en un microarray. O bien la información corresponde a un gen o a un exon, o a un péptido, a una proteína, o a una región genómica.

La característica con la que trabajamos en cada momento se cuantifica con distintos procedimientos (con frecuencia dependientes del fabricante del dispositivo). Hablaremos de características sin más. El número lo denotamos por N donde este valor es grande (miles). Estas características las observaremos en unas pocas muestras. El número de muestras es n (decenas con suerte). Lo básico[18] es que N es mucho mayor que n: $n \ll N$. En un contexto estadístico clásico N es el número de variables y n es el número de muestras. Y justo lo conveniente es la situación contraria. De hecho, muchos procedimientos estadísticos suponen que el tamaño de la muestra supera el número de variables. En Estadística de alta dimensión[19] esto no es así. Y eso da novedad a los procedimientos. Obviamente limita las posibilidades pero abre un nuevo campo de trabajo.

Las características las recogemos en una matriz[20] que denotaremos por

$$\boldsymbol{y} = [y_{ij}]_{i=1,\ldots,N;j=1,\ldots,n}$$

donde el valor y_{ij} nos cuantifica la característica i en la muestra j. [21]

Si y_{ij} corresponde a DNA microarray entonces mide un nivel de fluorescencia y tomará valores positivos.[22] Sea positivo a no un valor mayor indicará una mayor expresión del gen. Si trabajamos con datos obtenidos con RNA-Seq entonces tendremos conteos, esto es, número de lecturas cortas alineadas sobre un gen o sobre un exon o sobre una zona genómica de interés. En definitiva el dato primario es un número entero. Más lecturas indicará más expresión otra vez. Los valores observados en una misma fila (una misma características sobre todas las muestras) se suele decir que son, en transcriptómica, *perfil*[23] (de un modo genérico perfil de expresión).

En la matriz $\boldsymbol{y}$ los valores observados para las distintas muestras son *independientes* aunque posiblemente observados bajo distintas condiciones. No son pues réplicas de una misma condición experimental pero sí se observan independientemente. Son condicionalmente independientes. Sin embargo, las filas de $\boldsymbol{y}$ son realizaciones de vectores dependientes. Por ejemplo, los valores de expresión para las distintas filas no son independientes en una matriz de expresión ya que los genes actúan de un modo coordinado.

Habitualmente los datos de las columnas de la matriz $\boldsymbol{x}$ no son directamente comparables. Hay muchos artefactos técnicos así como ruido en la observación de la característica de interés. Se han desarrollado técnicas para corregirlo que llamaremos **preprocesado**. Veremos algunos procedimientos de preprocesado. En sentido estricto cuando realizamos estos los datos estos dejan de ser independientes. Sin embargo, esto no se suele considerar en la literatura. Los datos después de la normalización siguen considerándose independientes por columnas (muestras) y dependientes por filas.

De cada muestra tendremos información. Por ejemplo, si es una muestra control o bien corresponde a una muestra tomada bajo un tratamiento. [24] A esta información o variables que nos describen a las muestras las llamaremos los *metadatos* o *variables fenotípicas*[25]

[17] Y otros que el autor desconoce.

[18] Y a lo largo de las notas lo repetiremos muchas veces.

[19] High dimensional statistics.

[20] Que podemos llamar matriz de expresión o de metilación o de conteos o de mutaciones, aunque no necesariamente hablamos de expresión de un gen pero no parece malo utilizar esta nomenclatura de un modo genérico.

[21] Una matriz de datos en Estadística suele ser justo al contrario, esto es, con los subíndices invertidos. Es decir, la matriz transpuesta de la que vamos a utilizar aquí. En ocasiones se utiliza en la forma clásica pero es más frecuente esta forma de disponer los datos.

[22] Algunos procedimientos de procesado previo de los datos conocidos como normalización pueden dar lugar a expresiones negativas.

[23] Expression profile.

[24] La palabra tratamiento se utiliza en el sentido amplio de diseño experimental: tiempo, una cepa salvaje frente a una mutada por ejemplo.

[25] Entendido en un sentido

Usualmente tendremos varias variables fenotípicas. Denotaremos por $x = (x_1, \ldots, x_n)$ los valores observados de una variable en las n muestras. El caso maś frecuente de variable fenotípica será cuando tengamos dos grupos de muestras (casos y controles). En este caso tendremos $x_i = 1$ si es un caso e $x_i = 0$ si es un control.[26] Si tenemos más de dos grupos de muestras a comparar, por ejemplo k grupos, entonces $x_i \in \{1, \ldots, k\}$ para $i = 1, \ldots, n$.

[26] Los valores 0 y 1 son arbitrarios. Podemos tomar cualquier otro par de valores.

1.3 Problemas estadísticos

¿Y qué vamos a hacer con la matriz $\boldsymbol{y}$ y con las variables fenotípicas? Como siempre lo mejor que podamos. A veces no mucho. Las técnicas estadísticas que se utilizan son aplicaciones de procedimientos diseñados en muchas ocasiones para el contexto habitual en que tienes más muestra que variables. Y se usan aquí adaptándolos con mayor o menor fortuna. Yo diría con mayor o menos sentido en ocasiones.

Un problema fundamental que abordaremos es el que se conoce como **expresión diferencial**. Nos fijamos en una variable fenotipica y en una característica (gen por ejemplo). ¿Hay asociación entre el perfil de expresion y la variable fenotípica? Veremos distintos procedimientos para responder esta pregunta *para cada característica*. Utilizaremos la denominación de *análisis de expresión diferencial marginal*. Sin embargo, en la literatura biológica es más hablar de *análisis de expresión diferencial gen-a-gen*.

Las distintas características son dependientes entre si. Una evaluación de la posible asociación entre cada característica y la variable fenotípica es *limitada*. Nos puede hacer perder información sobre procesos biológicos de los cuales distintas características nos están dando información de modo que cada una de ellas recoge una variación no muy grande pero que conjuntamente es notable. En definitiva, el problema será estudiar la posible asociación entre grupos de características (grupos de filas en la matriz de expresión) y la variable fenotípica de interés. Es lo que se conoce como análisis de grupos de genes.[27]

[27] Este problema puede encontrarse bajo distintas denominaciones como: Gene set analysis, gene set enrichment analysis, functional enrichment testing.

También se verán problemas de reducción de dimensión. En concreto la técnica más utilizada, análisis de componentes principales. Es una técnica instrumental con muchas posibilidades de utilización en un contexto como es este con datos de alta dimensión.

Y clasificaremos tanto las características como las muestras como una herramienta exploratoria. Lo que se conoce como análisis cluster.

En lo que sigue se aborda también problemas de cálculo del tamaño muestral que está ligado al problema de la potencia del test.

Y todo esto siempre atendiendo al tipo de dato que estemos utilizando. Sin duda, el más desarrollado son las datos de expresión de gen utilizando microarrays. Será nuestro tipo de dato de referencia pero vamos introduciendo otros tipos de datos como pueden ser RNASeq o datos de abundancia de proteínas.

1.4 Bibliografía

A lo largo del manual se presta mucha atención a citar con precisión las referencias originales del material. La comprensión precisa de las técnicas supone la consulta de la referencia original.[28]

[28] Hay una peligrosa tendencia a creer que leyendo un resumen se conoce la técnica. No es cierto. Las referencias bibliográficas siempre hay que consultarlas.

Un libro que trata cómo hacer las cosas con R/Bioconductor pero no lo que se hace o porqué se hace es [80]. Es una guía de uso muy bien elaborada. Un texto con un objetivo similar al nuestro es [51]. Un manual online que sigue una línea muy similar a este es http://genomicsclass.github.io/book/.

Capítulo 2

Microarrays

2.1 Introducción

En este tema tratamos sobre datos de expresión obtenidos utilizando microarrays.[1] Nos ocupamos del procesado (o mejor, del preprocesado) de los datos desde los datos originales (o datos a nivel de sonda) hasta los datos tal y como los hemos estado analizando. En este tema nos ocupamos de este punto previo y no menor. Muchos de los análisis de expresión diferencial posterior dependen de un modo esencial de lo que hacemos antes. Con frecuencia el experimentador confía en el preprocesado que el fabricante del chip realiza. No necesariamente este preprocesado tiene porqué estar mal hecho pero en cualquier caso es preciso conocerlo y evaluarlo. En lo que sigue veremos como hay funciones que reproducen lo que el fabricante hace. En este capítulo es de interés los tres primeros capítulos de [44].

La expresión de un gen es el proceso mediante el que se transcribe el DNA en una serie de copias de mRNA. Los microarrays miden la cantidad de mRNA para cada gen. El valor de la expresión de un gen es una medida de luminiscencia relacionada con el mRNA presente. Esto es para un chip de DNA.

2.2 Sobre cómo usar un ExpressionSet

Los datos de microarrays de DNA suelen venir almacenados en la clase `Biobase::ExpressionSet`. En esta sección manejamos esta clase que es básica con este tipo de información.

2.2.1 sample.ExpressionSet

El experimento `Biobase::sample.ExpressionSet` tiene 26 muestras y 500 genes. Sobre las muestras conocemos tres variables (o covariables): `sex`, `type` (caso y control) y `score`. La última covariable es continua. Estos datos de expresión están almacenados en un objeto de clase `Biobase::ExpressionSet`. Para poder utilizar esta clase hemos de cargar el paquete [43, Biobase].

```
library(Biobase)
```

[1] La presentación asociada es https://www.uv.es/ayala/docencia/tami/presentaciones/t2_MicroarrayDNA_p.html.

Cargamos los datos.

```
data(sample.ExpressionSet)
```

¿De qué clase es el objeto?

```
class(sample.ExpressionSet)
```

```
[1] "ExpressionSet"
attr(,"package")
[1] "Biobase"
```

Las variables fenotípicas, que nos describen las distintas muestras con las que estamos trabajando, las podemos obtener con

```
phenoData(sample.ExpressionSet)
```

```
An object of class 'AnnotatedDataFrame'
  sampleNames: A B ... Z (26 total)
  varLabels: sex type score
  varMetadata: labelDescription
```

¿De qué clase es el objeto?

```
class(phenoData(sample.ExpressionSet))
```

```
[1] "AnnotatedDataFrame"
attr(,"package")
[1] "Biobase"
```

Vemos que esta clase está definida en [43, Biobase] y es de tipo

```
typeof(phenoData(sample.ExpressionSet))
```

```
[1] "S4"
```

Este sistema orientado a objetos es de uso mayoritario en Bioconductor ([89, capítulo 7]). Podemos acceder a los distintos `slots` que la componen con las funciones (accesors). Veamos cómo obtener los nombres de las muestras, los nombres de las variables fenotípicas e información adicional sobre estas variables fenotípicas.

```
sampleNames(sample.ExpressionSet)
```

```
 [1] "A" "B" "C" "D" "E" "F" "G" "H" "I" "J" "K"
[12] "L" "M" "N" "O" "P" "Q" "R" "S" "T" "U" "V"
[23] "W" "X" "Y" "Z"
```

```
varLabels(sample.ExpressionSet)
```

```
[1] "sex" "type" "score"
```

```
varMetadata(sample.ExpressionSet)
```

```
      labelDescription
sex Female/Male
type Case/Control
score Testing Score
```

¿Cómo podemos acceder al `Biobase::AnnotatedDataFrame` (notemos que no es un **`data.frame`**) que contiene la información?

```
head(pData(sample.ExpressionSet))
```

```
    sex type score
A Female Control 0.75
B Male Case 0.40
C Male Control 0.73
D Male Case 0.42
E Female Case 0.93
F Male Control 0.22
```

¿Cómo podemos acceder a una variable fenotípica dada, por ejemplo `type`? Tenemos varias opciones.

```
pData(sample.ExpressionSet)[,"type"]
pData(sample.ExpressionSet)$type
sample.ExpressionSet$type
```

La matriz de expresión (mostramos la primera fila) se obtiene con

```
head(exprs(sample.ExpressionSet),n=1)
```

```
                A B C D
AFFX-MurIL2_at 192.742 85.7533 176.757 135.575
                E F G H
AFFX-MurIL2_at 64.4939 76.3569 160.505 65.9631
                I J K L
AFFX-MurIL2_at 56.9039 135.608 63.4432 78.2126
                M N O P
AFFX-MurIL2_at 83.0943 89.3372 91.0615 95.9377
                Q R S T
AFFX-MurIL2_at 179.845 152.467 180.834 85.4146
                U V W X
AFFX-MurIL2_at 157.989 146.8 93.8829 103.855
               Y Z
AFFX-MurIL2_at 64.434 175.615
```

Lo que nos aparece a la izquierda son los identificadores de las sondas (en particular con sondas de control de Affymetrix). Podemos mostrar dos filas que no correspondan a sondas de control, por ejemplo, la que ocupa la fila 100 de la matriz de expresión del modo habitual.

```
exprs(sample.ExpressionSet)[100,]
```

```
      A B C D E
-28.99850 -30.05320 -26.97270 -23.00420 -18.31410
      F G H I J
316.92200 -21.24100 -14.67470 -22.34640 -26.95820
      K L M N O
-23.27410 -31.22210 -8.00193 -33.86130 -17.81590
      P Q R S T
-10.65560 -10.91990 -26.02170 -22.64670 -17.45640
      U V W X Y
-26.20010 -22.03100 -22.02760 -11.92950 -14.14710
      Z
-33.95400
```

Podemos ver a la izquierda de la salida anterior que aparecen los identificadores Affymetrix (PROBEID). ¿Dónde están almacenados?

```
rownames(sample.ExpressionSet)[100]
```

```
[1] "31339_at"
```

El sitio adecuado es el slot `featureNames`.

```
featureNames(sample.ExpressionSet)[100]
```

```
[1] "31339_at"
```

También tenemos un slot adicional en el que se pueden poner las correspondencias de estos identificacores con otras bases de datos. En este datos de ejemplo que manejamos no están definidos como podemos ver.

```
fData(sample.ExpressionSet)
```

```
data frame with 0 columns and 500 rows
```

¿Qué chip fue usado para obtener estos datos? Esto es necesario para conocer la correspondencia de nuestros identificadores con otros.

```
annotation(sample.ExpressionSet)
```

```
[1] "hgu95av2"
```

Finalmente veamos una breve descriptiva de las covariables que nos proporcionan información sobre las muestras y que utilizaremos para ilustrar estas notas. Para las categóricas vemos una tabla de frecuencias absolutas.

```
table(sample.ExpressionSet$sex)
```

```
Female Male
    11 15
```

```
table(sample.ExpressionSet$type)
```

```
  Case Control
    15 11
```

Y un resumen numérico de `score`.

```
summary(sample.ExpressionSet$score)
```

```
   Min. 1st Qu. Median Mean 3rd Qu. Max.
 0.1000 0.3275 0.4150 0.5369 0.7650 0.9800
```

Para tener información sobre un objeto de clase ExpressionSet es recomendable consultar [54, capítulo 2]. En la viñeta de [43, Biobase] tenemos una buena descripción de la creación y manipulación de la clase `Biobase::ExpressionSet`.

2.2.2 ALL

En esta sección pretendemos mostrar cómo utilizar funciones que manejan `ExpressionSet`. Vamos a utilizar los datos `ALL` que aparecen en el paquete [59, ALL]. Cargamos el paquete.

```
data(ALL,package="ALL")
```

Podemos ver que es un resumen de la información que tenemos en ALL.

```
ALL
```

```
ExpressionSet (storageMode: lockedEnvironment)
assayData: 12625 features, 128 samples
  element names: exprs
protocolData: none
phenoData
  sampleNames: 01005 01010 ... LAL4 (128
    total)
  varLabels: cod diagnosis ... date last
    seen (21 total)
  varMetadata: labelDescription
featureData: none
experimentData: use 'experimentData(object)'
  pubMedIds: 14684422 16243790
Annotation: hgu95av2
```

Podemos ver los niveles de expresión (en filas tenemos sondas y en columnas muestras). Los niveles de expresión han sido preprocesados y aparecen el logaritmo en base 2 de este valor ya preprocesado. Se obtienen (y no lo mostramos) con

```
exprs(ALL)
```

Los datos fenotípicos los tendremos con (tampoco los mostramos)

```
pData(ALL)
```

¿Cuál fue chip que se utilizó para obtener estos datos?

```
annotation(ALL)
```

```
[1] "hgu95av2"
```

Con `pData(ALL)` hemos visto los nombres (y los valores) de las covariables que nos describen las distintas muestras. Solamente los nombres los podemos ver con

```
names(pData(ALL))
```

Y si simplemente queremos ver los valores de la variable BT podemos hacerlo con

```
ALL$BT
```

Los datos de expressión aparecen en una matriz con filas (features que suelen corresponder a sondas) y con columnas (que corresponden con muestras). Vamos a seleccionar una parte de las muestras. En concreto, vamos a considerar las muestras tales que la variable fenotípica `mol.bio` toma el valor `NEG`. Determinamos las columnas.

```
selcol = ALL$mol.biol == "NEG"
```

Y nos quedamos con esa muestras.

```
ALL[,selcol]
```

Lo importante es ver que todo el `Biobase::ExpressionSet` tiene menos muestras y también se han modificado (eliminando) los datos de expresión y los datos fenotípicos.

2.3 Ejemplos

Comentaremos algunos de los bancos de datos de DNA microarrays que utilizamos en el resto del curso. Son datos de expresión de genes obtenidos utilizando microarrays de DNA. También se muestra en este tema el uso de la clase `Biobase::ExpressionSet`. Esta clase permite el manejo de datos de expresión obtenidos con microarrays de DNA.

2.3.1 ALL

Para una versión ampliada de esta sección podemos consultar [54, capítulo 1]. Los datos ALL son microarrays de 128 individuos distintos con leucemia linfoblástica aguda (ALL). De estos individuos 95 corresponden a leucemia linfoblástica precursora aguda de células B y y 33 son leucemia linfoblástica precursora aguda de células T. Son enfermedades bastante distintas y por ello se consideran por separado. Habitualmente trabajaremos con las muestras de leucemia linfoblástica precursora aguda de células B. Los datos han sido preprocesados utilizando el método RMA (robust multichip average) implementado en el paquete [56, affy] con la función `affy::rma` y están almacenados en forma de un `Biobase::ExpressionSet`. Empezamos cargando los datos.

```
pacman::p_load(Biobase,ALL)
data(ALL)
```

En lo que sigue no vamos a utilizar todas las muestras. Un subconjunto de muestras de interés con dos grupos son los grupos de tumores de células B que tienen la mutación BCR/ABL y los tumores de las células B sin ninguna anormalidad citogenética. Veamos cómo seleccionar estas muestras. En primer lugar seleccionamos las muestras de células B.

```
bcell = grep("^B",as.character(ALL$BT))
```

Y ahora seleccionamos las muestras correspondientes a los tipos moleculares BCR/ABL o NEG.

```
types = c("NEG","BCR/ABL")
moltyp = which(as.character(ALL$mol.biol) %in% types)
```

Ahora combinamos ambas selecciones para quedarnos con los tumores de las células B y que tienen o bien la translocación BCR/ABL o bien no tienen ninguna de las anormalidades moleculares evaluadas.

```
bcrneg = ALL[,intersect(bcell,moltyp)]
```

Habitualmente haremos uso de los datos ALL realizando previamente esta selección. Remitiremos a esta sección para consultar el código anterior.

2.3.2 Un experimento con levadura

Los datos que comentamos en esta sección son un experimento en donde vamos a considerar dos tipos de células en levadura, salvaje y mutada. Los datos los tenemos en un `Biobase::ExpressionSet` en [9].

```
data(gse6647,package="tamidata")
```

El número de genes y muestras es

```
dim(gse6647)
```

```
Features Samples
   6103 8
```

Podemos ver los datos fenotípicos.

```
head(pData(gse6647),n=2)
```

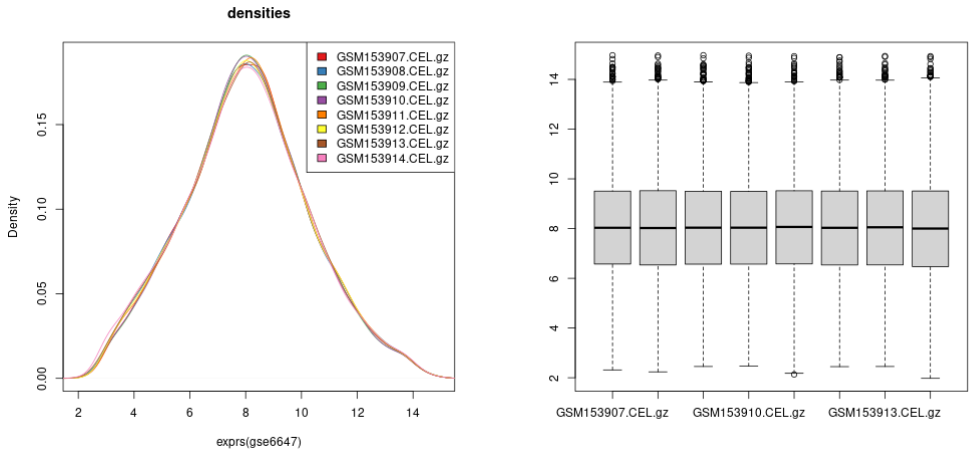
```
                type
GSM153907.CEL.gz wt
GSM153908.CEL.gz edc3D
```

Los datos han sido normalizados utilizando el método RMA. En la figura 2.1(a) mostramos las densidades estimadas.

```
geneplotter::multidensity(exprs(gse6647))
```

En la figura 2.1(b) tenemos los diagramas de cajas.

```
boxplot.matrix(exprs(gse6647))
```



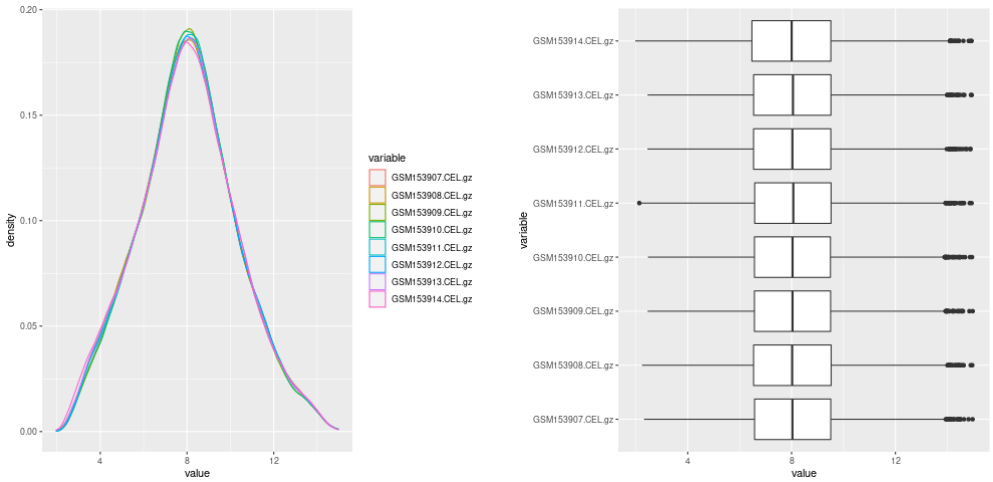
(a) (b)

Figura 2.1: Datos gse6647: estimadores de densidad y diagramas de cajas.

Vamos a reproducir los dibujos de la figura 2.1 utilizando el paquete [92, ggplot2].[29] El código es el que sigue y los tenemos en la figura 2.2.

```
pacman::p_load(reshape,ggplot2)
df = data.frame(gene = featureNames(gse6647),exprs(gse6647))
df1 = melt(df,id=c("gene"))
ggplot(df1,aes(x=value,colour=variable,group=variable)) +
    geom_density(kernel = "epanechnikov",fill=NA)
ggplot(df1,aes(x=variable,y = value)) + geom_boxplot() +
    coord_flip()
```

[29] Sobre el paquete [90, reshape] es interesante consultar [91].



(a) (b)

Figura 2.2: Datos gse6647: estimadores de densidad y diagramas de cajas utilizando [92, ggplot2].

2.3.3 Correspondencias múltiples

Necesitamos conocer la correspondencia entre sondas y genes para poder interpretar resultados o realizar análisis posteriores. Habitualmente recurrimos a paquetes de anotación que nos proporcionan la información de la correspondencia entre los identificadores del fabricante (`PROBEID`) y los identificadores en distintas bases de datos o denominaciones del gen (o entidad biológica que estemos utilizando). La opción más simple y primera a probar es utilizar un paquete de anotación. En Bioconductor es conveniente consultar https://www.bioconductor.org/packages/release/data/annotation/. En § 18 tratamos el problema del manejo de este tipo de paquetes.

En esta sección tratamos un problema al que se suele prestar poca atención pero es muy importante, las correspondencias múltiples. Distintas sondas corresponden a un mismo gen y distintos genes corresponden a una misma sonda. Esto es, no tenemos una biyección o correspondencia 1-1 entre sondas y genes. Y no es un problema menor. Vamos a ilustrar el problema con un ejemplo. Cargamos los datos bajados sin ningún tipo de normalización previa.

```
load(paste0(dirTamiData,"gse21779raw.rda"))
gse21779rma = affy::rma(gse21779raw)
```

¿Cuántas sondas tenemos después de la normalización?

```
nrow(gse21779rma)
```

```
Features
   54675
```

Establecemos las correspondencias con Entrez con `AnnotationDbi::select()`.

```
pacman::p_load(hgu133plus2.db)
a = AnnotationDbi::select(hgu133plus2.db,
                          keys=featureNames(gse21779rma),
                          columns=c("ENTREZID"),
                          keytype="PROBEID")
```

Vemos que tenemos más filas en `a` que en `gse21779rma`.

```
dim(a)
```

```
[1] 57156 2
```

Tenemos las correspondencias en forma de **`data.frame`**. ¿Cuántas y qué sondas tienen correspondencia con más de un gen?

```
b = which(table(a[,1]) > 1)
```

```
sel = is.element(a[,1],names(b))
table(table(a[sel,1]))
```

```
  2 3 4 5 6 7 8 9 10 11
1202 153 73 30 16 14 11 11 3 4
  12 13 14 15 17 21 22
   2 1 2 3 2 1 4
```

Una posibilidad puede ser quedarnos con la primera aparición.

```
c1 = match(unique(a[,1]),a[,1])
a1 = a[c1,]
```

Vamos a añadir esta información en los datos originales. Notemos que allí las sondas son únicas y estamos asignando la primera correspondencia que nos aparece.

```
fData(gse21779rma) = a1
```

¿Lo tenemos todo resuelto?

2.4 Ejercicios

* **Ex. 1** — Consideramos los datos `tamidata::gse28619`. Se pide:

1. ¿Cuántas sondas tenemos?
2. ¿Cuántas muestras?
3. Las sondas de control empiezan con "AFFX": ¿Cuántas sondas de control tenemos?
4. ¿Cuántas variables fenotípicas tenemos? ¿De qué tipo son?
5. Indica el valor de expresión de la sonda en fila 2000 y columna 3.
6. Calcular para la sonda en la fila 100 la expresión media.
7. Calcular para la sonda en la fila 100 la expresión media en cada uno de los grupos definidos por la variable fenotípica `type`.
8. Determinar la sonda con una expresión media mayor?
9. Determinar la muestras con un nivel de expresión medio menor?
10. Consideremos el slot `fData`.
    (a) ¿Qué información tenemos en el slot `fData`?
    (b) ¿De qué clase es el slot `fData`?
    (c) ¿Cuántas columnas componen el slot `fData`?

* **Ex. 2** — Consideramos los datos `tamidata::gse44456`. Repetir el ejercicio 1 con estos datos.

** **Ex. 3** — Se pide construir un `Biobase::ExpressionSet`. Este `Biobase::ExpressionSet` ha de tener la siguiente matriz de expresión.

```
      [,1] [,2] [,3] [,4] [,5]
 [1,] 24.61 22.70 23.49 24.93 25.27
 [2,] 22.93 22.24 24.92 22.52 21.94
 [3,] 22.28 24.38 21.58 20.65 22.68
 [4,] 22.05 23.46 23.68 23.56 23.50
 [5,] 23.05 22.72 24.52 26.27 23.68
 [6,] 25.28 21.99 23.90 23.25 24.01
 [7,] 24.43 22.47 21.16 21.94 21.77
 [8,] 25.56 24.47 24.10 21.48 23.15
 [9,] 21.81 22.77 23.61 24.20 24.92
[10,] 23.20 24.81 21.47 23.55 24.96
[11,] 24.46 23.03 22.07 22.93 23.68
[12,] 22.77 24.98 23.92 22.84 21.27
[13,] 24.14 22.64 24.34 23.31 20.97
[14,] 22.50 23.22 22.55 21.98 21.00
[15,] 22.09 21.97 20.80 22.86 22.66
[16,] 23.60 25.27 21.84 22.61 22.63
[17,] 22.65 24.99 24.83 22.91 22.56
[18,] 23.24 21.78 22.57 23.24 22.28
[19,] 22.77 21.55 24.58 22.53 22.50
[20,] 23.62 25.35 21.55 24.17 22.07
```

Los nombres de las filas ha de ser

```
 [1] "g1" "g2" "g3" "g4" "g5" "g6" "g7"
 [8] "g8" "g9" "g10" "g11" "g12" "g13" "g14"
[15] "g15" "g16" "g17" "g18" "g19" "g20"
```

Los nombres de las muestras serán

```
 [1] "m 1" "m 2" "m 3" "m 4" "m 5" "m 6"
 [7] "m 7" "m 8" "m 9" "m 10" "m 11" "m 12"
[13] "m 13" "m 14" "m 15" "m 16" "m 17" "m 18"
[19] "m 19" "m 20"
```

Como datos fenotípicos (covariables que describen las columnas) han de ser las siguientes

```
  tipo crecimiento
1 1 0.30
2 2 0.23
3 2 0.34
4 1 0.24
5 2 0.45
```

** **Ex. 4** — Se pide construir a partir de los datos `multtest::golub` ↪ un `Biobase::ExpressionSet`.

** **Ex. 5** — [2] Vamos a bajar y preprocesar los datos `GSE30129` de GEO. Se pide realizar los siguientes pasos.

1. Bajar los datos a nivel de sonda utilizando el paquete [29, GEOquery] y la función `GEOquery::getGEOSuppFiles()`.
2. Leer los datos utilizando la función `oligo::read.celfiles()`.
3. En el paso anterior si no tenemos instalado el paquete `pd.mogene` ↪ `.1.0.st.v1` nos dará un aviso.
4. Instalar el paquete indicado y repetir la lectura.
5. Representar los estimadores de densidades y los diagramas de cajas de las expresiones a nivel de sonda.

[2] Este problema es muy similar a lo que hacemos en **??**.

6. Obtener los MAplots o dibujos media-diferencia de Tukey.
7. Aplicar el método RMA.
8. Repetir los dibujos de los apartados 5 y 6 para los datos procesados utilizando el método RMA.

* **Ex. 6** — Consideremos los datos `tamidata::gse20986raw`.

1. ¿Cómo identifica los genes de las filas?
2. Cambiar los identificadores de los genes a sus códigos Ensembl.

Capítulo 3

RNA-seq

3.1 Introducción

En este tema trabajamos con datos obtenidos mediante la técnica conocida como RNA-Seq. En http://rnaseq.uoregon.edu/ tenemos una introducción muy simple y clara. Una buena visión general la tenemos en [71]. Un texto general que trata lo relativo al análisis de este tipo de datos es [57]. Una referencia breve pero de interés es [28] donde también da una visión global del análisis de este tipo de información.

En este manual se asume que se trabaja con un genoma de referencia. Los procedimientos que vamos a estudiar asumen este punto de partida.

De un modo análogo a § 2 comentaremos este tipo de información de expresión génica. En este capítulo mostramos algunos conceptos básicos y algunas herramientas útiles (y suficientes) para su manejo. El software y las posibilidades que ofrece es enorme y fuera del alcance de este manual (y de su autor). Pretendemos ofrecer una opción (o dos) sencillas y buenas para poder bajar los datos de un experimento y preprocesarlos hasta obtener la matriz de conteos. Una vez tengamos esa matriz de conteos y las variables fenotípicas podremos analizar nuestros datos que es el objetivo fundamental de este manual.

Una referencia de mucha utilidad para consultar cómo se obtienen y preprocesan este tipo de datos es [52].

3.2 Formatos

En § 3.2.1 y § 3.2.2 comentamos distintos formatos para almacenar secuencias tanto de ácidos nucleicos como de proteínas.

3.2.1 Formato FASTA

El formato FASTA está basado en texto y se utiliza representar secuencias bien de nucleótidos bien de aminoácidos. Tanto unos como otros son representados por una sola letra. También tiene símbolos para representar un hueco (gap) o parada en la traducción o bien que no se sabe el nucleótido o aminoácido. Es muy simple. Tiene una línea que comienza con el símbolo > al que sigue una descripción de la secuencia. En la siguiente línea empieza la secuencia de bases o

aminoácidos. Se recomienda que no tener más de 80 columnas y se pueden tener todas las filas que se precisen.

3.2.2 Formato FASTQ

El formato FASTQ es el más popular para datos de secuencias. Consiste de cuatro líneas por lectura con la siguiente información:

1. La primera que comienza con el carácter @ y contiene el nombre de la secuencia con alguna descripción opcional de la misma.
2. La segunda línea contiene la secuencia con las letras que correspondan dependiendo del tipo (nucleótidos, aminoácidos).
3. La tercera línea que comienza con + contiene información opcional sobre la secuencia.
4. La cuarta línea cuantifica la confianza o calidad en la determinación de cada base recogida en la segunda línea. En § 3.2.3 se comenta el índice Phred y su codificación.
5.

Un ejemplo es el siguiente:

```
@SRR1293399.1 ILLUMINA-545855_0026_FC629BG:6:1:1022:5049 length=50
ACAGGGACGCCATCGAATCCGGATCNTNNNNNNNNNNNNANNNNNNNNNN
+SRR1293399.1 ILLUMINA-545855_0026_FC629BG:6:1:1022:5049 length=50
dee\edYcdc`bbY`S]bb_]]Ua^BBBBBBBBBBBBBBBBBBBBBBBBB
```

3.2.3 Phred

¿Cómo se cuantifica la confianza o precisión o calidad de la cada una de las bases que tenemos en la secuencia?[30] Se utiliza el programa Phred.[31] Este programa lo que hace es asignar los picos de fluorescencia a una de las cuatro bases (o *base call*). En [35] tenemos la explicación del método de asignación. Si P denota la probabilidad[32] para una base dada de ser mal asignada o clasificada entonces el valor con el que se trabaja es

$$Q = -10 \log_{10} P. \tag{3.1}$$

Esto significa que una probabilidad P muy pequeña de clasificación incorrecta se traduce en un valor grande de Q. Por ejemplo, una probabilidad de clasificación incorrecta de 0.01 corresponde con un valor de Q de

```
−10 ∗ log10(0.01)
```

```
[1] 20
```

Supongamos que tenemos la siguiente secuencia con sus correspondientes Q valores.

```
         G T T
3.927473e-04 4.391522e-04 3.250830e-04
         T T C
5.390653e-04 9.702025e-04 5.957912e-04
         A T C
7.194114e-04 3.176761e-04 1.333130e-04
         A
6.040676e-05
```

[30] Observar que en ningún momento utilizo (tomo en vano) la palabra Probabilidad.

[31] El procedimiento que utiliza aparece en [36] y cómo estima las probabilidades de error cuando asigna cada base aparece en [35].

[32] No es realmente una probabilidad. Es una cuantificación de la calidad de la asignación y no más.

Los correspondientes Q valores serían

```
 G T T T T C A T C A
34 34 35 33 30 32 31 35 39 42
```

Si tenemos que guardar estos valores tendríamos que almacenar los dos dígitos y el blanco que los separa. Mucha memoria cuando tenemos muchas bases a almacenar. En lugar de registrar las probabilidades de clasificación incorrecta o su transformación Q lo que se utiliza es la codificación Sanger. Consiste en guardar el valor ASCII correspondiente a la posición $33 + Q$. Los caracteres ASCII correspondientes a los valores del 33 al 100 los podemos obtener con

```
intToUtf8(33:100)
```

```
[1] "!\"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\\]^_`
    ↪ abcd"
```

En particular, para el ejemplo previo tendríamos la siguiente codificación.

```
[1] "CCDB?A@DHK"
```

A lo largo del tiempo el valor 33 que corresponde con $Q = 0$ tomó diferentes valores. En principio, solamente para datos antiguos podemos encontrar que lo dicho no es válido (por ejemplo, sumando 64 en lugar de 33 o no utilizando ciertos valores). Claramente la codificación indicada no está diseñada para que nosotros la leamos sino como una forma de ahorrar memoria.

3.3 Ejemplos

En esta sección mostramos distintos bancos de datos bien contenidos en paquetes de Bioconductor (§ 3.3.1) o bien que podemos obtener de bases de datos en línea.

3.3.1 parathyroidSE::GSE37211

Estos datos corresponden al experimento con número de acceso GEO `GSE37211`. En la viñeta del paquete [53, parathyroidSE] tenemos una descripción detallada para obtener los conteos a nivel de gen o de exon partiendo de los ficheros originales NCBI-SRA. Ya los tenemos en el paquete [53, parathyroidSE].

```
data(parathyroidGenesSE,package="parathyroidSE")
```

Son datos relativos a los genes. ¿Qué clase tenemos?

```
class(parathyroidGenesSE)
```

```
[1] "RangedSummarizedExperiment"
attr(,"package")
[1] "SummarizedExperiment"
```

Es un `SummarizedExperiment::RangedSummarizedExperiment` y será nuestra clase de referencia cuando trabajamos con datos de RNA-Seq.[33] Para poder trabajar con esta clase cargamos el paquete.

```
pacman::p_load(SummarizedExperiment)
```

¿Cuántos genes y muestras tenemos?

[33] Es muy conveniente leer la viñeta de [68, SummarizedExperiment].

```
dim(parathyroidGenesSE)
```

```
[1] 63193 27
```

La matriz con los conteos nos la da `GenomicRanges::assay()` Es la análoga a `Biobase::exprs()` para `Biobase::ExpressionSet`. .

```
head(assay(parathyroidGenesSE),n=2)
```

```
                [,1] [,2] [,3] [,4] [,5] [,6]
ENSG00000000003 792 1064 444 953 519 855
ENSG00000000005 4 1 2 3 3 1
                [,7] [,8] [,9] [,10] [,11] [,12]
ENSG00000000003 413 365 278 1173 463 316
ENSG00000000005 0 1 0 0 0 0
                [,13] [,14] [,15] [,16] [,17]
ENSG00000000003 987 424 305 391 586
ENSG00000000005 0 0 0 0 0
                [,18] [,19] [,20] [,21] [,22]
ENSG00000000003 714 957 346 433 402
ENSG00000000005 0 1 0 0 0
                [,23] [,24] [,25] [,26] [,27]
ENSG00000000003 277 511 366 271 492
ENSG00000000005 0 0 0 0 0
```

Tenemos los metadatos de las columnas o variables fenotípicas o covariables asociadas a las muestras con `GenomicRanges::coldata()`. Corresponde con `Biobase::pData()`.

```
colData(parathyroidGenesSE)
```

Es un `S4Vectors::DFrame`.

```
class(colData(parathyroidGenesSE))
```

```
[1] "DFrame"
attr(,"package")
[1] "S4Vectors"
```

Por ejemplo, podemos ver los nombres de las variables fenotípicas con

```
names(colData(parathyroidGenesSE))
```

```
[1] "run" "experiment" "patient"
[4] "treatment" "time" "submission"
[7] "study" "sample"
```

Y acceder a los valores de la variable `treatment` con

```
colData(parathyroidGenesSE)[,"treatment"]
```

```
 [1] Control Control DPN DPN OHT
 [6] OHT Control Control DPN DPN
[11] DPN OHT OHT OHT Control
[16] Control DPN DPN OHT OHT
[21] Control DPN DPN DPN OHT
[26] OHT OHT
Levels: Control DPN OHT
```

o bien con (no mostrado)

```
colData(parathyroidGenesSE)$treatment
```

La información en las filas en este caso corresponde con genes.

```
rowRanges(parathyroidGenesSE)
```

```
GRangesList object of length 63193:
$ENSG00000000003
GRanges object with 17 ranges and 2 metadata columns:
      seqnames ranges strand |
         <Rle> <IRanges> <Rle> |
   [1] X 99883667-99884983 - |
   [2] X 99885756-99885863 - |
   [3] X 99887482-99887565 - |
   [4] X 99887538-99887565 - |
   [5] X 99888402-99888536 - |
   ... ... ... ... .
  [13] X 99890555-99890743 - |
  [14] X 99891188-99891686 - |
  [15] X 99891605-99891803 - |
  [16] X 99891790-99892101 - |
  [17] X 99894942-99894988 - |
       exon_id exon_name
      <integer> <character>
   [1] 664095 ENSE00001459322
   [2] 664096 ENSE00000868868
   [3] 664097 ENSE00000401072
   [4] 664098 ENSE00001849132
   [5] 664099 ENSE00003554016
   ... ... ...
  [13] 664106 ENSE00003512331
  [14] 664108 ENSE00001886883
  [15] 664109 ENSE00001855382
  [16] 664110 ENSE00001863395
  [17] 664111 ENSE00001828996
  -------
  seqinfo: 580 sequences (1 circular) from an unspecified genome

...
<63192 more elements>
```

Por tanto, las filas de un `SummarizedExperiment` es un `GenomicRanges` ↪ `::GRangesList` de modo que cada fila es un `GenomicRanges::` ↪ `GRanges` indicando los exones que se utilizaron para contar las secuencias de RNA. Los nombres de los genes los podemos obtener con

```
head(rownames(parathyroidGenesSE))
```

3.3.2 TCGA

La base de datos del TCGA contiene estudios de RNA-Seq. Se pueden bajar los datos utilizando GDC Data Transfer Tool. El paquete [27, TCGAbiolinks] facilita el uso de esta enorme base de datos. Además podemos conseguir datos de tipo muy distinto. En esta sección lo hacemos para bajar datos de RNA-Seq.

Datos TCGA-COAD

En esta sección vamos a utilizar, entre otros, datos procedentes de TCGA. Empezamos bajando los datos.

```
pacman::p_load(TCGAbiolinks)
query <-GDCquery(
  project = "TCGA-COAD",
  data.category = "Gene expression",
  data.type = "Gene expression quantification",
```

```
  file.type="normalized_results",
  platform = "Illumina HiSeq",
  experimental.strategy = "RNA-Seq",
  legacy = TRUE
)
GDCdownload(query, method = "api", files.per.chunk = 10)
tcga_coad = GDCprepare(query)
```

Para manejar la clase necesitamos el paquete [68, SummarizedExperiment].

```
pacman::p_load(SummarizedExperiment)
```

Eliminamos aquellas variables fenotípicas que tengan más de 7 datos faltantes así como aquellas que no son útiles desde el punto de vista clínico o bien porque son constantes.

```
toremain = which(apply(is.na(colData(tcga_coad)),2,sum)
                 <7)
names_toremain = names(colData(tcga_coad))[toremain]
names_toremove = c("barcode","patient","sample",
                   "sample_submitter_id",
                   "sample_id","state",
                   "pathology_report_uuid",
                   "submitter_id","is_ffpe",
                   "tissue_type",
                   "synchronous_malignancy",
                   "treatments",
                   "last_known_disease_status",
                   "classification_of_tumor",
                   "diagnosis_id",
                   "tumor_grade","alcohol_history",
                   "exposure_id","demographic_id",
                   "bcr_patient_barcode",
                   "project_id")
names_toremain = setdiff(names_toremain,names_toremove)
colData(tcga_coad) = colData(tcga_coad)[,names_toremain]
## Definimos tipos
tofactor = c(1:4,6,8,9,11:22,26:30)
colData(tcga_coad)[,tofactor] =
    lapply(colData(tcga_coad)[,tofactor],as.factor)
colData(tcga_coad)[,-tofactor] =
    lapply(colData(tcga_coad)[,-tofactor],as.numeric)
save(tcga_coad,file=paste0(dirTamiData,"tcga_coad.rda"))
```

```
load(paste0(dirTamiData,"tcga_coad.rda"))
```

Tenemos 19947 genes y 328 muestras.

Parte II

Fundamentos estadísticos y modelos lineales

Capítulo 4

Conceptos fundamentales de Estadística

El contenido tratado en los temas anteriores es intencionadamente muy básico. Se pretende llegar al lector que no tiene ningún conocimiento de Probabilidad y Estadística. Sin embargo, hay procedimientos que se tratan en este manual y que requieren un nivel de formalización mayor. En este tema se aborda este otro nivel. Para una primera lectura del manual no se requiere. Pero sí para una lectura completa. El lenguaje utilizado es más preciso.

4.1 Verosimilitud

Sea $y = (y_1, \ldots, y_n)$ una realización del vector aleatorio $Y = (Y_1, \ldots, Y_n)$. Es habitual asumir que Y tiene una función de densidad conjunta f en una cierta familia $\mathcal{F}$. Para una función dada f, el valor $f(y)$ nos muestra cómo varía la densidad dentro del espacio muestral de valores posibles de y. Y viceversa, si consideramos unos datos y y lo que hacemos variar es la función de densidad entonces estamos viendo cómo de verosímil es cada una de las funciones dados los datos y. Esta función recibe el nombre de **verosimilitud** de f dados los datos y y se suele denotar como

$$L(f; y) = f(y). \tag{4.1}$$

Con frecuencia, es conveniente trabajar con el logaritmo natural de la función anterior y hablaremos de la **log-verosimilitud**.

$$\ell(f; y) = \ln f(y). \tag{4.2}$$

Una simplificación adicional (que es habitual en las aplicaciones) supone que la función de densidad f pertenece a una familia paramétrica $\mathcal{F}$, esto es, cada elemento de la familia es conocido completamente salvo un número finito de parámetros $\theta = (\theta_1, \ldots, \theta_p)$ de modo que denotaremos $f(y; \theta)$, $f_Y(y; \theta)$ o $f(y|\theta)$.

Al conjunto de valores posibles de θ se le llama **espacio paramétrico** y lo denotaremos por Θ. En este caso, la logverosimilitud es

una función de θ y denotaremos

$$\ell(\theta; y) = \ln f(y; \theta). \tag{4.3}$$

Si asumimos que los distintos $Y_1, \ldots, Y_n$ son independientes entonces

$$L_Y(\theta; y) = f_Y(y) = \prod_{i=1}^{n} f_{Y_i}(y_i),$$

y

$$\ell_y(\theta; y) = \sum_{i=1}^{n} \ln f_{Y_i}(y_i) = \sum_{i=1}^{n} \ell(\theta; y_i).$$

Veamos algunos ejemplos de verosimilitud con modelos que usamos posteriormente.

Ejemplo 4.1 (Pruebas Bernoulli)**.** *$Y_1, \ldots, Y_n$ son independientes y con la misma distribución (i.i.d.) $P(Y_i = y_i) = \theta^{y_i}(1-\theta)^{1-y_i}$ y*

$$L(\theta; y) = \theta^{\sum_{i=1}^{n} y_i}(1-\theta)^{n-\sum_{i=1}^{n} y_i}$$

Ejemplo 4.2 (Número de éxitos en n pruebas Bernoulli)**.** *Nuestros datos son ahora el número total de éxitos en un número dado de pruebas de Bernoulli, r. Entonces la variable correspondiente R tiene una distribución binomial con n pruebas y una probabilidad de éxito θ. La verosimilitud viene dada por*

$$L(\theta; r) = \binom{n}{r}\theta^r(1-\theta)^{n-r}$$

Ejemplo 4.3 (Binomial negativa)**.** *Nuestros datos son ahora el número total de pruebas necesarias para alcanzar un número previamente especificado de éxitos. La variable aleatoria correspondiente N tendrá una distribución binomial negativa con r éxitos y una probabilidad de éxito θ. La función de verosimilitud correspondiente viene dada por*

$$L(\theta; n) = \binom{n-1}{r-1}\theta^r(1-\theta)^{n-r}$$

Consideremos los tres ejemplos anteriores 4.1, 4.2 y 4.3.

4.2 Estimación

Denotamos por Θ el espacio formado por los valores que puede tomar θ o espacio paramétrico.

Definición 4.1. *Un **estimador** del parámetros o vector paramétrico θ es cualquier función de la muestra $X_1, \ldots, X_n$ que toma valores en el espacio paramétrico.*

Si $\delta(X_1, \ldots, X_n)$ es un estimador del parámetro θ entonces se define el **error cuadrático medio** como

$$MSE(\delta) = E[\delta(X_1, \ldots, X_n) - \theta]^2 \tag{4.4}$$

En el caso en que se verifique que $E\delta(X_1, \ldots, X_n) = \mu_\delta = \theta$, es decir, que el estimador sea **insesgado** entonces:

$$MSE(\delta) = E[\delta(X_1, \ldots, X_n) - \theta]^2 = E[\delta(X_1, \ldots, X_n) - \mu_\delta]]^2 = var(\delta).$$

Para estimadores insesgados el error cuadrático medio no es más que la varianza del estimador. Consideremos la siguiente cadena de igualdades. Denotamos

$$MSE(\delta) = E[\delta - \theta]^2 = E[\delta - \mu_\delta + \mu_\delta - \theta]^2 = E[\delta - \mu_\delta]^2 + [\mu_\delta - \theta]^2 \quad (4.5)$$

La diferencia entre la media del estimador y el parámetro, $\mu_\delta - \theta$, recibe el nombre de **sesgo**. Finalmente lo que nos dice la ecuación anterior es que el error cuadrático medio $MSE(\delta)$ lo podemos expresar como la suma de la varianza del estimador, $E[\delta - \mu_\delta]^2$, más el sesgo al cuadrado, $[\mu_\delta - \theta]^2$.

A la raíz cuadrada de la varianza de un estimador, es decir, a su desviación típica o estándar se le llama **error estándar**. La expresión error estándar se usa en ocasiones indistintamente para referirse o bien dicha desviación típica o bien al estimador de la misma.

4.2.1 Estimación insesgada de media y varianza

Dada una muestra $Y_1, \ldots, Y_n$ de una variable. Un estimador habitualmente utilizado para estimar $\mu = EY_i$ es la media muestral dada por

$$\bar{Y} = \frac{1}{n} \sum_{i=1}^{n} Y_i. \quad (4.6)$$

Notemos que

$$E\bar{Y} = E\left[\frac{1}{n} \sum_{i=1}^{n} Y_i\right] = \frac{1}{n} \sum_{i=1}^{n} EY_i = \frac{1}{n} \sum_{i=1}^{n} \mu = \mu.$$

En definitiva, la media muestral es un estimador que no tiene ningún sesgo cuando estima la media de Y_i (la media poblacional) o, lo que es lo mismo, es un estimador **insesgado**.

Para estimar de un modo insesgado la varianza σ^2 a partir de una muestra $Y_1, \ldots, Y_n$ se utiliza la varianza muestral dada por

$$S^2 = \frac{1}{n-1} \sum_{i=1}^{n} (Y_i - \bar{Y})^2. \quad (4.7)$$

La razón de la división por $n-1$ en lugar de dividir por n viene de las siguientes igualdades.

$$E \sum_{i=1}^{n} (Y_i - \bar{Y})^2 = E \sum_{i=1}^{n} [(Y_i - \mu) - (\bar{Y})^2 - \mu)]^2 = \sum_{i=1}^{n} E(Y_i - \mu)^2 - nE(\bar{Y} - \mu)^2, \quad (4.8)$$

pero $E(Y_i - \mu)^2 = \sigma^2$ y $E(\bar{Y} - \mu)^2 = var(\bar{Y}) = \sigma^2/n$. En consecuencia,

$$E \sum_{i=1}^{n} (Y_i - \bar{Y})^2 = n\sigma^2 - \frac{\sigma^2}{n} = n\sigma^2 - \sigma^2 = (n-1)\sigma^2,$$

de donde,

$$ES^2 = E \frac{1}{n-1} \sum_{i=1}^{n} (Y_i - \bar{Y})^2 = \sigma^2,$$

es decir, S^2 estima la varianza σ^2 sin sesgo.

4.2.2 Estimación insesgada del vector de medias y la matriz de covarianzas

Ahora consideramos una muestra de un vector de dimensión d, $Y_1, \ldots, Y_n$ i.i.d. con vector de medias $\mu = EY_i$ y matriz de covarianzas $\Sigma = cov(Y_i)$. Los estimadores insesgados de μ y Σ son las versiones multivariantes de la media y varianza muestrales. Si

$$Y_i = \begin{bmatrix} Y_{i1} \\ \vdots \\ Y_{ip} \end{bmatrix}$$

Entonces podemos representar toda la muestra como la siguiente matriz

$$\boldsymbol{Y} = \begin{bmatrix} \boldsymbol{Y}_1^T \\ \vdots \\ \boldsymbol{Y}_n^T \end{bmatrix} = \begin{bmatrix} Y_{11} & \ldots & Y_{1d} \\ \vdots & \vdots & \vdots \\ Y_{n1} & \ldots & Y_{nd} \end{bmatrix}$$

mientras que los datos observados, la matriz de datos, vendría dada por

$$\boldsymbol{y} = \begin{bmatrix} \boldsymbol{y}_1^T \\ \vdots \\ \boldsymbol{y}_n^T \end{bmatrix} = \begin{bmatrix} y_{11} & \ldots & y_{1d} \\ \vdots & \vdots & \vdots \\ y_{n1} & \ldots & y_{nd} \end{bmatrix}$$

El vector de medias muestral viene dado por la siguiente expresión en términos de la matriz $\boldsymbol{Y}$,

$$\bar{\boldsymbol{Y}} = \frac{1}{n}\sum_{i=1}^{n} Y_i = \frac{1}{n}\boldsymbol{Y}^T \mathbf{1}_n. \tag{4.9}$$

siendo $\mathbf{1}_n$ el vector $n \times 1$ con todos los valores iguales a uno. También denotaremos

$$\bar{\boldsymbol{Y}} = \begin{bmatrix} \bar{Y}_{\cdot 1} \\ \vdots \\ \bar{Y}_{\cdot p} \end{bmatrix}$$

El estimador de la matriz de covarianzas (poblacional) Σ sería la matriz de covarianzas muestral que tiene en la posición (j,k) la covarianza muestral entre las componentes j y k,

$$S_{jk} = \frac{1}{n-1}\sum_{i=1}^{n}(Y_{ij} - \bar{Y}_{\cdot j})(Y_{ik} - \bar{Y}_{\cdot k}),$$

de modo que

$$S = \begin{bmatrix} S_{11} & \ldots & S_{1d} \\ \vdots & \vdots & \vdots \\ S_{d1} & \ldots & S_{dd} \end{bmatrix} = \frac{1}{n-1}\sum_{i=1}^{n}(Y_i - \bar{Y})(Y_i - \bar{Y})^T = \frac{1}{n-1}Q.$$

Es inmediato que $E\bar{\boldsymbol{Y}} = \boldsymbol{\mu}$ porque componente a componente hemos visto que se verifica la igualdad. A partir de los vectores Y_i consideramos $X_i = Y_i - \mu$ de modo que se verifica $\bar{\boldsymbol{X}} = \bar{\boldsymbol{X}} - \mu$. Se sigue

que

$$\sum_{i=1}^{n}(\boldsymbol{Y}_i-\bar{\boldsymbol{Y}})(\boldsymbol{Y}_i-\bar{\boldsymbol{Y}})^T=\sum_{i=1}^{n}(\boldsymbol{X}_i-\bar{\boldsymbol{X}})(\boldsymbol{X}_i-\bar{\boldsymbol{X}})^T=\sum_{i=1}^{n}\boldsymbol{X}_i\boldsymbol{X}_i^T-n\bar{\boldsymbol{X}}\bar{\boldsymbol{X}}^T.$$

Los vectores $\boldsymbol{X}_1,\ldots,\boldsymbol{X}_n$ tienen vector de medias nulo y matriz de covarianzas $\boldsymbol{\Sigma}$, la misma que los $\boldsymbol{Y}_i$. En consecuencia, $E\bar{\boldsymbol{X}}\bar{\boldsymbol{X}}^T=\boldsymbol{\Sigma}$ y

$$E\boldsymbol{Q}=\sum_{i=1}^{n}cov(\boldsymbol{Y}_i)-n\,cov(\bar{\boldsymbol{Y}})=n\boldsymbol{\Sigma}-n\,cov(\bar{\boldsymbol{Y}})=n\boldsymbol{\Sigma}-n\frac{\boldsymbol{\Sigma}}{n}=(n-1)\boldsymbol{\Sigma}.$$

Tenemos pues que $\boldsymbol{S}$ es un estimador insesgado de la matriz $\boldsymbol{\Sigma}$.

Finalmente, si denotamos por r_{jk} el coeficiente de correlación entre las variables j y k, es decir,

$$r_{jk}=\frac{\sum_{i=1}^{n}(Y_{ij}-\bar{Y}_{\cdot j})(Y_{ik}-\bar{Y}_{\cdot k})}{\sqrt{\sum_{i=1}^{n}(Y_{ij}-\bar{Y}_{\cdot j})^2\sum_{i=1}^{n}(Y_{ik}-\bar{Y}_{\cdot k})^2}}=\frac{S_{jk}}{\sqrt{S_{jj}S_{kk}}}\qquad(4.10)$$

Denotaremos por R la **matriz de correlaciones muestrales** $R=[r_{jk}]$.

4.3 Estimador máximo verosímil

El método de estimación que vamos a utilizar en este curso el **método de máxima verosimilitud**. El estimador máximo verosímil de θ, que denotaremos por $\hat{\theta}$, se obtienen máximizando la función de verosimilitud o, equivalentemente, la transformación monótona de dicha función que es la función de logverosimilitud. Utilizaremos para denotar el estimador máximo verosímil la notación inglesa **MLE**.

$$L(\hat{\theta})=\max_{\theta\in\Theta}L(\theta),\qquad(4.11)$$

o también

$$\hat{\theta}=argmax_{\theta\in\Theta}L(\theta),\qquad(4.12)$$

Ejemplo 4.4 (Bernoulli)**.** *Se puede comprobar sin dificultad que* $\hat{p}=\frac{\sum_{i=1}^{n}x_i}{n}$.

Una propiedad importante de los estimadores máximo verosímiles consiste en que si $\theta^*=f(\theta)$ siendo f una biyección entonces el estimador máximo verosímil de θ^* es verifica que

$$\hat{\theta^*}=f(\hat{\theta}).\qquad(4.13)$$

Ejemplo 4.5 (Normal)**.** *En este caso se comprueba que* $\hat{\mu}=\hat{X}_n$ *y que* $\widehat{\sigma^2}=\frac{n-1}{n}S^2=\frac{1}{n}\sum_{i=1}^{n}(X_i-\bar{X}_n)^2$. *Teniendo en que cuenta la propiedad enunciada en 4.13 tendremos que* $\hat{\sigma}=\sqrt{\frac{n-1}{n}S^2}$.

En muchas situaciones la función $L(\theta)$ es cóncava y el estimador máximo verosímil $\hat{\theta}$ es la solución de las **ecuaciones de verosimilitud** $\frac{\partial L(\theta}{\partial\theta}=0$. Si $cov(\hat{\theta})$ denota la matriz de covarianzas de $\hat{\theta}$ entonces, para un tamaño muestral grande y bajo ciertas condiciones de regularidad (ver [75], página 364), se verifica que $cov(\hat{\theta})$ es la inversa de la **matriz de información** cuyo elemento (j,k) viene dado por

$$-E\left(\frac{\partial^2 l(\theta)}{\partial\theta_j\partial\theta_k}\right)\qquad(4.14)$$

Notemos que el error estándar de $\hat{\theta}_j$ será el elemento que ocupa la posición (j, j) en la inversa de la matriz de información. Cuanto mayor es la curvatura de la logverosimilitud menores serán los errores estándar. La racionalidad que hay detrás de esto es que si la curvatura es mayor entonces la logverosimilitud cae rápidamente cuando el vector θ se aleja de $\hat{\theta}$. En resumen, es de esperar que θ esté más próximo a $\hat{\theta}$.

Ejemplo 4.6 (Binomial)**.** *Supongamos que una muestra en una población finita y consideremos como valor observado el número de éxitos. Entonces la verosimilitud sería*

$$L(p) = \binom{n}{y} p^y (1-p)^{n-y}, \tag{4.15}$$

y la logverosimilitud viene dada como

$$l(p) = \log \binom{n}{y} + y \log p + (n-y) \log(1-p), \tag{4.16}$$

La ecuación de verosimilitud sería

$$\frac{\partial l(p)}{\partial p} = \frac{y}{p} - \frac{n-y}{1-p} = \frac{y - np}{p(1-p)}. \tag{4.17}$$

Igualando a cero tenemos que la solución es $\hat{p} = \frac{y}{n}$ que no es más que la proporción muestral de éxitos en las n pruebas. La varianza asíntótica sería

$$-E\left[\frac{\partial^2 l(p)}{\partial p^2}\right] = E\left[\frac{y}{p^2} + \frac{n-y}{(1-p)^2}\right] = \frac{n}{p(1-p)}. \tag{4.18}$$

En consecuencia asintóticamente $\hat{p}$ tiene varianza $\frac{p(1-p)}{n}$ lo cual era de prever pues si consideramos la variable Y que nos da el número de éxitos entonces sabemos que $EY = np$ y que $var(Y) = np(1-p)$.

4.4 Contraste de hipótesis

Genéricamente vamos a considerar situaciones en donde particionamos el espacio paramétrico Θ en dos conjuntos Θ_0 y Θ_1, es decir, $\Theta_0 \cap \Theta_1 = \emptyset$ (son disjuntos) y y $\Theta_0 \cup \Theta_1 = \Theta$ (cubren todo el espacio paramétrico). Consideramos el contraste de hipótesis siguiente.

$$H_0 : \theta \in \Theta_0 \tag{4.19}$$

$$H_1 : \theta \in \Theta_1 \tag{4.20}$$

Basándonos en una muestra aleatoria $X_1, \ldots, X_n$ hemos de tomar una decisión. Las decisiones a tomar son una entre dos posibles: (i) Rechazar la hipótesis nula o bien (ii) no rechazar la hipótesis nula. Notemos que, una vez hemos tomado una decisión, podemos tener dos posibles tipos de error como recoge la siguiente tabla. En las columnas indicamos la realidad mientras que en las filas indicamos la decisión que tomamos.

Supongamos que $\mathbb{R}^n$ es el conjunto de valores que puede tomar el vector aleatorio $(X_1, \ldots, X_n)$. Entonces el contraste de hipótesis se basa en tomar un estadístico o función de la muestra que denotamos

	H_0	H_1
Rechazamos H_0	Error tipo I	
No rechazamos H_0		Error tipo II

$\delta(X_1, \ldots, X_n)$ de modo que si $\delta(X_1, \ldots, X_n) \in C$ entonces rechazamos la hipótesis nula mientras que si $\delta(X_1, \ldots, X_n) \notin C$ entonces no rechazamos la hipótesis nula. Notemos que simplemente estamos particionando el espacio muestral (que suponemos) $\mathbb{R}^n$ en dos partes, C y C^c, de modo que tomamos una decisión basándonos en si el estadístico δ está en C o bien está en el complementario de C. Al conjunto C se le suele llamar la **región crítica**. La función potencia se define como

$$\pi(\theta) = P(\delta \in C|\theta). \tag{4.21}$$

4.4.1 Test del cociente de verosimilitudes

El cociente de verosimilitudes para contrastar estas hipótesis se define como

$$\Lambda = \frac{\max_{\theta \in \Theta_0} L(\theta)}{\max_{\theta \in \Theta} L(\theta)} \tag{4.22}$$

Es razonable pensar que en la medida en que Λ tome valores menores entonces la hipótesis alternativa sea más plausible que la hipótesis nula y por lo tanto rechacemos la hipótesis nula. Realmente se suele trabajar con $-2 \log \Lambda$ pues bajo la hipótesis nula tiene una distribución asintótica ji-cuadrado donde el número de grados de libertad es la diferencia de las dimensiones de los espacios paramétricos $\Theta = \Theta_0 \cup \Theta_1$ y Θ_0. Si denotamos $L_0 = \max_{\theta \in \Theta_0} L(\theta)$ y $L_1 = \max_{\theta \in \Theta} L(\theta)$ entonces $\Lambda = \frac{L_0}{L_1}$ y

$$-2 \log \lambda = -2 \log \frac{L_0}{L_1} = -2(l_0 - l_1) \tag{4.23}$$

siendo l_0 y l_1 los logaritmos de L_0 y L_1 respectivamente que también corresponden con los máximos de la logverosimilitud sobre Θ_0 y sobre Θ.

4.4.2 Test de Wald

Supongamos que el θ es un parámetro y $\hat{\theta}$ denota su estimador máximo verosímil. Supongamos que queremos contrastar las siguientes hipótesis:

$$H_0 : \theta = \theta_0, \tag{4.24}$$

$$H_1 : \theta \neq \theta_0. \tag{4.25}$$

Denotamos por $SE(\hat{\theta})$ el error estándar bajo la hipótesis alternativa de $\hat{\theta}$. Entonces el estadístico

$$z = \frac{\hat{\theta} - \theta_0}{SE(\hat{\theta})} \tag{4.26}$$

tiene, bajo la hipótesis nula, aproximadamente una distribución normal estándar, $z \sim N(0, 1)$. Este tipo de estadísticos donde se utiliza

el error estándar del estimador bajo la hipótesis alternativa recibe el nombre de *estadístico de Wald*.

Supongamos que θ es un vector de parámetros y queremos contrastar las hipótesis dadas en 4.24. La versión multivariante del estadístico dado en 4.26 viene dada por

$$W = (\hat{\theta} - \theta_0)^T [cov(\hat{\theta})]^{-1} (\hat{\theta} - \theta_0), \tag{4.27}$$

donde $cov(\hat{\theta})$ se estima como la matriz de información observada en el MLE $\hat{\theta}$. La distribución asintótica de W bajo la hipótesis nula es una distribución ji-cuadrado donde el número de grados de libertad coincide con el número de parámetros no redundantes en θ.

4.4.3 Intervalos de confianza

Empezamos recordando el concepto de intervalo de confianza con un ejemplo muy conocido como es la estimación de la media en poblaciones normales.

Ejemplo 4.7 (Intervalo de confianza para la media de una normal)**.** *Veámoslo con un ejemplo y luego planteamos la situación más general. Tenemos una muestra aleatoria $X_1, \ldots, X_n$ i.i.d. tales que $X_i \sim N(\mu, \sigma 2)$. Entonces es conocido que*

$$\frac{\bar{X}_n - \mu}{S/\sqrt{n}} \sim t_{n-1}. \tag{4.28}$$

Vemos cómo $\frac{\bar{X}_n - \mu}{S/\sqrt{n}}$ depende tanto de la muestra que conocemos como de un parámetro (la media μ) que desconocemos. Fijamos un valor de α (habitualmente tomaremos $\alpha = 0.05$) y elegimos un valor $t_{n-1,1-\alpha/2}$ tal que

$$P(-t_{n-1,1-\alpha/2} \leq \frac{\bar{X}_n - \mu}{S/\sqrt{n}} \leq t_{n-1,1-\alpha/2}) = 1 - \alpha. \tag{4.29}$$

La ecuación anterior la podemos reescribir como

$$P(\bar{X}_n - t_{n-1,1-\alpha/2}\frac{S}{\sqrt{n}} \leq \mu \leq \bar{X}_n + t_{n-1,1-\alpha/2}\frac{S}{\sqrt{n}}) = 1 - \alpha. \tag{4.30}$$

Tenemos una muestra aleatoria $X_1, \ldots, X_n$ y por lo tanto tenemos un intervalo aleatorio dado por $[\bar{X}_n - t_{n-1,1-\alpha/2}\frac{S}{\sqrt{n}}, \bar{X}_n + t_{n-1,1-\alpha/2}\frac{S}{\sqrt{n}}]$. Este intervalo tiene una probabilidad de $1-\alpha$ de contener a la verdadera media. Tomemos ahora la muestra y consideremos no los valores aleatorios de $\bar{X}_n$ y de S^2 sino los valores observados $\bar{x}_n$ y s. Tenemos ahora un intervalo $[\bar{x}_n - t_{n-1,1-\alpha/2}\frac{s}{\sqrt{n}}, \bar{x}_n + t_{n-1,1-\alpha/2}\frac{s}{\sqrt{n}}]$ fijo. Es posible que μ esté en este intervalo y es posible que no lo esté. Sabemos que antes de tomar la muestra teníamos una probabilidad *de $1-\alpha$ de contener a la verdadera media pero después de tomar la muestra tenemos una* ***confianza*** *de $1-\alpha$ de contener a la verdadera media. Al intervalo $[\bar{x}_n - t_{n-1,1-\alpha/2}\frac{s}{\sqrt{n}}, \bar{x}_n + t_{n-1,1-\alpha/2}\frac{s}{\sqrt{n}}]$ se le llama* ***intervalo de confianza*** *para μ con* ***nivel de confianza*** *$1-\alpha$.*

Vamos a ver un planteamiento más general del problema. Supongamos que tenemos un test para contrastar la hipótesis simple $H_0 : \theta = \theta_0$ frente a la alternativa $H_1 : \theta \neq \theta_0$. Supongamos que

elegimos un nivel de significación α para contrastar las hipótesis anteriores y consideramos el siguiente conjunto formado por todos los θ_0 tales que no rechazamos la hipótesis nula al nivel α. Este conjunto es un **conjunto de confianza** al nivel $1-\alpha$. Cuando el conjunto de confianza es un intervalo hablamos de **intervalo de confianza**.

Supongamos que consideramos el test del cociente de verosimilitudes. Denotemos por $\chi^2_k(1-\alpha)$ el percentil $1-\alpha$ de una distribución ji-cuadrado con k grados de libertad. Entonces el intervalo de confianza al nivel $1-\alpha$ sería el conjunto

$$\{\theta_0 : -2[l(\theta_0) - l(\hat{\theta})] < \chi^2_k(1-\alpha)\} \tag{4.31}$$

Consideremos ahora un test de Wald. En este caso, el intervalo de confianza de Wald vendría dado por el siguiente conjunto:

$$\{\theta_0 : \frac{|\hat{\theta} - \theta_0|}{SE(\hat{\theta})} < Z_{1-\alpha/2}\} \tag{4.32}$$

donde $SE(\hat{\theta})$ es el error estándar estimado de $\hat{\theta}$ bajo la hipótesis alternativa.

Capítulo 5

Modelos lineales

En este tema pretendemos repasar conceptos básicos de regresión lineal múltiple. Es lo básico que necesitamos para entender los datos en que la respuesta es numérica y tenemos variables predictoras (variables fenotípicas en este contexto) que pretendemos ver cómo influyen en la expresión del gen. Lo que veamos en este capítulo es de aplicación cuando trabajamos microarrays de DNA, metilación, microarrays de proteínas.

Utilizamos para ilustrar los datos `tamidata2::gse25171`.

```
pacman::p_load(Biobase)
data(gse25171,package="tamidata2")
```

Veamos las variables fenotípicas de las que disponemos.

```
head(pData(gse25171),n=2)
```

```
              time time2 Pi replication
GSM618324.CEL.gz 0 Short Treatment 1
GSM618325.CEL.gz 0 Short Control 2
```

Vamos a plantearnos cómo pueden influir el tiempo de observación de la muestra (**`time`**) y la presencia o no de fósforo en la expresión del gen (`Pi`).

Elegimos la sonda `261892_at`. Construimos un **`data.frame`** con la información necesaria en donde recogemos los dos predictores (**`time`** y `Pi`) y la variable respuesta que sería la fila de la matriz de expresión correspondiente a la sonda `261892_at`.

```
sel0 = which("261892_at"==fData(gse25171)[,"PROBEID"])
df0 = data.frame(pData(gse25171)[,c("time","Pi")],
                 expression=exprs(gse25171)[sel0,])
```

5.1 Sobre lo que vamos a tratar

5.1.1 Problemas y datos

¿Qué problemas queremos resolver? ¿Qué información tenemos para resolverlos? Empezamos por la información. Tenemos información (numérica, categórica) sobre una serie de muestras u observaciones. Pueden tomar valores fijados por nosotros o bien valores observados (realizaciones de variables aleatorias). No todas las variables de las que disponemos tienen el mismo interés. Habitualmente hay una

variable **importante**: expresión de un gen, número de lecturas alineadas. A esta variable de interés la denotamos como y_i para la i-ésima observación y la llamaremos **variable respuesta** (en denominaciones más clásicas, variable dependiente). El resto de variables del estudio son las variables predictoras o variables independientes o (menos frecuente) **regresores** o **inputs**. Habitualmente en nuestro caso serán las variables fenotípicas que nos describen las muestras. Para la i-ésima observación tenemos p variables predictoras que recogemos en el vector columna $\boldsymbol{x}_i \in \mathbb{R}^p$ donde

$$\boldsymbol{x}_i = \begin{bmatrix} x_{i1} \\ \vdots \\ x_{ip} \end{bmatrix}.$$

Denotamos el vector traspuesto como $\boldsymbol{x}_i^T = (x_{i1}, \ldots, x_{ip})$. Nuestra información consiste en los pares $(\boldsymbol{x}_i, y_i)$ con $i = 1, \ldots, n$.

¿Qué problema queremos resolver? Una respuesta fácil es decir que queremos conocer el valor de la variable respuesta utilizando las variables predictoras. ¿Solamente queremos conocer el valor de la respuesta? Quizás estamos pensando en un futuro en donde conozcamos las variables predictoras y nos interese saber cuál será el valor de la variable respuesta correspondiente. Sin embargo, conocer el valor de la respuesta tiene distintas interpretaciones posibles: ¿El valor exacto de la respuesta? ¿La media de la respuesta? ¿Un valor numérico que aproxime cada una de estas cantidades o bien un intervalo que las contenga?

Los valores observados y_i consideraremos que son realizaciones de variables aleatorias Y_i.[34] Realmente en lo que sigue modelizaremos el comportamiento aleatorio de la variable Y_i condicionada a los valores observados $\boldsymbol{x}_i$. Es decir, nuestro interés estará en *l*a distribución condicionada: tenemos el vector aleatorio $\boldsymbol{X}_i$ (donde $\boldsymbol{x}_i$ son los valores observados) y la variable aletoria Y_i observadas conjuntamente. Entonces el vector $(\boldsymbol{X}_i^T, Y_i)^T$ tendrá densidad conjunta[35] $f(y_i, \boldsymbol{x}_i)$ y podemos considerar la densidad condicionada $f(y_i|\boldsymbol{x}_i)$ (o la probabilidad condicionada P_i). La media condicionada que nos interesa es, cuando tenemos una distribución (absolutamente) continua respecto de la medida de Lebesgue, la siguiente

$$\mu_i = E[Y_i|\boldsymbol{x}_i] = \int_{-\infty}^{+\infty} y_i f(y_i|\boldsymbol{x}_i) dy_i. \tag{5.1}$$

Si consideramos una distribución discreta será

$$\mu_i = E[Y_i|\boldsymbol{x}_i] = \sum_{y_i} y_i f(y_i|\boldsymbol{x}_i). \tag{5.2}$$

Y si consideramos una variable que no es ni discreta ni continua y denotamos por P_i la probabilidad condicional entonces

$$\mu_i = E[Y_i|\boldsymbol{x}_i] = \int_{-\infty}^{+\infty} y_i P_i(dy_i). \tag{5.3}$$

En lo que sigue vamos a asumir que las correspondientes distribuciones condicionadas son independientes entre si: las Y_i serán condicionalmente independientes.

[34] Denotamos con mayúsculas las variables aleatorias y con minúsculas los valores observados.

[35] Entendemos densidad en su sentido más genérico incluyendo distribuciones continuas, discretas y distribuciones que no son ni continuas ni discretas.

Nuestro interés fundamental (pero no único) estará en conocer las medias condicionadas $\mu_i = E[Y_i|\boldsymbol{x}_i]$. La variable respuesta podrá ser cuantitativa (continua o discreta) o cualitativa (posiblemente ordinal). Las variables predictoras pueden ser numéricas o categóricas.

5.1.2 Modelos sobre la media

En la sección anterior una de las opciones que nos planteamos de conocer la respuesta es conocer la media de la respuesta aleatoria. Tenemos unas variables aleatorias. Queremos conocer esa respuesta media cuando están dadas estas variables predictoras. En definitiva pretendemos conocer la media de la respuesta aleatoria **condicionada** a los valores de las variables predictoras. Si denotamos por Y la respuesta aleatoria y por $\boldsymbol{x}$ las variables predictoras[1] nuestro interés es conocer $E[Y|\boldsymbol{x}]$. Si consideramos la i-ésima respuesta tendremos $E[Y_i|\boldsymbol{x}_i]$. Para simplificar esta notación denotaremos simplemente $\mu_i = E[Y_i|\boldsymbol{x}_i]$ sin indicar explícitamente las variables predictoras.

5.1.3 Dependencia lineal

¿Qué tipos de dependencia vamos a considerar? En la mayor parte de los casos dependencias de tipo lineal. En lo que sigue vemos cómo expresar dependencias de la media condicionada μ_i respecto de los predictoras cuando estos son números o categóricos o numéricos y categóricos. Suponemos dos predictores $\boldsymbol{x} = (x_1, x_2)^T$ tales que x_1 es numérico y el segundo es una variable categórica binaria codificada con 1 y 0.[36] ¿Cómo modelizamos la dependencia de la media condicionada respecto de x_1? Una dependencia lineal vendría dada como

$$\mu_i = \beta_0 + \beta_1 x_{i1}. \tag{5.4}$$

¿Y la dependencia de las μ_i respecto de la variable binaria? Obviamente simplemente tenemos dos valores. Un modo simple es

$$\mu_i = \beta_0 + \beta_2 x_{i2}. \tag{5.5}$$

Si lo hacemos así tenemos que cuando $x_{i2} = 0$ entonces $\mu_i = \beta_0$ mientras que cuando $x_{i2} = 1$ tendremos $\mu_i = \beta_0 + \beta_2$.

¿Y las dos variables predictoras conjuntamente consideradas? Quizás el modelo más sencillo sería:

$$\mu_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2}. \tag{5.6}$$

Si consideramos este modelo tendremos que estamos realmente especificando dos modelos para la media. Cuando $x_{i2} = 0$ entonces $\mu_i = \beta_0 + \beta_1 x_{i1}$. Cuando $x_{i2} = 1$ entonces $\mu_i = \beta_0 + \beta_2 + \beta_1 x_{i1}$. Realmente estamos modificando la ordenada en el origen de la línea recta. En la figura 5.2 mostramos ambas líneas. Tenemos pues dos líneas paralelas.

¿Como podemos expresar la dependencia de ambas covariables? Otra vez recurrimos a la opción de expresarlo de un modo lineal.

$$\mu_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i1} x_{i2} \tag{5.7}$$

[1] Observemos que denotamos y denotaremos el valor aleatorio en mayúscula mientras que las variables predictoras las consideramos dadas, observadas, no aleatorias y por lo tanto las denotamos en minúscula.

[36] Indicando presencia o ausencia de un atributo.

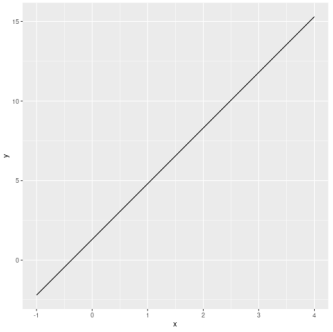


Figura 5.1: Modelo para la media según modelo en ecuación 5.4.

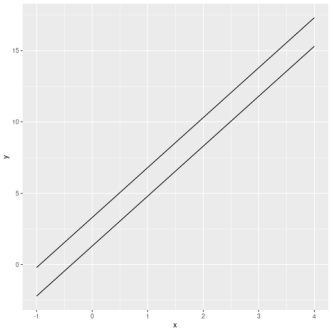


Figura 5.2: Modelo para la media en ecuación 5.6.

Tendremos dos líneas que expresan la dependencia. Cuando $x_{i2} = 0$ tenemos

$$\mu_i = \beta_0 + \beta_1 x_{i1}. \tag{5.8}$$

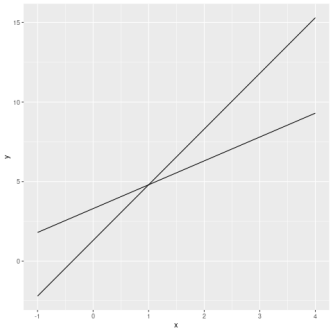


Figura 5.3: Modelo para la media propuesto en ecuación 5.9.

Cuando $x_{i2} = 1$ tenemos

$$\mu_i = (\beta_0 + \beta_2) + (\beta_1 + \beta_3)x_{i1}. \tag{5.9}$$

Vemos que se modifica tanto la ordenada en origen como la pendiente de la recta. Hay, lo que luego llamaremos, una interacción entre las dos variables predictoras.

Cuando consideramos una variable predictora categórica con más de dos categorías entonces es habitual codificarla utilizando variables tontas.[37] Si la variable predictora categórica tiene I categorías entonces se elige una categoría de referencia, por ejemplo la primera,[38] entonces las variables binarias asociadas a la variable original x serían: $v_1 = 1$ si $x = 2$ y cero en otro caso; $v_2 = 1$ si $x = 3$ y cero en otro caso; ...; $v_{I-1} = 1$ si $x = I$ y cero en otro caso. Obviamente cuando todas las variables v son nulas estamos en la primera categoría. Si solamente tenemos la variable categórica como predictora entonces la media sería función lineal de x del siguiente modo:

$$\mu_i = \beta_0 + \beta_1 v_1 + \ldots + \beta_{I-1} v_{I-1}.$$

[37] Dummy variables.

[38] Cada software por defecto elige una y podemos modificarla. En R la categoría de referencia es la primera.

Si como variables predictoras tenemos distintas variables algunas numéricas y otras categóricas (con dos o más de dos categorías) tendremos el mismo modo de modelizar que acabamos de ver simplemente añadiendo más términos.

Por ejemplo, supongamos una numérica x y una categórica v con I categorías. Construimos las variables tontas siendo I la de referencia. Podemos considerar modelos como

$$\mu_i = \beta_0 + \beta_1 x_i, \tag{5.10}$$

que lo expresamos como $y \sim x$. Un modelo que contiene solamente a v sería el dado previamente

$$\mu_i = \beta_0 + \beta_1 v_{i1} + \ldots + \beta_{I-1} v_{i,I-1}. \tag{5.11}$$

y lo expresamos como $y \sim v$. Un modelo que contempla ambas variables puede ser

$$\mu_i = \beta_0 + \beta_1 x_i + \beta_2 v_{i1} + \ldots + \beta_I v_{i,I-1} \tag{5.12}$$

Este modelo lo podemos abreviar (y así se le indicará a R) como $y \sim x + v$. Un modelo más completo que contempla la posible interacción sería

$$\mu_i = \beta_0 + \beta_1 x_i + \beta_2 v_{i1} + \ldots + \beta_I v_{i,I-1} + \beta_{I+1} x_i v_{i1} + \ldots + \beta_{2I-1} x_i v_{i,(I-1)} \tag{5.13}$$

Esto lo indicaremos como $y \sim x * v$ o bien como $y \sim x + v + x : v$.

Supongamos dos variables categóricas u y v con I y J categorías respectivamente. Tendríamos el modelo $y \sim u$ y el modelo $y \sim v$ en donde solo se considera la influencia o efecto de cada una de ellas

Tabla 5.1: Especificaciones simplificadas de los modelos.

Modelo	Notación en R
Regresión lineal simple	$y \sim x$
Regresión lineal múltiple	$y \sim x_1 + \ldots + x_p$
Anova de una vía	$y \sim u$
Anova de dos vías sin interacción	$y \sim u + v$
Anova de dos vías con interacción	$y \sim u + v + u : v$
	$y \sim u * v$

aislada o marginalmente. Un modelo más completo sería

$$\begin{aligned}\mu_i = \beta_0 + & \\ & \beta_1 u_{i1} + \ldots + \beta_I u_{i,I-1} + \\ & \beta_{I+1} v_{i1} + \ldots + \beta_{2I-1} v_{i,(I-1)} + \\ & \beta_{I+1} u_{i1} v_{i1} + \ldots + \beta_{3I-2} u_{i1} v_{i,(I-1)} + \ldots + \\ & \beta_{I+1} u_{i(I-1)} v_{i1} + \ldots + \beta_{(I-1)^2+2(I-1)+1} u_{i(I-1)} v_{i,(I-1)}\end{aligned} \quad (5.14)$$

Esto se indicaría como $y \sim u * v$.

En la tabla 5.1 se indica cómo se suele denominar al modelo resultante. Se utiliza en esta tabla la notación propuesta en [93] y que utilizaremos extensamente en el curso. Es lo que se conoce como **`formula`** en R.

5.1.4 Efectos

Cuando se habla de efectos en modelos lineales nos referimos a los parámetros. El *efecto* que producen en la variable respuesta es a traves de este coeficiente.

¿Cómo interpretamos los coeficientes en un modelo lineal? El caso más simple sería un modelo con una sola covariable o variable predictora: $\mu_i = \beta_0 + \beta_1 x_{i1}$. Parece natural y simple decir algo como: un incremento unitario en la variable x se traduce en un incremento unitario de la media. Formalmente es correcta la afirmación. Sin embargo, en términos de interpretación la cosa no es tan correcta. Si pudiéramos realizar un experimento y sobre la misma unidad experimental modificar el valor de x y ver su efecto entonces sí que sería correcto. Esto no es lo habitual. Por ello, un interpretación mejor podría ser la siguiente: Consideramos dos subpoblaciones formadas por los individuos donde el valor de la covariable es x y la formada por los individuos donde la covariable toma el valor $x + 1$. La diferencia de medias entre ambas subpoblaciones es el coeficiente β_1. Una cosa es el formalismo matemático y otra cosa es la interpretación estadística.[39]

[39] Este manual está lleno de comentarios de este tipo.

Supongamos que tenemos más de una covariable. Nuestro modelo ahora es $\mu_i = \beta_0 + \beta_1 x_{i1} + \ldots + \beta_{p-1} x_{i,p-1}$. ¿Cómo interpretamos el valor de β_1, el efecto de la variable x_{i1} sobre la variable respuesta? La afirmación siguiente es matemáticamente correcta: si mantenemos todas las demás covariables constantes e incrementamos en una unidad la covariable x_{i1} entonces la media cambia en β_1 unidades. Correcto matemáticamente lo es. Pero, ¿es posible? En ocasiones no podemos mantener constantes todas las covariables y modificar el valor de x_{i1}. ¿Por qué? Porque puede haber dependencias entre ellas. Cambiar el valor de x_{i1} supone cambiar el valor de las otras si trabajamos con

datos observacionales, no controlados. Incluso pueden darse combinaciones de las covariables imposibles. Pensemos que una covariable puede ser el sexo de la persona y valores de otras covariables solamente se pueden dar para un sexo y no para otro. Otra vez: ¿cómo interpretamos β_1? Consideramos otra vez dos subpoblaciones con valores x_{i1} y $x_{i1}+1$ para la primera covariable. Y suponemos que, en ambas subpoblaciones, el valor de $\beta_2 x_{i2} + \ldots + \beta_1 x_{ip}$ es el mismo. La diferencia de medias de la respuesta en ambas subpoblaciones viene dada por β_1. Dicho de una manera más técnica, β_1 es la diferencia de medias cuando modificamos en una unidad x_{i1} **ajustando** por el resto de las covariables.

5.1.5 Matriz modelo y espacio modelo

Tenemos el vector de medias a estimar

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \vdots \\ \mu_n \end{bmatrix}.$$

Podemos considerar la matriz que, en cada fila, tenga los predictores correspondientes a la i-ésima observación

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & \ldots & x_{1p} \\ \vdots & \ddots & \vdots \\ x_{n1} & \ldots & x_{np} \end{bmatrix}.$$

[40] A veces, en diseño de experimentos se habla de **matriz de diseño**.

Esta matriz recibe el nombre de **matriz modelo**.[40] De hecho, los ejemplos que hemos comentado previamente todos verifican que

$$\mu_i = \sum_{j=1}^{p} \beta_j x_{ij},$$

por tanto, podemos expresar esta dependencia como

$$\boldsymbol{\mu} = \boldsymbol{X}\boldsymbol{\beta},$$

siendo el vector de coeficientes

$$\boldsymbol{\beta} = \begin{bmatrix} \beta_1 \\ \vdots \\ \beta_p \end{bmatrix}.$$

Pretendemos estimar $\boldsymbol{\mu} = E\boldsymbol{Y}$. Tenemos unas covariables o predictores. ¿En dónde estamos jugando? ¿Qué conjunto de posibles valores para $\boldsymbol{\mu}$ tenemos si asumimos un modelo lineal? Puesto que asumimos $\boldsymbol{\mu} = \boldsymbol{X}\boldsymbol{\beta}$ entonces el conjunto vendría dado por

$$C(\boldsymbol{X}) = \{\boldsymbol{\eta} : \boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta}, \boldsymbol{\beta} \in \mathbb{R}^p\},$$

o lo que es lo mismo, el espacio vectorial generado por las columnas de la matriz de modelo $\boldsymbol{X}$. A este espacio vectorial lo podemos llamar **espacio modelo**. Es claro que si tenemos dos matrices modelo $\boldsymbol{X}_1$ y $\boldsymbol{X}_2$ tales que $C(\boldsymbol{X}_1) = C(\boldsymbol{X}_2)$ entonces los posibles valores de $\boldsymbol{\mu}$ son los mismos y tenemos pues el mismo modelo. Podemos tener también

la situación en que $C(\boldsymbol{X}_1) \subset C(\boldsymbol{X}_2)$, por ejemplo, porque hemos eliminado alguna columna. En este caso la matriz de modelo $\boldsymbol{X}_1$ representa un modelo simplificado del modelo más general formulado con la matriz modelo $\boldsymbol{X}_2$. La dimensión de $C(\boldsymbol{X}) = \{\boldsymbol{\eta} : \boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta}, \boldsymbol{\beta} \in \mathbb{R}^p\}$ es el número de columnas (que coincide con el de filas A.8) linealmente independientes: $dim(C(\boldsymbol{X})) = rank(\boldsymbol{X})$. Siempre asumiremos (salvo que se indique explícitamente) que tenemos más observaciones que variables o lo que es lo mismo que n, número de observaciones, supera a p número de variables: $n \geq p$. Por tanto la matriz tendrá rango completo cuando $rank(\boldsymbol{X}) = p$.

El espacio nulo de $\boldsymbol{X}$ es

$$null(\boldsymbol{X}) = \{\boldsymbol{\psi} : \boldsymbol{X}\boldsymbol{\psi} = \boldsymbol{0}\}.$$

Se tiene que (A.11)

$$dim(C(\boldsymbol{X})) + dim(null(\boldsymbol{X})) = p.$$

Es claro que si la matriz de modelo no es de rango completo entonces no tenemos bien definidos los parámetros del modelo.

5.2 Regresión lineal simple

En un primer momento consideramos solamente la variable predictora (fenotípica en nuestro contexto) `time`. En figura 5.4 tenemos el tiempo en abscisas y la expresión observada en la sonda `261892_at` en ordenadas.

```
pacman::p_load(ggplot2)
p = ggplot(df0,aes(x=time,y=expression))+geom_point()
ggsave(paste0(dirTamiFigures,"gse25171_261892_at.png"),p)
```

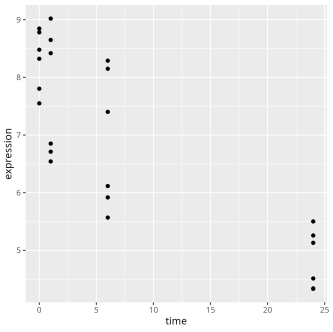


Figura 5.4: Una sonda de tamidata2::gse25171.

5.2.1 Recta de mínimos cuadrados

Nuestros datos son (x_i, y_i) con $i = 1, \ldots, n$ y pretendemos estudiar la posible dependencia de los valores y respecto de los valores x. Una posible dependencia, sin duda, la más simple es considerar que la respuesta es función lineal de la predictora,

$$y_i = \beta_0 + \beta_1 x_i,$$

con $i = 1, \ldots, n$. Obviamente no es posible. No hay solución para las ecuaciones anteriores. Podemos sustituir la idea de resolver las ecuaciones con la de encontrar una **buena** solución aproximada,

$$y_i \approx \beta_0 + \beta_1 x_i,$$

para $i = 1, \ldots, n$. Una posibilidad para determinar unos **buenos** valores para β_0 y β_1 es considerar la siguiente suma de cuadrados S_t

$$S_t = \sum_{i=1}^{n} (y_i - \beta_0 - \beta_1 x_i)^2, \tag{5.15}$$

y considerar los valores de β_0 y β_1 que la minimizan. Haciendo algún cálculo se obtiene

$$\hat{\beta}_1 = \frac{\sum_{i=1}^n x_i y_i - (\sum_{i=1}^n x_i \sum_{i=1}^n y_i)/n}{\sum_{i=1}^n x_i^2 - (\sum_{i=1}^n x_i)^2/n}. \tag{5.16}$$

$$\hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}. \tag{5.17}$$

Ya tenemos una buena aproximación que nos permite relacionar la variable predictora con la variable respuesta. Sería la **recta de mínimos cuadrados**:

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x.$$

Consecuencia inmediata de la ecuación (5.17), la recta de mínimos cuadrados se puede escribir como

$$\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x = \bar{y} + \hat{\beta}_1 (x - \bar{x}). \tag{5.18}$$

Y, en particular, a partir de (5.18) se sigue que pasa por el punto $(\bar{x}, \bar{y})$. Si denotamos, como es usual,

$$S_{xy} = \sum_{i=1}^{n} (x_i - \bar{x})(y_i - \bar{y}), \tag{5.19}$$

y

$$S_{xx} = \sum_{i=1}^{n} (x_i - \bar{x})^2, \quad S_{yy} = \sum_{i=1}^{n} (y_i - \bar{y})^2, \tag{5.20}$$

entonces

$$\hat{\beta}_1 = \frac{S_{xy}}{S_{xx}}. \tag{5.21}$$

Ejemplo 5.1. *¿Cómo obtenemos con R estos estimadores de la ordenada en el origen y de la pendiente de la curva?*

```
(fit0 = lm(expression ~time, data = df0))
```

```
Call:
lm(formula = expression ~time, data = df0)

Coefficients:
(Intercept) time
    7.9684 -0.1331
```

Los coeficientes de la recta de mínimos cuadrados son

```
coef(fit0)
```

```
(Intercept) time
  7.9684362 -0.1330703
```

Podemos representar la recta de mínimos cuadrados con el siguiente código y aparece en la figura 5.5.

```
p = ggplot(df0,aes(x=time,y=expression))+
    geom_point() + geom_smooth(method='lm', se = FALSE)
```

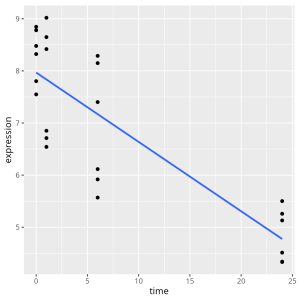


Figura 5.5: Datos de figura 5.5 superponiendo la recta de mínimos cuadrados.

Nos planteábamos en un principio encontrar unas buenas aproximaciones para y_i utilizando x_i. Lo natural sería tomar el siguiente valor la **predicción** o **valor ajustado** para y_i:

$$\hat{y}_i = \hat{\beta}_0 + \hat{\beta}_1 x_i.$$

Las predicciones, $\hat{y}_i$ las tenemos con (`utils::head()` muestra las primeras),

```
head(predict(fit0))
```

```
GSM618324.CEL.gz GSM618325.CEL.gz
       7.968436 7.968436
GSM618326.CEL.gz GSM618327.CEL.gz
       7.835366 7.835366
GSM618328.CEL.gz GSM618329.CEL.gz
       7.170014 7.170014
```

La diferencia entre valor observado, y_i, y su predicción, $\hat{y}_i$, se conoce como **residuo**. Esencialmente una estimación del error,

$$e_i = y_i - \hat{y}_i.$$

Se prueba que cuando el modelo tiene una constante entonces la suma de todos los residuos es nula,

$$\sum_{i=1}^{n} e_i = 0.$$

Los residuos los podemos calcular con

```
head(resid(fit0))
```

```
GSM618324.CEL.gz GSM618325.CEL.gz
      0.3545741 0.8777147
GSM618326.CEL.gz GSM618327.CEL.gz
     -0.9836174 0.5831095
GSM618328.CEL.gz GSM618329.CEL.gz
     -1.2520196 0.9791937
```

5.2.2 Sumas de cuadrados

¿Cuánto de la variación que hay en la variable respuesta ha sido explicada por la regresión? Para responder la pregunta quizás lo mejor sería preguntarnos ¿qué entendemos por la variación de la variable respuesta? Se puede interpretar de dos formas: la primera sería simplemente como

$$\sum_{i=1}^{n} y_i^2,$$

o bien, como variación respecto de la media que vendría dada por

$$\sum_{i=1}^{n} (y_i - \bar{y})^2.$$

Consideremos las siguientes sumas de cuadrados.

Suma de cuadrados total

$$SS(total) = \sum_{i=1}^{n} (y_i - \bar{y})^2,$$

Suma de cuadrados debida a la regresión

$$SS(regression) = \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2,$$

Suma de cuadrados residual

$$SS(residual) = \sum_{i=1}^{n}(y_i - \hat{y}_i)^2.$$

Se tiene la siguiente igualdad

$$SS(Total) = SS(Regression) + SS(Residual). \tag{5.22}$$

Las tres sumas de cuadrados de la ecuación (5.22) suelen disponerse en forma de tabla que recibe el nombre de tabla de análisis de la varianza. La mostramos en la tabla 5.2. Más adelante veremos esta misma tabla y la interpretaremos con detalle. Ahora quizás darse cuenta que estamos considerando un cociente que cuantifica hasta qué punto la suma de cuadrados de la regresión es grande en relación a la suma de cuadrados residual. Este cociente, considerando los grados de libertad, tiene una distribución de probabilidad que es una F de Fisher. En lo que sigue probamos que esto es efectivamente así. Ahora, cuanto mayor es el valor del estadístico (`F value`) o equivalentemente cuanto menor es el área a la derecha del estadístico (`Pr(>F)`) mejor es el ajuste que hemos obtenido.

Tabla 5.2: Tabla de análisis de la varianza asociada al ajuste de regresión.

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
Regression	1	$SS(Regression)$	$SS(Regression)/1$	$\frac{SS(Regression)/1}{SS(Residual)/(n-2)}$	
Residuals	$n-2$	$SS(Residual)$	$SS(Residual)/(n-2)$		

Ejemplo 5.2. *Veamos la tabla de análisis de la varianza.*

```
fit1 = aov(expression ~time, data = df0)
summary(fit1)
```

```
            Df Sum Sq Mean Sq F value Pr(>F)
time 1 39.60 39.60 50.73 3.84e-07
Residuals 22 17.18 0.78

time ***
Residuals
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Tabla 5.3: Tabla 5.2 observada para la sonda.

	Df	Sum Sq	Mean Sq	F value	Pr(>F)
time	1	39.60	39.60	50.73	0.0000
Residuals	22	17.18	0.78		

5.2.3 Coeficiente de determinación R^2

Es una medida de la calidad del ajuste. Se define como

$$R^2 = \frac{\sum_{i=1}^{n}(\hat{y}_i - \bar{y})^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = \frac{SS(Regression)}{SS(Total)}. \tag{5.23}$$

Si tenemos en cuenta la ecuación (5.22) entonces

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} = 1 - \frac{SS(Residual)}{SS(Total)}. \tag{5.24}$$

A partir de (5.24) se ve fácilmente que un mejor ajuste de la recta de mínimos cuadrados, esto es, cuanto más cerca estén los puntos de la recta mayor será el valor del coeficiente de determinación R^2.

```
fit0.s = summary(fit0)
fit0.s$r.squared
```

```
[1] 0.6974934
```

5.2.4 Modelo

En lo que hemos hecho hasta ahora nos hemos limitado a obtener una buena aproximación lineal de las respuesta a partir del predictor. No se puede ir mucho más allá sin un modelo probabilístico que nos permita valorar lo que estamos haciendo.

Definición 5.1 (Regresión lineal simple)**.**

1. $Y_i = \beta_0 + \beta_1 x_i + \epsilon_i$ *para* $i = 1, \ldots, n$.
2. $\epsilon_i \sim N(0, \sigma^2)$.
3. *Los errores* ϵ_i *son independientes.*

En definitiva estamos asumiendo que la distribución condicionada de Y_i al predictor x_i es normal con media $\beta_0 + \beta_1 x_i$ y varianza σ^2,

$$Y_i | x_i \sim N(\beta_0 + \beta_1 x_i, \sigma^2),$$

y que los distintos Y_i son independientes entre sí.

Es conveniente y útil considerar una representación matricial del modelo de regresión lineal simple. Tenemos el **vector de respuestas aleatorias**, $\boldsymbol{Y}$; la **matriz de modelo**, $\boldsymbol{X}$; el **vector de coeficientes**, $\boldsymbol{\beta}$ y el **vector de errores aleatorios**, $\boldsymbol{\epsilon}$ dados por

$$\boldsymbol{Y} = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix}; \quad \boldsymbol{X} = \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix}; \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix}; \quad \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

Y el modelo de regresión lineal simple se puede formular de un modo más compacto como

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}. \tag{5.25}$$

Ejemplo 5.3. *En el ajuste propuesto en el ejemplo podemos obtener la matriz de modelo con*

```
head(model.matrix(fit0))
```

```
                (Intercept) time
GSM618324.CEL.gz           1    0
GSM618325.CEL.gz           1    0
GSM618326.CEL.gz           1    1
GSM618327.CEL.gz           1    1
GSM618328.CEL.gz           1    6
GSM618329.CEL.gz           1    6
```

5.2.5 Verosimilitud

¿Cuál es la función de verosimitud? Asumiendo el modelo formulado en § 5.2.4 la función de verosimilitud viene dada por

$$L(\beta_0, \beta_1, \sigma^2; y_1, \ldots, y_n) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2\sigma^2}(y_i - \beta_0 - \beta_1 x_i)^2} = \frac{1}{(2\pi)^{n/2}\sigma^n} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2} \quad (5.26)$$

Como sabemos es más habitual y práctico trabajar con el logaritmo natural de la verosimilitud o **logverosimilitud**. En este caso la función de logverosimilitud viene dada por

$$\ell(\beta_0, \beta_1, \sigma^2; y_1, \ldots, y_n) = -\frac{n}{2}\ln(2\pi) - n\ln(\sigma) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \beta_0 - \beta_1 x_i)^2. \quad (5.27)$$

Por ser una transformación monótona los valores máximos para los parámetros son los mismos en la función de verosimilitud y en la de logverosimilitud. Estos estimadores son los estimadores máximo verosímiles. Si observamos la expresión de la función de logverosimilitud en (5.27) es equivalente maximizar esta función a minimizar la suma de cuadrados. De otro modo los estimadores máximo verosímiles de los coeficientes corresponden con los estimadores mínimo cuadráticos obtenidos previamente.

5.2.6 Contrastes e intervalos para la pendiente

[41] Cosa que habrá que ver.

Si el modelo propuesto en (5.1) es razonablemente asumible[41] entonces podemos plantearnos determinar intervalos de confianza y contrastar hipótesis para los coeficientes que expresan la dependencia de la variable respuesta respecto de la variable predictora. Tenemos

$$\hat{\beta}_1 = \frac{S_{xy}}{S_{xx}} = \sum_{i=1}^{n} \frac{x_i - \bar{x}}{S_{xx}} Y_i. \quad (5.28)$$

Se verifica que $E[\hat{\beta}_1] = \beta_1$. La varianza de $\hat{\beta}_1$ es σ^2/S_{xx}. En consecuencia la desviación estándar de $\hat{\beta}_1$ es

$$\sigma_{\hat{\beta}_1} = \frac{\sigma}{S_{xx}^{1/2}}.$$

Supongamos (más adelante lo justificaremos) que estimamos σ^2 con

$$\widehat{\sigma^2} = \frac{1}{n-2}\sum_{i=1}^{n} e_i^2.$$

La desviación estándar la estimaríamos con

$$\hat{\sigma} = \sqrt{\frac{1}{n-2}\sum_{i=1}^{n} e_i^2}.$$

La desviación estándar estimada o **error estándar** de $\hat{\beta}_1$ es

$$\hat{\sigma}_{\hat{\beta}_1} = \frac{\hat{\sigma}}{S_{xx}^{1/2}}.$$

De hecho tenemos que

$$\frac{\hat{\beta}_1 - \beta_1}{\hat{\sigma}/S_{xx}^{1/2}} \sim t_{n-2}, \tag{5.29}$$

es decir, se distribuye como un t de Student con $n-2$ grados de libertad. Utilizando (5.29) podemos obtener un intervalo de confianza (por supuesto, no único) con nivel $1-\alpha$ para β_1 que tendría por extremos

$$\hat{\beta}_1 \pm t_{n-2,1-\alpha/2} \frac{\hat{\sigma}}{S_{xx}^{1/2}}. \tag{5.30}$$

Para contrastar: $H_0 : \beta_1 = b_1$ vs. $H_0 : \beta_1 \neq b_1$ utilizamos el estadístico

$$\frac{\hat{\beta}_1 - b_1}{\hat{\sigma}/S_{xx}^{1/2}} = \frac{(\hat{\beta}_1 - b_1)S_{xx}^{1/2}}{\hat{\sigma}} \sim t_{n-2}, \tag{5.31}$$

bajo la hipótesis nula. De otra forma, la distribución nula de $\frac{(\hat{\beta}_1 - b_1)S_{xx}^{1/2}}{\hat{\sigma}}$ es una t de Student con $n-2$ grados de libertad.

Ejemplo 5.4. *En el siguiente resumen podemos ver los contrastes para los coeficientes.*

```
summary(fit0)
```

```
Call:
lm(formula = expression ~time, data = df0)

Residuals:
    Min 1Q Median 3Q Max
-1.6009 -0.5772 0.2931 0.7483 1.1839

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
(Intercept) 7.96844 0.23130 34.451 < 2e-16
time -0.13307 0.01868 -7.122 3.84e-07

(Intercept) ***
time ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.8836 on 22 degrees of freedom
Multiple R^2: 0.6975, Adjusted R^2: 0.6837
F-statistic: 50.73 on 1 and 22 DF, p-value: 3.842e-07
```

Y los intervalos de confianza con nivel 0.95 lo tenemos con

```
confint(fit0)
```

```
              2.5 % 97.5 %
(Intercept) 7.4887585 8.4481140
time -0.1718183 -0.0943223
```

Y con nivel de confianza 0.99.

```
confint(fit0,level=.99)
```

```
              0.5 % 99.5 %
(Intercept) 7.3164703 8.62040218
time -0.1857356 -0.08040499
```

5.2.7 Comentarios

Hemos visto cómo realizar un análisis de regresión lineal simple con R y cómo obtener los estadísticos y contrastes más básicos. Veamos el uso de `str()` y `attributes()` para estudiar lo que la función `lm` nos devuelve cuando realizamos el ajuste de regresión. Lo primero es saber la clase del objeto que nos devuelve.

```
class(fit0)
```

```
[1] "lm"
```

Veamos los elementos que componen esta clase con `str`.

```
str(fit0)
```

Vemos que `lm` nos ha devuelto una lista (`List`) con distintos componentes. Es útil ir viendo qué es cada uno de los elementos que componen esta lista. Es claro que los coeficientes los tenemos con

```
fit0$coefficients
fit0$model
```

Con `attributes` tenemos una descripción más manejable de los elementos de la lista.

```
attributes(fit0)
```

```
$names
 [1] "coefficients" "residuals"
 [3] "effects" "rank"
 [5] "fitted.values" "assign"
 [7] "qr" "df.residual"
 [9] "xlevels" "call"
[11] "terms" "model"

$class
[1] "lm"
```

5.3 Análisis de la varianza con un factor fijo

Vamos a estudiar qué es el análisis de la varianza con un solo factor (fijo). Estudiamos de un modo sencillo la situación en que tenemos una variable respuesta cuantitativa y un único predictor de carácter categórico (factor experimental).

5.3.1 Análisis de la varianza de una vía

Con frecuencia tenemos una variable de interés Y y pretendemos estudiar su posible dependencia de un factor experimental. Por ejemplo, la variable respuesta puede cuantificar el resultado de un tratamiento médico y pretendemos comparar sus valores para distintos tratamientos. El experimentador tiene un **factor** de interés con distintos niveles y se pretende evaluar la dependencia de la variable respuesta de este factor experimental. Si el factor tiene dos niveles entonces podemos comparar las medias mediante un test de la t o t-test. Con más de dos niveles necesitamos otros procedimientos. Habitualmente, pero

no siempre, el factor corresponderá a una variable de control fijada por el experimentador. En este tema nos ocupamos (de un modo muy simple) de lo que se conoce como **experimentos con un solo factor completamente aleatorizado**.

Ejemplo 5.5. *Como ejemplo a analizar en esta sección vamos a seguir con los datos* `tamidata2::gse25171`*, en particular, la sonda* `261892_at`*. Podemos ver en las variables fenotípicas que tenemos definida la variable* `time2` *en la que el tiempo (*`time`*) es discretizado en dos valores (*`Short` *y* `Medium`*). También tenemos la variable categórica que nos indica la presencia o no de fosfatos (*`Pi`*). Vamos a construir una variable categórica (*`time2Pi`*) que recoge las cuatro combinaciones posibles de las dos variables binarias.*

```
time2Pi = vector("list",ncol(gse25171))
for(i in seq_along(time2Pi))
 time2Pi[[i]] = paste0(pData(gse25171)[,"time2"][i],
                       pData(gse25171)[,"Pi"][i])
time2Pi = factor(unlist(time2Pi))
```

Podemos ver los distintos valores que puede tomar la variable `time2Pi` ↪ .

```
levels(time2Pi)
```

```
[1] "MediumControl" "MediumTreatment"
[3] "ShortControl" "ShortTreatment"
```

Construimos un `data.frame` `df1` *en el que consideramos la expresión de la sonda y la variable que acabamos de construir.*

```
sel0 = which("261892_at"==fData(gse25171)[,"PROBEID"])
df1 = data.frame(time2Pi,expression=exprs(gse25171)[sel0,])
```

La variable respuesta es `expression` *y nuestra predictora es* `time2Pi`*.*

```
summary(df1[,"time2Pi"])
```

```
 MediumControl MediumTreatment ShortControl
             6 6 6
ShortTreatment
             6
```

Tenemos los cuatro grupos equilibrados. Pretendemos evaluar si los valores de la variable respuesta dependen del tratamiento. Quizás, para empezar, no viene mal hacer un diagrama de cajas comparando los valores de la variable respuesta en los grupos definidos por la variable predictora. En la figura 5.6 aparece el diagrama de cajas.

```
p = ggplot(df1,aes(x=time2Pi,y=expression)) + geom_boxplot()
```

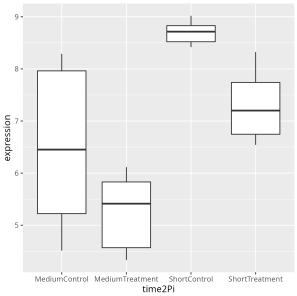


Figura 5.6: Diagrama de cajas mostrando la expresión en la sonda considerando el factor.

5.3.2 Comparando grupos

Supongamos que tenemos I condiciones distintas y en cada una de ellas n_i muestras de modo que $\sum_{i=1}^{I} n_i = n$, el total de muestras de las que disponemos. Supongamos que Y_{ij} denota la respuesta aleatoria en la j-ésima muestra de la i-ésima condición. Suponemos que los valores Y_{ij} con $j = 1, \ldots, n_i$ son independientes y con la misma distribución. El modelo de análisis de la varianza de una vía es

$$Y_{ij} = \mu_i + \epsilon_{ij}, \tag{5.32}$$

donde se asume que $\epsilon_{ij} \sim N(0, \sigma^2)$ y son independientes entre si para los distintos grupos y dentro de cada grupo.

Una formulación alternativa y habitual del modelo (5.32) es

$$Y_{ij} = \beta_0 + \beta_i + \epsilon_{ij}, \tag{5.33}$$

donde estamos expresando la media μ_i en el grupo i-ésimo de observaciones como

$$\mu_i = E[Y_{ij}] = \beta_0 + \beta_i. \tag{5.34}$$

Es claro que en la formulación (5.34) tenemos I medias μ_i pero $I+1$ parámetros β. En particular notemos que podemos sumar una cantidad δ a β_0 y restar esa misma cantidad a cada uno de los β_i con $i = 1, \ldots, I$ y las ecuaciones se mantienen. Esto es, tenemos un problema de identificabilidad de los parámetros. Por ello hemos de asumir una ecuación más. Lo habitual es considerar una categoría de referencia (por ejemplo pero no obligatoriamente la primera) y asumir que el parámetro es nulo. Por ejemplo, asumir que $\beta_1 = 0$.

Realmente estamos asumiendo en el modelo (5.34) que $Y_{ij} \sim N(\beta_0 + \beta_i, \sigma^2)$. La interpretación de los distintos parámetros es la siguiente:

1. β_0 es la media en el grupo de referencia.
2. β_i sería la diferencia de la media del grupo i respecto de la media del grupo de referencia.
3. ϵ_{ij} sería el error aleatorio de la observación j-ésima del grupo i-ésimo respecto de la media de la variable respuesta en el grupo, $\beta_0 + \beta_i$.

Se trata de evaluar si hay **diferencias entre grupos**. Bajo el modelo que acabamos de formular se traduce en la siguiente hipótesis nula,

$$H_0 : \beta_1 = \beta_2 = \ldots = \beta_I = 0,$$

frente a que algúno los β_i con $i \geq 2$ sea no nulo. Recordemos que asumimos $\beta_1 = 0$ por defecto.

¿Cómo podemos contrastar la hipótesis nula anterior? Si y_{ij} es la j-ésima muestra observada bajo la condición i ($i = 1, \ldots, I$ y $j = 1, \ldots, n_i$) entonces las medias muestrales para cada grupo serán

$$\bar{y}_{i\cdot} = \sum_{j=1}^{n_i} \frac{y_{ij}}{n_i},$$

y la media de todas las observaciones o media total será

$$\bar{y}_{\cdot\cdot} = \sum_{i=1}^{I} \sum_{j=1}^{n_i} \frac{y_{ij}}{n}.$$

Definimos la *suma de cuadrados intra* o **del error** como

$$SS(Within) = \sum_{i=1}^{I} \sum_{j=1}^{n_i} (y_{ij} - \bar{y}_{i\cdot})^2,$$

y la *suma de cuadrados entre* como

$$SS(Between) = \sum_{i=1}^{I} n_i (\bar{y}_{i\cdot} - \bar{y}_{\cdot\cdot})^2.$$

El estadístico para contrastar esta hipótesis nula es

$$F = \frac{SS(Between)/(I-1)}{SS(Within)/(n-I)}.$$

Estas sumas de cuadrados se suelen disponer en forma de tabla.

Tabla 5.4: Tabla de análisis de la varianza.

Source	SS	df	MS	F	p
Between	$SS(Between)$	$I-1$	$SS(Between)/(I-1)$	$F = \frac{SS(Between)/(I-1)}{SS(Within)/(n-I)}$	$P(>F)$
Within	$SS(Within)$	$n-I$	$SS(Within)/(n-I)$		
Total	$SS(Between)+SS(Within)$				

Bajo la hipótesis nula de que todas las medias son la misma (y puesto que asumimos una misma varianza) tendríamos una distribución común bajo todas las condiciones. Asumiendo la hipótesis nula el estadístico F se distribuye como un F con $I-1$ y $n-I$ grados de libertad ,

$$F \sim F_{I-1,n-I}.$$

Es claro que, bajo la hipótesis alternativa, los valores de F tenderán a ser **grandes** o mayores que los esperables bajo la hipótesis nula. En resumen, la **región crítica** (donde rechazamos la hipótesis nula) será un intervalo de la forma $[c,+\infty)$. Si tomamos como valor c el valor observado tendremos el p-valor.

Ejemplo 5.6. *Consideramos como variable respuesta* **`expression`** *y como factor experimental (predictora) la variable* **`time2Pi`**.

```
fit2 = aov(expression ~time2Pi,data=df1)
summary(fit2)
```

```
            Df Sum Sq Mean Sq F value Pr(>F)
time2Pi 3 37.52 12.505 12.98 6.22e-05
Residuals 20 19.26 0.963

time2Pi ***
Residuals
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Como vemos resulta significativa la diferencia entre medias con un nivel de significación de 0.05.

5.3.3 Representación matricial

Vamos a considerar el vector de respuestas aleatorias de modo que las n_1 primeras posiciones las ocupan $Y_{11},\ldots,Y_{1n_1}$, las n_2 posiciones siguientes $Y_{21},\ldots,Y_{2n_2}$ y así sucesivamente. A este vector lo denotamos $\boldsymbol{Y}$ como es habitual. Haciendo lo análogo con los errores aleatorios tendríamos el vector $\boldsymbol{\epsilon}$ obtenido acumulando los errores de cada uno de los grupos considerados en el mismo orden. El modelo propuesto en (5.33) se puede escribir como considerando como categoría de referencia el primer grupo se puede escribir como

$$E[Y_{1j}] = \beta_0$$

y

$$E[Y_{ij}] = \beta_0 + \beta_i$$

para $i = 2, \ldots, I$. De un modo conjunto:

$$E[Y_{ij}] = \beta_0 + \beta_2 v_{2j} + \ldots + \beta_I v_{Ij}$$

donde $v_{ij} = 1$ si estamos en el grupo i y cero en otro caso. Volviendo a considerar el vector $\boldsymbol{Y}$ en donde apilamos las respuestas según el orden del grupo tendríamos el siguiente modelo,

$$\boldsymbol{Y} = \begin{bmatrix} \mathbf{1}_{n_1} & \mathbf{0}_{n_1} & \cdots & \mathbf{0}_{n_1} \\ \mathbf{1}_{n_2} & \mathbf{1}_{n_2} & \cdots & \mathbf{0}_{n_2} \\ \vdots & \vdots & \vdots & \vdots \\ \mathbf{1}_{n_I} & \mathbf{0}_{n_I} & \cdots & \mathbf{1}_{n_I} \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_2 \\ \vdots \\ \beta_I \end{bmatrix} + \boldsymbol{\epsilon}.$$

La matriz de modelo tiene rango completo I ya que las columnas son linealmente independientes.

Ejemplo 5.7. *El modelo que acabamos de considerar con una categoría de referencia se puede ajustar del siguiente modo.*

```
fit3 = lm(expression ~time2Pi,data=df1)
```

Los coeficientes ajustados son

```
coef(fit3)
```

```
          (Intercept) time2PiMediumTreatment
            6.4975788 -1.2418791
   time2PiShortControl time2PiShortTreatment
            2.2010323 0.7990972
```

Veamos qué ofrece el resumen del modelo.

```
summary(fit3)
```

```
Call:
lm(formula = expression ~time2Pi, data = df1)

Residuals:
     Min 1Q Median 3Q Max
-1.98463 -0.62805 0.04225 0.54559 1.79155

Coefficients:
                    Estimate Std. Error
(Intercept) 6.4976 0.4007
time2PiMediumTreatment -1.2419 0.5666
time2PiShortControl 2.2010 0.5666
time2PiShortTreatment 0.7991 0.5666
                    t value Pr(>|t|)
(Intercept) 16.217 5.66e-13 ***
time2PiMediumTreatment -2.192 0.040407 *
time2PiShortControl 3.884 0.000922 ***
time2PiShortTreatment 1.410 0.173828
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.9814 on 20 degrees of freedom
Multiple R^2: 0.6607, Adjusted R^2: 0.6098
F-statistic: 12.98 on 3 and 20 DF, p-value: 6.219e-05
```

Es distinto.

5.4 Mínimos cuadrados

[42] [4], capítulo 2.

[42] Esta sección considera la situación en que pretendemos predecir una variable respuesta continua y tenemos más de un predictor.

5.4.1 El problema

Disponemos de los datos $(\boldsymbol{x}_i, y_i)$ con $i = 1, \ldots, n$ siendo $\boldsymbol{y} = (y_1, \ldots, y_n)^T$, las respuestas observadas; $\boldsymbol{x}_i = (x_{i1}, \ldots, x_{ip})^T$, los predictores correspondientes a la i-ésima observación. La matriz de modelo que recoge los valores de los predictores:

$$\boldsymbol{X} = \begin{bmatrix} x_{11} & \ldots & x_{1p} \\ \vdots & \vdots & \vdots \\ x_{n1} & \ldots & x_{np} \end{bmatrix}$$

En las secciones anteriores hemos visto modelos (regresión lineal simple y análisis de la varianza de un factor fijo) en donde, para el vector aleatorio de respuestas $\boldsymbol{Y} = (Y_1, \ldots, Y_n)^T$, su vector de medias $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)^T$ (con medias $\mu_i = E[Y_i|\boldsymbol{x}_i]$) verifica

$$\boldsymbol{\mu} = \boldsymbol{X}\boldsymbol{\beta}.$$

¿Cómo estimamos $\boldsymbol{\beta}$?

5.4.2 Planteamiento

Se trata de estimar $\boldsymbol{\beta}$. Estamos relacionando el vector de medias con los predictores. No conocemos $\boldsymbol{\mu}$. La idea es sustituir la media μ_i por el valor observado y_i. ¿Podemos resolver el siguiente sistema?

$$\boldsymbol{y} = \boldsymbol{X}\boldsymbol{\beta}.$$

Este sistema no tiene solución. Es un sistema sobre determinado en el cual tenemos más ecuaciones que incógnitas. Sustituimos la idea de resolver el sistema por la de buscar una buena solución. Una opción clásica son los mínimos cuadrados en donde pretendemos que

$$\|\boldsymbol{y} - \hat{\boldsymbol{\mu}}\|^2$$

sea mínimo donde $\hat{\boldsymbol{\mu}} = \boldsymbol{X}\hat{\beta}$. Obviamente lo que estamos haciendo es sustituir el vector de medias desconocido por los valores observados.

$$\|\boldsymbol{y} - \hat{\boldsymbol{\mu}}\|^2 = \sum_{i=1}^{n}(y_i - \hat{\mu}_i)^2 = \sum_{i=1}^{n}(y_i - \sum_{j=1}^{p}\hat{\beta}_j x_{ij})^2 \tag{5.35}$$

Consideramos la función

$$S_t(\boldsymbol{\beta}) = \sum_{i=1}^{n}(y_i - \mu_i)^2 = \sum_{i=1}^{n}\left(y_i - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2.$$

Los estimadores mínimo cuadráticos minimizan la función S_t.

5.4.3 Predicciones y la matriz H

Tenemos que

$$\boldsymbol{\mu} = \boldsymbol{X}\boldsymbol{\beta},$$

por tanto, las medias estimadas las obtenemos con

$$\hat{\boldsymbol{\mu}} = \boldsymbol{X}\hat{\boldsymbol{\beta}} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y}.$$

Si denotamos

$$\boldsymbol{H} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T, \tag{5.36}$$

entonces

$$\hat{\boldsymbol{\mu}} = \boldsymbol{H}\boldsymbol{y}.$$

La matriz $\boldsymbol{H}$ recibe el nombre de matriz sombrero (hat matrix) o matriz de influencia.

Ejemplo 5.8. *Vamos a analizar los datos* `df0`*. Como variable respuesta utilizaremos la variable* `expression` *y como predictoras* `time` *y* `Pi`*. Vamos a obtener los coeficientes mínimo cuadráticos. Realizamos el ajuste.*

```
fit4 = lm(expression~ time + Pi,data=df0)
```

Podemos ver que en el `data.frame` *solamente tenemos las variables que estamos utilizando. Por ello, es suficiente especificar quién es la respuesta e indicar con un punto que utilizamos las demás como predictoras. Ambos códigos son equivalentes.*

```
fit4 = lm(expression~.,data=df0)
```

La matriz modelo es (mostramos primeras columnas).

```
head(model.matrix(fit4),n=5)
```

```
               (Intercept) time PiTreatment
GSM618324.CEL.gz 1 0 1
GSM618325.CEL.gz 1 0 0
GSM618326.CEL.gz 1 1 1
GSM618327.CEL.gz 1 1 0
GSM618328.CEL.gz 1 6 1
```

También la podemos obtener con (no mostramos la salida).

```
fit4$model
```

¿Cómo podemos obtener la matriz $\boldsymbol{H}$*?*

```
X = model.matrix(fit4) ## Usamos base::model.matrix
H = X %*% solve(t(X) %*% X) %*% t(X) ## Cálculo explícito
```

donde `%*%` *es el producto matricial,* `base::solve()` *nos devuelve la inversa de la matriz y* `base::t()` *es la matriz traspuesta. Los residuos los obtenemos con*

```
head(resid(fit4))
```

```
GSM618324.CEL.gz GSM618325.CEL.gz
     1.01552765 0.21676116
GSM618326.CEL.gz GSM618327.CEL.gz
    -0.32266386 -0.07784405
GSM618328.CEL.gz GSM618329.CEL.gz
    -0.59106601 0.31824015
```

5.4.4 Estimando la variación

Hasta ahora no nos hemos ocupado de la varianza del error. Asumimos errores independientes y equidistribuidos con una varianza constante. Abreviadamente, el modelo lineal que consideramos incorporando el error aleatorio es

$$\boldsymbol{Y} = \boldsymbol{X}\boldsymbol{\beta} + \boldsymbol{\epsilon},$$

con $var(\boldsymbol{\epsilon}) = \sigma^2\boldsymbol{I}$. En esta sección proponemos el estimador (insesgado) de σ^2 habitualmente utilizado.

Estimando la varianza

El resultado fundamental es el siguiente.

Proposición 5.1.

$$E\left[\frac{\boldsymbol{Y}^T(\boldsymbol{I}-\boldsymbol{P_X})\boldsymbol{Y}}{n-p}\right] = E\sum_{i=1}^{n}\frac{(Y_i-\hat{\mu}_i)^2}{n-p} = \sigma^2. \tag{5.37}$$

siendo $\boldsymbol{P_X} = \boldsymbol{H}$.

Por tanto, un estimador insesgado de la varianza del error aleatorio sería

$$S^2 = \sum_{i=1}^{n}\frac{(Y_i-\hat{\mu}_i)^2}{n-p}, \tag{5.38}$$

esto es, la suma de los cuadrados de los residuos dividida por el número de observaciones menos el número de parámetros. S^2 recibe el nombre de **cuadrado medio del error** o **cuadrado medio residual**. Su raiz cuadrada, S, es el **error estándar residual**. Por la proposición 5.1, S^2 es un estimador insesgado de σ^2 mientras que S no es un estimador insesgado de σ.

Ejemplo 5.9. *Si consideramos el modelo lineal solamente con un término constante:* $\boldsymbol{X} = \boldsymbol{1}_n$, $\hat{\mu}_i = \bar{Y}$ *y el estimador de* (5.38) *sería*

$$S^2 = \sum_{i=1}^{n}\frac{(Y_i-\bar{Y})^2}{n-1},$$

que es insesgado.

Ejemplo 5.10. *El error estándar residual lo tenemos con*

```
summary(fit4)$sigma
```

```
[1] 0.5644856
```

Sumas de cuadrados

Definición 5.2. *La* ***suma de cuadrados total*** *(o* ***suma de cuadrados corregida****) se define como*

$$SS(Total) = \sum_{i=1}^{n} y_i^2 - n\bar{y}^2 = \sum_{i=1}^{n}(y_i-\bar{y})^2.$$

La ***suma de cuadrados de la regresión*** *o* ***suma de cuadrados del modelo*** *es*

$$SS(Regression) = \sum_{i=1}^{n} (\hat{\mu}_i - \bar{y})^2.$$

La ***suma de cuadrados residual*** *o* ***suma de cuadrados del error*** *es*

$$SS(Residual) = \sum_{i=1}^{n} (y_i - \hat{\mu}_i)^2.$$

Proposición 5.2.

$$SS(Total) = SS(Regression) + SS(Residual). \tag{5.39}$$

En ocasiones a $\sum_{i=1}^{n} y_i^2$ se le llama **suma de cuadrados total** pero es más frecuente la notación que utilizamos aquí.

Es claro que un mejor ajuste corresponde con una $SS(Residual)$ pequeña y una $SS(Regression)$ grande. Notemos que la suma de las dos es fija dados los datos.

Coeficiente de determinación y correlación múltiple

Obviamente un ajuste es tanto mejor cuando menor es la suma de cuadrados residual y mayor la suma de cuadrados de la regresión. Una medida razonable de cuantificar la bondad del ajuste es considerar el siguiente cociente conocido como **coeficiente de determinación**. [43]

[43] Lo veremos etiquetado con **R-squared**.

$$R^2 = \frac{SS(Regression)}{SS(Total)} = \frac{SS(Total) - SS(Residual)}{SS(Total)} = \frac{\sum_{i=1}^{n}(y_i - \bar{y})^2 - \sum_{i=1}^{n}(y_i - \hat{\mu}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2}. \tag{5.40}$$

También es razonable cuantificar la calidad del ajuste viendo si los valores que intentamos predecir están correlados con las predicciones que hemos obtenido, esto es, considerar el coeficiente de correlación muestral

$$corr(\boldsymbol{y}, \hat{\boldsymbol{\mu}}) = \frac{\sum_{i=1}^{n}(y_i - \bar{y})(\hat{\mu}_i - \bar{\hat{\mu}})}{\sqrt{\sum_{i=1}^{n}(y_i - \bar{y})^2 \sum_{i=1}^{n}(\hat{\mu}_i - \bar{\hat{\mu}})^2}}. \tag{5.41}$$

Un valor extremo de la correlación muestral indicaría una clara asociación y por lo tanto un buen ajuste.

Proposición 5.3.

$$corr(\boldsymbol{y}, \hat{\boldsymbol{\mu}}) = +\sqrt{R^2}.$$

La raiz cuadrada positiva de R^2 recibe el nombre de **correlación múltiple**. Tenemos $0 \leq R \leq 1$. Con una sola variable predictora entonces R es el coeficiente de correlación entre esta variable predictora y la variable respuesta como vimos en § 5.2.3.

Por la propia definición del coeficiente de determinación más variables en el modelo supone un incremento de R^2. Una cuantificación muy similar pero que no necesariamente verifica esto es la R^2 ajustada definida como

$$R^2_{adjusted} = 1 - \frac{SS(Residual)/(n-p)}{SS(Total)/(n-1)} = 1 - \frac{n-1}{n-p}(1 - R^2). \tag{5.42}$$

No hay grandes diferencias en su uso con la R^2.

5.4.5 Residuos, leverages e influencia

En esta sección nos ocupamos de valorar las hipótesis del modelo lineal normal. Una referencia útil es [39].

Incorrelación y normalidad

Los residuos contienen información de los datos que el modelo no es capaz de explicar.

Proposición 5.4. *Los residuos y los valores ajustados están incorrelados.*

Atendiendo a la proposición una forma de evaluar el modelo sería un diagrama de puntos en donde representemos los residuos frente a las predicciones o valores ajustados. Debiéramos de observar una correlación nula. Un diagrama sin forma. Una varianza no constante (heterocedasticidad) o bien un modelo para la media no lineal debiera de dar una representación con forma. En los siguientes ejemplos vemos el dibujo de residuos frente a predicciones con un modelo correcto, con un modelo donde la varianza no es constante y, finalmente, con un modelo donde la dependencia no es lineal.

Ejemplo 5.11 (Modelo correcto)**.** *Generamos los datos.*

```
pacman::p_load(ggplot2)
x = seq(0,3,length.out=100)
beta0 = 2
beta1 = 3
y = beta0 + beta1* x + rnorm(100,sd=1)
df = data.frame(x,y)
fit1 = lm(y ~x,data=df)
df1 = data.frame(hatmu = predict(fit1),e = resid(fit1))
```

```
ggplot(df1,aes(x=hatmu,y=e)) + geom_point()
```

En la figura 5.7 mostramos con los datos que acabamos de generar el dibujo de predicciones en abscisas frente a residuos en ordenadas.

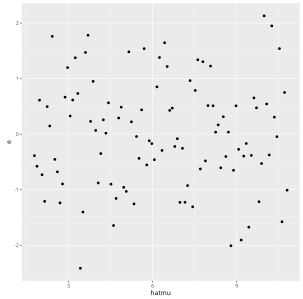


Figura 5.7: Residuos frente a predicciones con el modelo verificando tanto linealidad como homocedasticidad.

Ejemplo 5.12 (Heterocedasticidad)**.** *Generamos los datos con varianza creciente.*

```
pacman::p_load(ggplot2)
x = seq(0,3,length.out=100)
beta0 = 2
beta1 = 3
y = beta0 + beta1 * x + rnorm(100,mean=0,sd=x)
df = data.frame(x,y)
fit2 = lm(y ~x,data=df)
df1 = data.frame(hatmu = predict(fit2),e = resid(fit2))
```

```
ggplot(df1,aes(x=hatmu,y=e)) + geom_point()
```

En la figura 5.8 mostramos con los datos que acabamos de generar el dibujo de predicciones en abscisas frente a residuos en ordenadas.

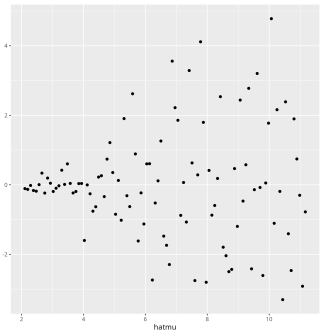


Figura 5.8: Residuos frente a predicciones con varianza no constante.

Ejemplo 5.13 (No linealidad)**.** *Generamos los datos donde la dependencia de la media no es lineal.*

```
x = seq(0,3,length.out=100)
beta0 = 2
beta1 = 3
y = beta0 + beta1 * x^2 + rnorm(100,mean=0,sd=1)
```

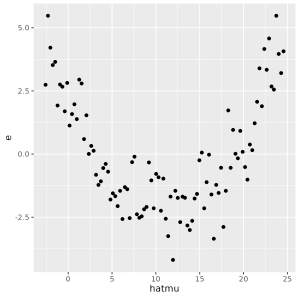


Figura 5.9: Residuos frente a predicciones con dependencia no lineal de la media.

```
df = data.frame(x,y)
fit3 = lm(y ~x,data=df)
df1 = data.frame(hatmu = predict(fit3),e = resid(fit3))
```

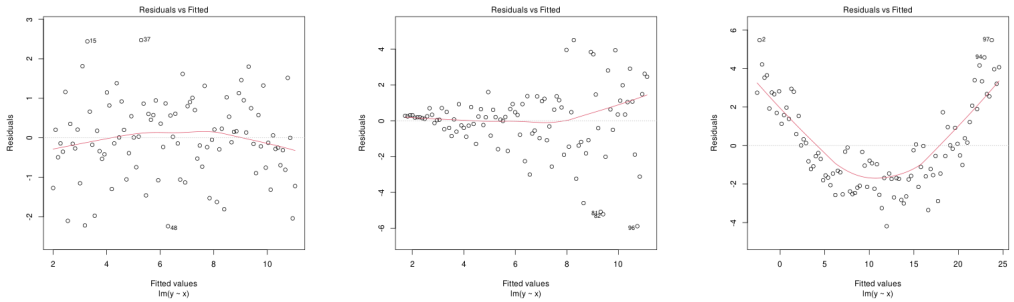
En la figura 5.9 mostramos con los datos que acabamos de generar el dibujo de predicciones en abscisas frente a residuos en ordenadas.

```
ggplot(df1,aes(x=hatmu,y=e)) + geom_point()
```

Los dibujos anteriores pueden ser generados utilizando el método correspondiente a la función genérica **`plot`** para un objeto de clase **`lm`**.

```
plot(fit1,which=1)
plot(fit2,which=1)
plot(fit3,which=1)
```

Y la representación conjunta la tenemos en la figura 5.10.

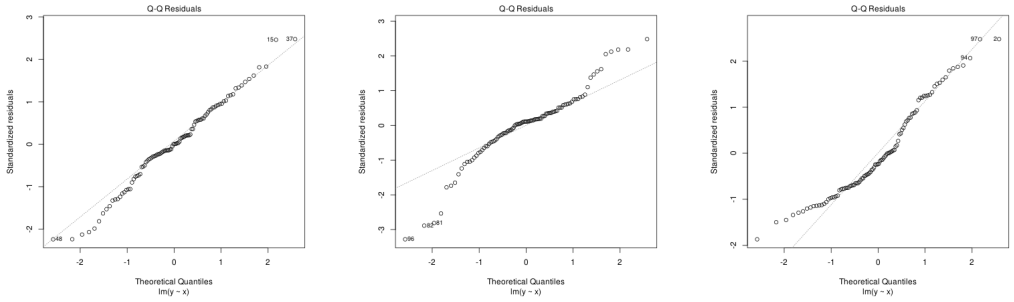


(a) (b) (c)

Figura 5.10: Residuos frente a valores ajustados o predicciones. (a) Modelo correcto. (b) Heterocedasticidad. (c) No linealidad.

Los residuos tienen una distribución normal. Por ello una forma de evaluar la validez de la hipótesis sobre la distribución del error es un dibujo cuantil-cuantil. En la figura 5.11 tenemos los dibujos Q-Q para los tres conjuntos de residuos con los datos generados.

```
plot(fit1,which=2)
plot(fit2,which=2)
plot(fit3,which=2)
```



(a) (b) (c)

Figura 5.11: Dibujos cuantil-cuantil para los residuos: (a) Modelo correcto. (b) Heterocedasticidad. (c) No linealidad.

Residuos estandarizados y studentizados

En el modelo lineal $var(\boldsymbol{Y}) = \sigma^2 \boldsymbol{I}_n$. Si consideramos la matriz de proyección $\boldsymbol{H}$ tenemos

$$\sigma^2 \boldsymbol{I}_n = \sigma^2 \boldsymbol{H} + \sigma^2(\boldsymbol{I} - \boldsymbol{H}).$$

Pero $\hat{\boldsymbol{\mu}} = \boldsymbol{H}\boldsymbol{Y}$[44] de donde

$$var(\hat{\boldsymbol{\mu}}) = \boldsymbol{H} var(\boldsymbol{Y}) \boldsymbol{H}' = \sigma^2 \boldsymbol{H}.$$

[44] Denotamos las predicciones aleatorias con las predicciones, esto es, con $\hat{\boldsymbol{\mu}}$.

En consecuencia,

$$var(\hat{\mu}_i) = \sigma^2 h_{ii},$$

donde $\boldsymbol{H} = [h_{ij}]$, es decir, h_{ii} es el elemento i-ésimo de la diagonal principal de la matriz $\boldsymbol{H}$. Notemos que h_{ii} es una varianza y por lo tanto ha de ser no negativo. Tenemos $\boldsymbol{Y} - \hat{\boldsymbol{\mu}} = (\boldsymbol{I} - \boldsymbol{H})\boldsymbol{Y}$. La matriz $\boldsymbol{I} - \boldsymbol{H}$ es idempotente y se sigue que

$$var(\boldsymbol{e}) = var(\boldsymbol{Y} - \hat{\boldsymbol{\mu}}) = (\boldsymbol{I} - \boldsymbol{H}) var(\boldsymbol{Y}) (\boldsymbol{I} - \boldsymbol{H})^T = \sigma^2 (\boldsymbol{I} - \boldsymbol{H}).$$

Vemos que los residuos están correlados y la varianza del i-ésimo residuo vendrá dada por

$$var(e_i) = var(y_i - \hat{\mu}_i) = \sigma^2 (1 - h_{ii}).$$

Se ha de dar que $var(e_i) \geq 0$ por lo que $h_{ii} \leq 1$. En resumen tenemos que

$$0 \leq h_{ii} \leq 1.$$

Definición 5.3 (Leverage)**.** [2] *Se define el leverage de la observación i-ésima como el valor* h_{ii}.

Puesto que $var(\hat{\mu}_i) = \sigma^2 h_{ii} \leq \sigma^2 = var(Y_i)$. Si nos planteamos estimar μ_i con Y_i o con $\hat{\mu}_i$ entonces ambos estimadores son insesgados pero $\hat{\mu}_i$ tiene una varianza menor que Y_i.

Definición 5.4 (Residuo estandarizado)**.**

$$r_i = \frac{y_i - \hat{\mu}_i}{s\sqrt{1 - h_{ii}}}.$$

siendo $s = \sqrt{\sum_{i=1}^{n} \frac{e_i^2}{n-p}}$.

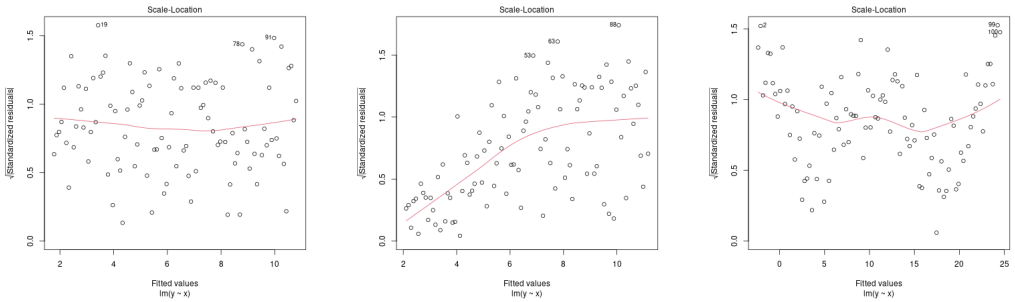
Bajo la hipótesis de normalidad de los residuos se debe de dar que la mayoría de los residuos estandarizados deben de estar en el intervalo $[-3, 3]$ y, además, no debieran de tener dependencia de los valores ajustados.

El **residuo studentizado** se define como el residuo estandarizado lo que ocurre es que el ajuste se realiza sin considerar el propio punto.

Ejemplo 5.14. *En la figura 5.12 mostramos los residuos estandarizados en los tres ejemplos generados.*

```
plot(fit1,which=3)
plot(fit2,which=3)
plot(fit3,which=3)
```

[2] Sobre la traducción de leverage. Posiblemente la mejor sea **apalancamiento**. Al menos la que más me gusta. Sin embargo, al ser un uso muy técnico de la palabra la mantenemos en su versión original.



(a) (b) (c)

Figura 5.12: Residuos estandarizados frente a valores ajustados: (a) Modelo correcto. (b) Heterocedasticidad. (c) No linealidad.

Ejemplo 5.15 (df0)**.** *Los residuos estandarizados los podemos obtener con*

```
head(rstandard(df0.fit))
```

```
GSM618324.CEL.gz GSM618325.CEL.gz
       1.9071725 0.4070799
GSM618326.CEL.gz GSM618327.CEL.gz
      -0.6037708 -0.1456623
GSM618328.CEL.gz GSM618329.CEL.gz
      -1.0944653 0.5892791
```

mientras que los residuos studentizados se obtienen con

```
head(MASS::studres(df0.fit))
```

```
GSM618324.CEL.gz GSM618325.CEL.gz
       2.0468991 0.3988461
GSM618326.CEL.gz GSM618327.CEL.gz
      -0.5944017 -0.1422237
GSM618328.CEL.gz GSM618329.CEL.gz
      -1.0999195 0.5798919
```

Leverage e influencia

Recordemos que $var(e_i) = var(y_i - \hat{\mu}_i) = \sigma^2(1 - h_{ii})$ y que $var(\hat{\mu}_i) = \sigma^2 h_{ii}$. Supongamos que el valor de h_{ii} es próximo a 1. Entonces tendríamos que $var(\hat{\mu}_i) \approx \sigma^2$ y que $var(e_i) \approx 0$. Indica que $\hat{\mu}_i$ depende fundamentalmente de y_i y poco de las demás observaciones. En el extremo si $h_{ii} = 1$ tenemos $\hat{\mu}_i = y_i$.[45] Con un valor de $h_{ii} = 0$ tendríamos la situación opuesta. Tendríamos $var(\hat{\mu}_i)$ pequeña y su estimación depende mucho de todas las observaciones. Observaciones con un alto leverage podrían ser influyentes en el ajuste. Si tenemos p variables predictoras (con la constante) entonces un valor grande podría ser mayor que $3p/n$.

[45] De hecho, con probabilidad uno.

Solamente un valor grande del leverage no indica que la observación pueda influir mucho en el ajuste globalmente. Un punto influyente es el que modifica (en exceso) el ajuste. Y un ajuste no es razonable que dependa de una sola observación. Realmente lo que define una observación influyente debe combinar un leverage grande y un residuo estandarizado grande.

Definición 5.5 (Distancia de Cook)**.** *Supongamos que* $\hat{\boldsymbol{\beta}}_{(i)}$ *son los estimadores de los coeficientes sin la* i*-ésima observación.*

$$D_i = \frac{(\hat{\boldsymbol{\beta}}_{(i)} - \hat{\boldsymbol{\beta}})^T [\widehat{var(\hat{\boldsymbol{\beta}})}]^{-1} (\hat{\boldsymbol{\beta}}_{(i)} - \hat{\boldsymbol{\beta}})}{p} = \frac{(\hat{\boldsymbol{\beta}}_{(i)} - \hat{\boldsymbol{\beta}})^T [\boldsymbol{X}^T \boldsymbol{X}] (\hat{\boldsymbol{\beta}}_{(i)} - \hat{\boldsymbol{\beta}})}{ps^2}.$$

Se tiene que

$$D_i = r_i^2 \left(\frac{h_{ii}}{p(1-h_{ii})} \right) = \frac{(y_i - \hat{\mu}_i)^2 h_{ii}}{ps^2 (1-h_{ii})^2}.$$

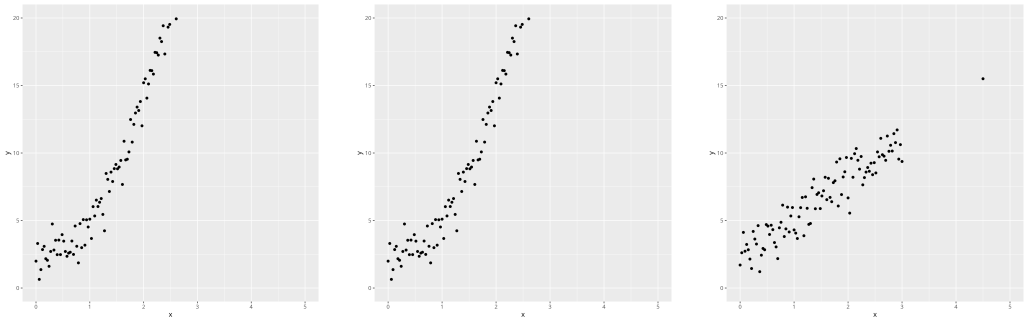
Si el residuo estandarizado y el leverage son grandes entonces también lo es D_i.

Ejemplo 5.16. *Generamos datos en los que no tenemos puntos con leverages altos. Y luego añadimos un punto con leverage alto e influyente en una segunda matriz de datos. El tercer banco de datos tiene un punto de leverage alto pero no influyente.*

```
x = seq(0,3,length.out=100)
beta0 = 2
beta1 = 3
y = beta0 + beta1 * x + rnorm(100,mean=0,sd=1)
df = data.frame(x,y) ## Datos sin leverages altos
p1 = c(4.5,3) ## Punto influyente
df1 = rbind(df,p1)
p2 = c(4.5,beta0+beta1*4.5) ## Punto no influyente
df2 = rbind(df,p2)
```

En la figura 5.13 mostramos los tres bancos de datos.

```
ggplot(df,aes(x=x,y=y)) + geom_point() + xlim(0,5)+ylim(0,20)
ggplot(df1,aes(x=x,y=y)) + geom_point() + xlim(0,5)+ylim(0,20)
ggplot(df2,aes(x=x,y=y)) + geom_point() + xlim(0,5)+ylim(0,20)
```



(a) (b) (c)

Figura 5.13: (a) Datos originales. (b) Datos originales con punto de leverage alto e influyente. (c) Datos originales con punto no influyente de alto leverage.

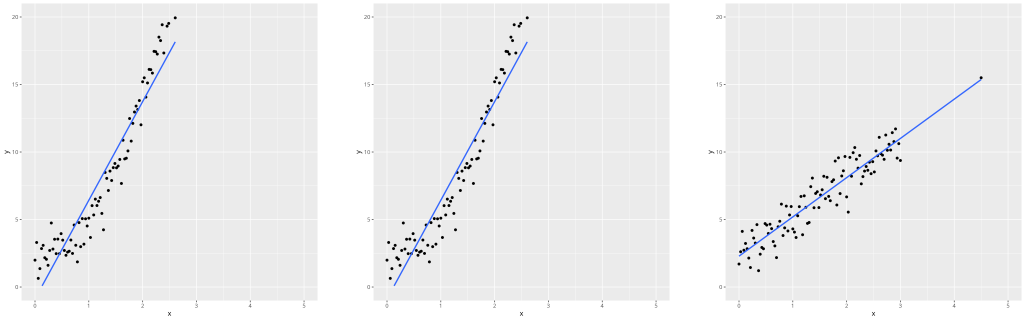
Ajustamos los tres modelos.

```
fit1 = lm(y ~x,data=df); fit11 = lm(y ~x,data=df1); fit12 = lm(y ~x,data=df2
    ↪ )
```

Representamos en la figura 5.14 los modelos ajustados junto con los datos originales.

```
p1= ggplot(df,aes(x=x,y=y)) + geom_point() + xlim(0,5)+ylim(0,20)
p1+ geom_smooth(method='lm', se = FALSE)
```

```
p2= ggplot(df1,aes(x=x,y=y)) + geom_point() + xlim(0,5)+ylim(0,20)
p2 + geom_smooth(method='lm', se = FALSE)
p3 = ggplot(df2,aes(x=x,y=y)) + geom_point() + xlim(0,5)+ylim(0,20)
p3 + geom_smooth(method='lm', se = FALSE)
```

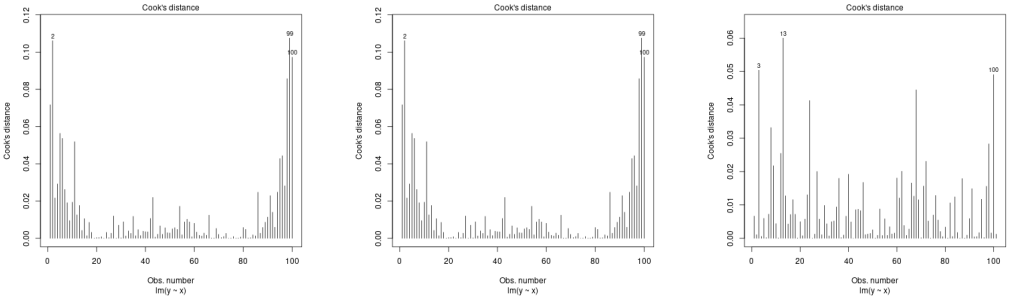


(a) (b) (c)

Figura 5.14: Datos originales y modificados incluyendo la correspondiente recta de regresión.

En la figura 5.15 mostramos las distancias de Cook.

```
plot(fit1,which=4)
plot(fit11,which=4)
plot(fit12,which=4)
```



(a) (b) (c)

Figura 5.15: Distancias de Cook: (a) Datos originales. (b) Datos originales con punto de leverage alto e influyente. (c) Datos originales con punto no influyente de alto leverage.

Finalmente, consideramos una representación conjunta de leverages, residuos estandarizados y distancias de Cook.

```
plot(fit1,which=5)
plot(fit11,which=5)
plot(fit12,which=5)
```

Ejemplo 5.17 (df0)**.** *Vamos a analizar los datos faraway::df0.*

```
plot(df0.fit,which=1)
plot(df0.fit,which=2)
plot(df0.fit,which=3)
plot(df0.fit,which=4)
plot(df0.fit,which=5)
```

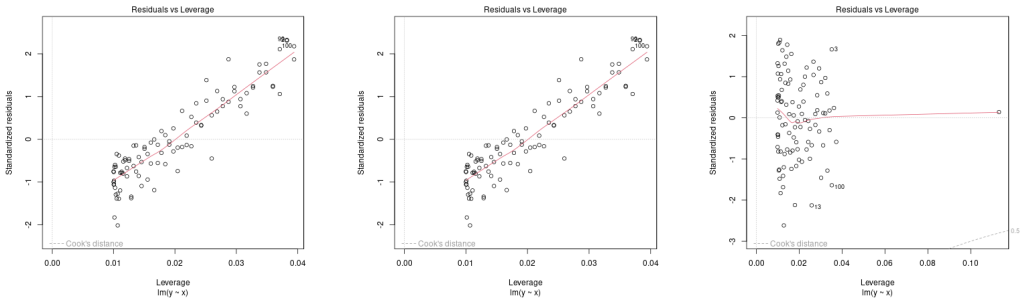


Figura 5.16: Residuos estandarizados frente a leverages considerando las distancias de Cook. (a) Datos originales. (b) Datos originales con punto de leverage alto e influyente. (c) Datos originales con punto no influyente de alto leverage.

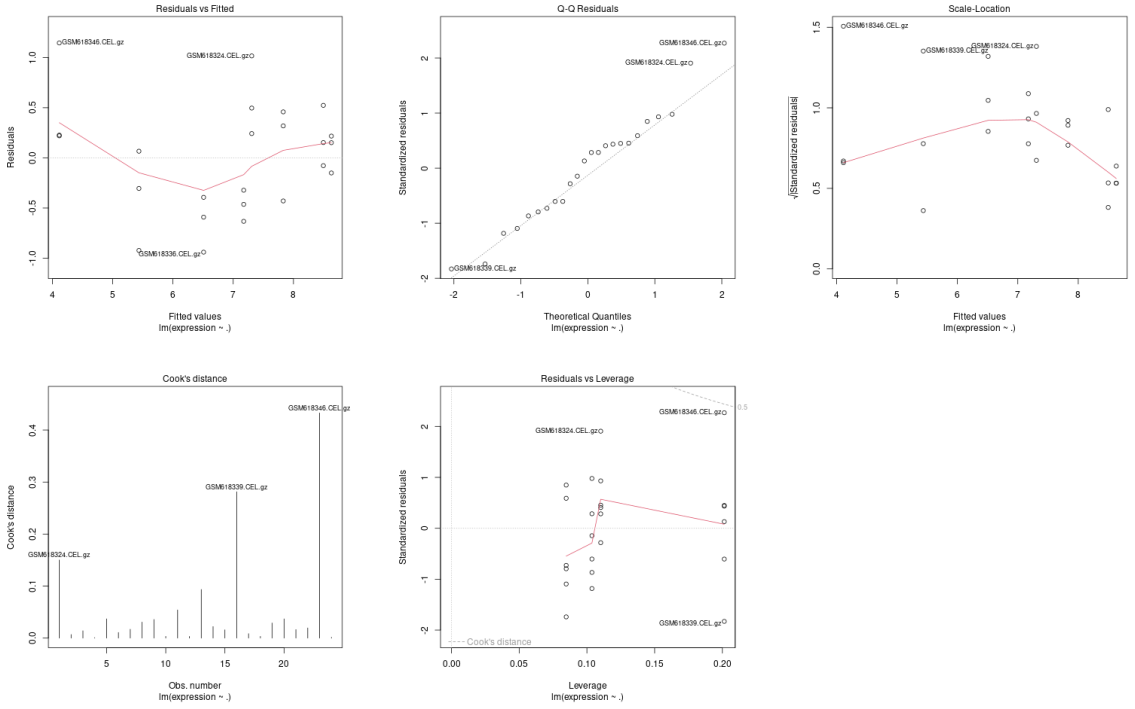


Figura 5.17: Diagnóstico de modelo para df0.fit

5.5 Muchos modelos

Hemos elegido trabajar con una sonda elegida de un modo arbitrario. Tenemos interés en todas las sondas. Hemos de ajustar un modelo lineal para cada una de estas sondas. Es claro que los predictores serán variables fenotípicas compartidas por todas las sondas. Por tanto los distintos modelos comparten los predictores y difieren en la variable respuesta. Veamos algunas opciones para hacerlo.

GSEAlm

Podemos utilizar el paquete [70].

```
pacman::p_load(GSEAlm)
data(gse25171,package="tamidata2")
fits1 = GSEAlm::lmPerGene(gse25171,~ time + Pi)
```

La matriz de modelo común a todos los ajustes la tenemos con

```
head(fits1$x)
```

```
              (Intercept) time PiTreatment
GSM618324.CEL.gz 1 0 1
GSM618325.CEL.gz 1 0 0
GSM618326.CEL.gz 1 1 1
GSM618327.CEL.gz 1 1 0
GSM618328.CEL.gz 1 6 1
GSM618329.CEL.gz 1 6 0
```

La matriz de proyección sobre el espacio modelo o matriz hat $\boldsymbol{H}$ con

```
fits1$Hmat
```

Cada uno de los ajustes correspondiendo con cada una de las filas de la matriz de expresión nos proporciona un vector de coeficientes que podemos obtener con

```
fits1$coefficients
```

Cada columna de la matriz anterior corresponde con cada fila de la matriz de expresión. Por ejemplo para la primera sonda tenemos los coeficientes.

```
fits1$coefficients[,1]
```

```
(Intercept) time PiTreatment
5.080830082 0.003844458 0.010492799
```

limma

Podemos utilizar para ajustar todos los modelos el paquete [81, limma].

```
design = model.matrix(~ pData(gse25171)[,"time"] + pData(gse25171)[,"Pi"])
fits2 = limma::lmFit(gse25171,design)
```

Los coeficientes los tenemos con

```
head(fits2$coefficients,n=2)
```

```
        (Intercept) pData(gse25171)[, "time"]
244901_at 5.080830 0.003844458
244902_at 4.843889 -0.005471131
        pData(gse25171)[, "Pi"]Treatment
244901_at 0.0104927991
244902_at -0.0009233809
```

En este caso cada fila corresponde con un ajuste de modo que los coeficientes del ajuste para la primera sonda es

```
fits2$coefficients[1,]
```

```
                  (Intercept)
                  5.080830082
      pData(gse25171)[, "time"]
                  0.003844458
pData(gse25171)[, "Pi"]Treatment
                  0.010492799
```

Más detalles de esta opción las podemos ver en § 10.2.

5.6 Modelos lineales normales

[46] [46] [4, capítulo 3].

En este tema se asume un modelo estocástico para el vector de respuestas, $\boldsymbol{Y} = (Y_1, \ldots, Y_n)^T$. Asumiremos que sigue una distribución normal multivariante, $\boldsymbol{Y} \sim N_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$, con vector de medias $\boldsymbol{\mu}$ y matriz de covarianzas $\boldsymbol{\Sigma}$.

5.6.1 Modelo y verosimilitud

El modelo lineal normal asume que $\boldsymbol{Y} \sim N_n(\boldsymbol{\mu}, \sigma^2 \boldsymbol{I}_n)$, siendo $\mu_i = \boldsymbol{x}_i^T \boldsymbol{\beta}$ de donde la función de verosimilitud es

$$L(\boldsymbol{\beta}, \sigma^2) = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma^n} \exp\{-\frac{1}{\sigma^2}(\mathbf{y} - \boldsymbol{\mu})^T(\mathbf{y} - \boldsymbol{\mu})\} = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma^n} \exp\{-\frac{1}{\sigma^2} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^T \boldsymbol{\beta})^2\}. \quad (5.43)$$

La función de logverosimilitud es

$$\ell(\boldsymbol{\beta}, \sigma^2) = -\frac{n}{2} \ln(2\pi) - n \ln(\sigma) - \frac{1}{\sigma^2} \sum_{i=1}^{n} (y_i - \boldsymbol{x}_i^T \boldsymbol{\beta})^2. \quad (5.44)$$

En § 5.4 hemos visto que $\hat{\boldsymbol{\beta}} = (\boldsymbol{X}^T \boldsymbol{X})^{-1} \boldsymbol{X}^T \boldsymbol{Y}$, $\hat{\boldsymbol{\mu}} = \boldsymbol{H}\boldsymbol{Y}$ y $\hat{\boldsymbol{e}} = (\boldsymbol{I} - \boldsymbol{H})\boldsymbol{Y}$. En resumen, los estimadores mínimo cuadráticos de los coeficientes, las estimaciones de las medias o predicciones y los residuos son transformaciones lineales del vector (aleatorio) de observaciones. Y por ello todos ellos tienen distribución normal multivariante. Sin embargo, sus matrices de varianzas no son proporcionales a la matriz identidad. Por tanto, los distintos estimadores de los coeficientes no son independientes entre sí. Lo mismo podemos decir de las estimadores de las medias. Los estimadores de las distintas medias no son independientes. Tampoco son independientes entre sí los distintos residuos.

5.6.2 Formas cuadráticas con matrices de proyección

El siguiente resultado fue originalmente probado en [26].

Proposición 5.5. *Si* $\boldsymbol{Y} \sim N_n(\boldsymbol{\mu}, \sigma^2 \boldsymbol{I}_n)$ *y* $\boldsymbol{P}$ *es simétrica tenemos que* $\frac{1}{\sigma^2}(\boldsymbol{Y}-\boldsymbol{\mu})^T \boldsymbol{P}(\boldsymbol{Y}-\boldsymbol{\mu}) \sim \chi^2_r$ *si y solamente si* $\boldsymbol{P}$ *es una matriz de proyección de rango* r*.*

5.6.3 Constrastes para modelos lineales normales

Anova de una vía

El modelo es

$$Y_{ij} = \beta_0 + \beta_i + \epsilon_{ij}$$

siendo $\epsilon_{ij} \sim N(0, \sigma^2)$ e independientes entre sí. Asumimos la restricción $\beta_1 = 0$. Pretendemos contrastar: $H_0 : \mu_1 = \cdots = \mu_I$ con $(\mu_i = \beta_0 + \beta_i = E[Y_{ij}])$ frente a la alternativa de que al menos dos medias sean diferentes. Este contraste sería equivalente a $H_0 : \beta_1 = \cdots = \beta_I = 0$.

La matriz de proyección del modelo con todos los predictores es una matriz diagonal en bloques donde el i-ésimo bloque de la diagonal es $\frac{1}{n_i}\mathbf{1}_{n_i}\mathbf{1}_{n_i}^T$. En notación habitual sería

$$\boldsymbol{P_X} = diag(\frac{1}{n_1}\mathbf{1}_{n_1}\mathbf{1}_{n_1}^T, \ldots, \frac{1}{n_I}\mathbf{1}_{n_I}\mathbf{1}_{n_I}^T).$$

Si consideramos el modelo solamente con la constante vimos que la matriz de proyección era $\boldsymbol{P}_0 = \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T$. Tenemos la siguiente descomposición de la matriz identidad en matrices de proyección

$$\boldsymbol{I}_n = \boldsymbol{P}_0 + (\boldsymbol{P_X} - \boldsymbol{P}_0) + (\boldsymbol{I}_n - \boldsymbol{P_X}).$$

Estas matrices tienen rangos 1, $I-1$ y $n-I$. Además las sumas de cuadrados que se obtienen son

$$\boldsymbol{Y}^T(\boldsymbol{P_X} - \boldsymbol{P}_0)\boldsymbol{Y} = \sum_{i=1}^{I} n_i(\bar{Y}_{i\cdot} - \bar{Y}_{\cdot\cdot})^2,$$

$$\boldsymbol{Y}^T(\boldsymbol{I}_n - \boldsymbol{P_X})\boldsymbol{Y} = \sum_{i=1}^{I}\sum_{j=1}^{n_i}(Y_{ij} - \bar{Y}_{i\cdot})^2.$$

Utilizando el teorema de Cochran se sigue

$$\frac{1}{\sigma^2}\sum_{i=1}^{I} n_i(\bar{Y}_{i\cdot} - \bar{Y}_{\cdot\cdot})^2 \sim \chi^2_{I-1,\lambda}$$

siendo $\lambda = \frac{1}{\sigma^2}\boldsymbol{\mu}^T(\boldsymbol{P_X} - \boldsymbol{P}_0)\boldsymbol{\mu}$. Además,

$$\frac{1}{\sigma^2}\sum_{i=1}^{I}\sum_{j=1}^{n_i}(\bar{Y}_{ij} - \bar{Y}_{i\cdot})^2 \sim \chi^2_{n-I},$$

y las dos formas cuadráticas son independientes entre sí. Para los cuadrados residuales el parámetro de no centralidad es nulo puesto que

$\boldsymbol{\mu}^T(\boldsymbol{I}_n - \boldsymbol{P_X})\boldsymbol{\mu} = \boldsymbol{\mu}^T(\boldsymbol{\mu} - \boldsymbol{P_X}\boldsymbol{\mu}) = 0$. Tenemos pues que el estadístico para contrastar la hipótesis nula sigue la siguiente distribución

$$F = \frac{\sum_{i=1}^{I} n_i(\bar{Y}_{i\cdot} - \bar{Y}_{\cdot\cdot})^2/(I-1)}{\sum_{i=1}^{I}\sum_{j=1}^{n_i}(Y_{ij} - \bar{Y}_{i\cdot})^2/(n-I)} \sim F_{I-1,n-I,\lambda}.$$

Se comprueba fácilmente que

$$\boldsymbol{\mu}^T\boldsymbol{P_X}\boldsymbol{\mu} = \sum_{i=1}^{I} n_i\mu_i^2,$$

$$\boldsymbol{\mu}^T\boldsymbol{P}_0\boldsymbol{\mu} = n\bar{\mu}^2$$

con $\bar{\mu} = \sum_{i=1}^{I} \frac{n_i}{n}\mu_i$. El parámetro de no centralidad viene dado por

$$\lambda = \frac{1}{\sigma^2}\sum_{i=1}^{I} n_i(\mu_i - \bar{\mu})^2.$$

Bajo la hipótesis nula todas las medias son iguales tenemos que $\lambda = 0$ y $F \sim F_{I-1,n-I}$. Si observamos la expresión del parámetro de no centralidad deducimos que la potencia del test será mayor cuando mayores sean los tamaños muestrales de los distintos grupos y cuando mayor sea la variabilidad entre las medias.

Modelos anidados

Tenemos dos modelos, M_0 y M_1, de modo que M_0 está anidado en M_1. Las matrices de proyección de cada modelo serán $\boldsymbol{P}_0$ y $\boldsymbol{P}_1$. Suponemos que tienen rangos r_0 y r_1. La descomposición que consideramos ahora es

$$\boldsymbol{I}_n = \boldsymbol{P}_0 + (\boldsymbol{P}_1 - \boldsymbol{P}_0) + (\boldsymbol{I} - \boldsymbol{P}_1)$$

y la descomposición ortogonal única será

$$\boldsymbol{y} = \boldsymbol{P}_0\boldsymbol{y} + (\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{y} + (\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{y}.$$

Los valores ajustados para los modelos M_0 y M_1 serían $\hat{\boldsymbol{\mu}}_0 = \boldsymbol{P}_0\boldsymbol{Y}$ y $\hat{\boldsymbol{\mu}}_1 = \boldsymbol{P}_1\boldsymbol{Y}$. La descomposición de las sumas de cuadrados sería

$$\boldsymbol{y}^T\boldsymbol{y} = \boldsymbol{y}^T\boldsymbol{P}_0\boldsymbol{y} + \boldsymbol{y}^T(\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{y} + \boldsymbol{y}^T(\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{y}. \tag{5.45}$$

Tenemos

$$\boldsymbol{y}^T(\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{y} = \boldsymbol{y}^T(\boldsymbol{I} - \boldsymbol{P}_1)^T(\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{y} = \sum_{i=1}^{n}(y_i - \hat{\mu}_{i1})^2,$$

que corresponden con la suma de cuadrados residual correspondiente al modelo M_1, $SS(Residual)_{M_1}$. De hecho,

$$\begin{aligned}\boldsymbol{y}^T(\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{y} = \\ \boldsymbol{y}^T(\boldsymbol{I} - \boldsymbol{P}_0)\boldsymbol{y} - \boldsymbol{y}^T(\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{y} = \\ \sum_{i=1}^{n}(y_i - \hat{\mu}_{i0})^2 - \sum_{i=1}^{n}(y_i - \hat{\mu}_{i1})^2 = \\ SS(Residual)_{M_0} - SS(Residual)_{M_1}.\end{aligned} \tag{5.46}$$

Pero $\boldsymbol{P}_1 - \boldsymbol{P}_0$ es una matriz de proyección por tanto

$$\boldsymbol{y}^T(\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{y} = \boldsymbol{y}^T(\boldsymbol{P}_1 - \boldsymbol{P}_0)^T(\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{y} = (\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_0)^T(\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_0).$$

En consecuencia,

$$SS(Residual)_{M_0} - SS(Residual)_{M_1} = (\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_0)^T(\hat{\boldsymbol{\mu}}_1 - \hat{\boldsymbol{\mu}}_0) = \sum_{i=1}^{n}(\hat{\mu}_{i1} - \hat{\mu}_{i0})^2 = SS(Residual)_{M_1|M_0}. \quad (5.47)$$

Además,

$$rank(\boldsymbol{I} - \boldsymbol{P}_1) = tr(\boldsymbol{I} - \boldsymbol{P}_1) = tr(\boldsymbol{I}) - tr(\boldsymbol{P}_1) = n - r_1,$$

y $\boldsymbol{P}_1$ tiene rango completo, $rank(\boldsymbol{P}_1) = r_1$. De un modo análogo tenemos que $rank(\boldsymbol{P}_1 - \boldsymbol{P}_0) = r_1 - r_0$. Asumiendo la hipótesis nula y por el teorema de Cochran se sigue que

$$\frac{SS(Residual)_{M_0} - SS(Residual)_{M_1}}{\sigma^2} \sim \chi^2_{r_1 - r_0, \lambda_1}$$

y que

$$\frac{SS(Residual)_{M_1}}{\sigma^2} \sim \chi^2_{n - r_1, \lambda_2}.$$

Además $\frac{SS(Residual)_{M_0} - SS(Residual)_{M_1}}{\sigma^2}$ y $\frac{SS(Residual)_{M_1}}{\sigma^2}$ son independientes. Bajo la hipótesis nula de que se verifica el modelo M_0 tenemos que los parámetros de no centralidad de las distribuciones ji-cuadrado no centrales serían

$$\lambda_1 = \frac{1}{\sigma^2}\boldsymbol{\mu}^T(\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{\mu} = 0$$

y

$$\lambda_2 = \frac{1}{\sigma^2}\boldsymbol{\mu}^T(\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{\mu} = 0$$

ya que $\boldsymbol{P}_1\boldsymbol{\mu} = \boldsymbol{P}_0\boldsymbol{\mu} = \boldsymbol{\mu}$. En resumen, si el modelo M_0 es cierto tenemos que

$$F = \frac{(SS(Residual)_{M_0} - SS(Residual)_{M_1})/(r_1 - r_0)}{(SS(Residual)_{M_1})/(n - r_1)} \sim F_{r_1 - r_0, n - r_1} \quad (5.48)$$

Claramente rechazamos la hipótesis nula de que el modelo M_0 es asumible corresponde con valores grandes del estadístico en ecuación 5.48.

Todos los efectos nulos

Consideremos un caso particular de lo considerado en la sección anterior. En concreto que todos los coeficientes correspondientes a todas las variables predictoras son iguales a cero. Por tanto, el modelo M_0 corresponde al modelo solamente con la constante. El modelo M_1 tiene todas las variables predictoras que estamos considerando. Vimos en el tema anterior que la matriz de proyección sobre el subespacio que genera M_0 es que $\boldsymbol{P}_0 = \frac{1}{n}\mathbf{1}_n\mathbf{1}_n^T = \frac{1}{n}\boldsymbol{J}_n$. También obtuvimos la descomposición del vector $\boldsymbol{y}$

$$\boldsymbol{y} = \boldsymbol{P}_0\boldsymbol{y} + (\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{y} + (\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{y},$$

y de ahí inmediatamente la descomposición de la suma de cuadrados. Si denotamos por $\hat{\mu}_i$ los valores ajustados con el modelo completo entonces la tabla de análisis de la varianza aparece en tabla 5.5.

Tabla 5.5: Tabla de análisis de la varianza para el modelo completo.

Source	Projection matrix	df	Sum of squares	Mean Square	F	p
Intercept	$\boldsymbol{P}_0 = {}^1n\mathbf{1}_n\mathbf{1}_n^T$	1	$\boldsymbol{y}^T\boldsymbol{P}_0\boldsymbol{y} = n\bar{y}^2$	$n\bar{y}^2$		
Regression	$(\boldsymbol{P}_1 - \boldsymbol{P}_0)$	$p-1$	$\boldsymbol{y}^T(\boldsymbol{P}_1 - \boldsymbol{P}_0)\boldsymbol{y} = \sum_i(\hat{\mu}_i - \bar{y})^2$	$\frac{\sum_i(\hat{\mu}_i-\bar{y})^2}{p-1}$		
Error	$\boldsymbol{I} - \boldsymbol{P}_1$	$n-p$	$\boldsymbol{y}^T(\boldsymbol{I} - \boldsymbol{P}_1)\boldsymbol{y} = \sum_i(y_i - \hat{\mu}_i)^2$	$\frac{\sum_i(y_i-\hat{\mu}_i)^2}{n-p}$		
Total	$\boldsymbol{I}$	n	$\sum_i y_i^2$			

Contrastes para los coeficientes

Supongamos que nos planteamos la hipótesis nula de que H_0 : $\beta_j = 0$ vs $H_1 : \beta_j \neq 0$, es decir, que un coeficiente individual es nulo. Tenemos dos modelos anidados, uno sería el modelo con todos los predictores y el otro el modelo en que eliminamos el j-ésimo predictor. Se verifica (lo damos sin prueba) que

$$F = \frac{(SS(Residual)_{M_0} - SS(Residual)_{M_1})/1}{SS(Residual)_{M_1}/(n-p)} = \frac{\hat{\beta}_j^2}{(SE_j)^2}, \quad (5.49)$$

siendo SE_j el error estándar de $\hat{\beta}_j$. Recordemos que $var(\boldsymbol{\beta}) = \sigma^2(\boldsymbol{X}^T\boldsymbol{X})^{-1}$. Por tanto, $SE_j = s\sqrt{u_{jj}}$ siendo u_{jj} el elemento en posición (j, j) de la matriz $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$. El estadístico en ecuación 5.49 sigue una distribución $F_{1,n-p}$. En la ecuación 5.49 tenemos el cuadrado de $\frac{\hat{\beta}_j}{SE_j}$ que sigue por lo tanto una distribución t de Student con $n-p$ grados de libertad,

$$\frac{\hat{\beta}_j}{SE_j} \sim t_{n-p}.$$

Comparando modelos

Volvemos a ajustar el modelo `fit4`.

```
fit4 = lm(expression~ time + Pi,data=df0)
```

Los contrastes para los coeficientes así como el contraste global de si todos los predictores podemos considerar que son nulos lo tenemos con un **`summary`** del ajuste que nos devuelve la función **`lm`**.

```
summary(fit4)
```

```
Call:
lm(formula = expression ~time + Pi, data = df0)

Residuals:
   Min 1Q Median 3Q Max
-0.9399 -0.4025 0.1085 0.2609 1.1446

Coefficients:
          Estimate Std. Error t value Pr(>|t|)
(Intercept) 8.62939 0.18738 46.053 < 2e-16
time -0.13307 0.01194 -11.148 2.79e-10
PiTreatment -1.32191 0.23045 -5.736 1.08e-05

(Intercept) ***
time ***
```

```
PiTreatment ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.5645 on 21 degrees of freedom
Multiple R^2: 0.8821, Adjusted R^2: 0.8709
F-statistic: 78.6 on 2 and 21 DF, p-value: 1.774e-10
```

En la salida anterior podemos ver el test global en que todas las variables predictoras son no significativas, esto es, que todas los coeficientes de todas las predictoras son nulos. Este test aparece en la última línea del resumen. Para cada variable podemos ver junto a su estimación, su error estándar, el correspondiente cociente con distribución t de Student y el p-valor.

5.6.4 Intervalos de confianza y de predicción

En esta sección nos ocupamos de obtener intervalos de confianza para los coeficientes, la media de la respuesta para unos predictores dados y el propio valor aleatorio de la respuesta.

Para construir estos intervalos vamos a necesitar un resultado previo.

Proposición 5.6. *Los estimadores de los coeficientes, $\boldsymbol{\beta}$, y los residuos, $\mathbf{e}$, son vectores aleatorios independientes.*

Corolario 5.1. *$\hat{\beta}_j$ es independiente de $s^2 = \frac{1}{n-p}\sum_{i=1}^{n} e_i^2$.*

Proposición 5.7. *Se tiene que*

$$\frac{n-p}{\sigma^2}S^2 = \frac{1}{\sigma^2}\sum_{i=1}^{n} e_i^2 \sim \chi^2_{n-p}.$$

Intervalo para un coeficiente

Consideramos la hipótesis nula $H_0 : \beta_j = \beta_{j0}$ frente a que $H_1 : \beta_j \neq \beta_{j0}$ y el estadístico

$$t = \frac{\hat{\beta}_j - \beta_{j0}}{SE_j}. \tag{5.50}$$

Utilizando el corolario 5.1 tenemos que el numerador y el denominador de 5.50 son independientes. El estadístico definido en ecuación 5.50 tiene una distribución t con $n-p$ grados de libertad utilizando la igualdad 5.49. El intervalo de confianza al nivel $1-\alpha$ vendría dado por los valores β_{j0} para los cuales no rechazamos la hipótesis nula $H_0 : \beta_j = \beta_{j0}$ y vendría dado por

$$\hat{\beta}_j \pm t_{n-p,1-\alpha/2} SE_j.$$

Ejemplo 5.18. *Veamos cómo obtenerlos utilizando la función genérica* `confint` *aplicada a un objeto de clase* `lm` *Los intervalos de confianza, con nivel de confianza $1-\alpha = .95$, vistos en esta sección los obtenemos con*

```
confint(fit4,conf.level=0.95)
```

```
                2.5 % 97.5 %
(Intercept) 8.2397130 9.0190666
time -0.1578931 -0.1082475
PiTreatment -1.8011547 -0.8426596
```

Intervalo de confianza para la media

Supongamos que tomamos unos predictores dados $\boldsymbol{x}_0$ (vector columna) y queremos el intervalo de confianza para la media condicionada $\mu_0 = E[Y|\boldsymbol{x}_0] = \boldsymbol{x}_0^T\boldsymbol{\beta}$. Otra vez lo construimos a partir de un t-test. La predicción correspondiente a los nuevos predictores sería $\hat{\mu}_0 = \boldsymbol{x}_0^T\hat{\boldsymbol{\beta}}$. Su varianza vendrá dada por

$$var(\hat{\mu}_0) = var(\boldsymbol{x}_0^T\hat{\boldsymbol{\beta}}) = \boldsymbol{x}_0^T var(\hat{\boldsymbol{\beta}})\boldsymbol{x}_0 = \sigma^2\boldsymbol{x}_0^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_0.$$

Pero $\boldsymbol{x}_0^T\hat{\boldsymbol{\beta}}$ tiene una distribución normal por ser función lineal de $\boldsymbol{Y}$. Por lo tanto,

$$Z = \frac{\boldsymbol{x}_0^T\hat{\boldsymbol{\beta}} - \boldsymbol{x}_0^T\boldsymbol{\beta}}{\sigma\sqrt{\boldsymbol{x}_0^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_0}} \sim N(0,1).$$

Estimamos σ con S y tenemos el cociente

$$t = \frac{\boldsymbol{x}_0^T\hat{\boldsymbol{\beta}} - \boldsymbol{x}_0^T\boldsymbol{\beta}}{S\sqrt{\boldsymbol{x}_0^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_0}} \sim t_{n-p}.$$

Este último resultado se sigue de la proposición 5.7 y de que $\hat{\boldsymbol{\beta}}$ y S^2 son independientes (proposición 5.6) por lo que $\boldsymbol{x}_0^T\hat{\boldsymbol{\beta}}$ y S también lo son. El intervalo de confianza para la media condicionada $\boldsymbol{x}_0^T\boldsymbol{\beta}$ al nivel $1-\alpha$ sería

$$\boldsymbol{x}_0^T\hat{\boldsymbol{\beta}} \pm t_{n-p,1-\alpha/2}s\sqrt{\boldsymbol{x}_0^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_0}.$$

Intervalo de predicción

Consideramos unos predictores $\boldsymbol{x}_0$ y pretendemos construir un intervalo que contenga a la variable Y condicionada a los predictores $\boldsymbol{x}_0$ con una confianza dada. En resumen, un intervalo de predicción para una observación futura. Obviamente el intervalo será mayor. Es más complejo estimar la observación que estimar la media de la observación de lo que nos hemos ocupado esencialmente hasta ahora. Nuestro modelo asumido es

$$Y = \boldsymbol{x}_0^T\boldsymbol{\beta} + \epsilon \ con \ \epsilon \sim N(0,\sigma^2).$$

Ese valor aleatorio (futuro) tendrá como predicción $\hat{\mu}_0 = \boldsymbol{x}_0^T\hat{\boldsymbol{\beta}}$, esto es, nuestra predicción es el valor ajustado para la media. Lo que no será igual es el intervalo que lo contenga. Tendremos un residuo (aleatorio) $e = Y - \hat{\mu}$ y se satisface

$$Y = \boldsymbol{x}_0^T\hat{\boldsymbol{\beta}} + e.$$

Obviamente la nueva observación Y será independiente de las previas observaciones $Y_1, \ldots, Y_n$ utilizadas en el ajuste previo para estimar $\hat{\boldsymbol{\beta}}$ y el actual $\hat{\mu}_0$. En consecuencia

$$var(e) = var(Y - \hat{\mu}_0) = var(Y) + var(\hat{\mu}_0) = \sigma^2(1 + \boldsymbol{x}_0^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_0).$$

Se tiene que

$$\frac{Y - \hat{\mu}_0}{\sigma\sqrt{1 + \boldsymbol{x}_0^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_0}} \sim N(0,1)$$

y que

$$\frac{Y - \hat{\mu}_0}{S\sqrt{1 + \boldsymbol{x}_0^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_0}} \sim t_{n-p}.$$

De un modo análogo invirtiendo los contrastes (de otro modo, consideramos los valores para los que no rechazamos la hipótesis nula) tenemos como intervalo de predicción para la futura observación Y con predictores $\boldsymbol{x}_0$ el siguiente

$$\boldsymbol{x}_0^T\hat{\boldsymbol{\beta}} \pm t_{n-p,1-\alpha/2}s\sqrt{1 + \boldsymbol{x}_0^T(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{x}_0}.$$

Ejemplos de intervalos para media y observación

Ejemplo 5.19. *Con los datos* `df0` *consideramos cómo obtener las predicciones y los intervalos de confianza para las medias y para las predicciones. Utilizamos los propios datos que se han utilizado para ajustar el modelo. Con la función genérica* **`predict`** *obtenemos las predicciones así como los intervalos de confianza para las medias de las predicciones (*`predict` *con la opción* `interval="confidence"`*) y los intervalos de confianza para las observaciones (*`predict` *con la opción* `interval"="prediction`*).*

Primero obtengamos los valores ajustados o predicciones.

```
head(predict(fit4))
```

```
GSM618324.CEL.gz GSM618325.CEL.gz
        7.307483 8.629390
GSM618326.CEL.gz GSM618327.CEL.gz
        7.174412 8.496319
GSM618328.CEL.gz GSM618329.CEL.gz
        6.509061 7.830968
```

Los intervalos de confianza para la media se obtienen con

```
head(predict(fit4,interval = "confidence"))
```

```
                  fit lwr upr
GSM618324.CEL.gz 7.307483 6.917806 7.697159
GSM618325.CEL.gz 8.629390 8.239713 9.019067
GSM618326.CEL.gz 7.174412 6.796373 7.552451
GSM618327.CEL.gz 8.496319 8.118280 8.874358
GSM618328.CEL.gz 6.509061 6.167409 6.850713
GSM618329.CEL.gz 7.830968 7.489316 8.172620
```

Los intervalos de confianza para las observaciones (en este caso sobre los propios datos) se obtienen con

```
head(predict(fit4,interval = "prediction"))
```

```
Warning in predict.lm(fit4, interval = "prediction"): predictions on
    ↪ current data refer to _future_ responses
```

```
                  fit lwr upr
GSM618324.CEL.gz 7.307483 6.070584 8.544381
GSM618325.CEL.gz 8.629390 7.392491 9.866288
GSM618326.CEL.gz 7.174412 5.941131 8.407694
GSM618327.CEL.gz 8.496319 7.263038 9.729601
GSM618328.CEL.gz 6.509061 5.286442 7.731679
GSM618329.CEL.gz 7.830968 6.608350 9.053586
```

Es interesante el comentario. Nos indica que se refieren a ***futuras observaciones*** *que pudieran tener los mismos predictores que estamos usando.*

5.6.5 Selección de variables

Habitualmente tenemos un **gran** conjunto de variables y pretendemos determinar un **buen** conjunto de variables. ¿En qué sentido bueno? En primer lugar, bueno en el sentido de sencillo. Lo que se conoce como parsimonioso. Con lo que esto supone de facilidad de interpretación. Menos variables lo hace más fácilmente interpretable por el experimentador. Tener pocas variables supone un modelo más simple, más fácil de entender. Es bueno tener pocas variables porque luego no tendremos que recoger esta información. Es bueno porque los estimadores de los parámetros que intervienen en el modelo son mucho menos variables.

En el tema § 5.6 hemos visto cómo comparar modelos que estaban anidados uno dentro de otro.

Sin embargo, es posiblemente más frecuente en la actualidad partir de un gran número de predictores y buscar un buen modelo que no necesariamente han de estar anidados.

Criterios

[47] Pretendemos comparar modelos no anidados. La idea es definir una medida de calidad del modelo y buscar entre los posibles aquel que la optimice (minimice o maximice atendiendo a cómo la definamos).

[47] Esta sección utiliza mucho material de las secciones 2.11 y 2.12 de [31].

Si proponemos medidas de calidad que atiendan fundamentalmente a la suma de cuadrados residual entonces siempre elegiríamos modelos con todas las variables ya que más variables supone una suma de cuadrados residual menor. Hay software que lo hace pero entonces limitan el número de variables a utilizar. Buscan el mejor modelo con tres variables o con cuatro variables o con cinco variables. Y así sucesivamente.

Otra opción, posiblemente más razonable que limitar el número de variables es utilizar **buenas** medidas de calidad del modelo que tengan en cuenta la calidad del ajuste y penalicen, de alguna forma, el número de variables y no tengamos **demasiadas** variables.

Una medida global puede ser AIC (Akaike Information Criterion) definido como

$$AIC = -2 \max logverosimilitud + 2p,$$

en donde la expresión de la logverosimilitud la tenemos en (5.44) y p corresponde con el total de parámetros del modelo que en nuestro caso corresponde con el número de variables predictoras más el término constante. Por su definición es claro que un **mejor** modelo corresponde con un menor valor de AIC. Pretendemos **minimizar** el valor de AIC. Para modelos lineales normales la AIC es igual a

$$AIC = n \ln(SS(Residual)/n) + 2p. \tag{5.51}$$

Otra opción es el criterio de información bayesiano[48] **BIC** definido, para modelos lineales normales, como

[48] Bayesian Information Criterium.

$$BIC = n \ln(SS(Residual)/n) + p \ln n. \tag{5.52}$$

El término que penaliza el número de variables es $p \ln n$. Obviamente estamos penalizando más que en AIC ya que $\ln n > 2$ para valores no muy pequeños de n.

La medida AIC atiende más a la predicción. Busca modelos que proporciones mejores predicciones. La medida BIC pide, para incorporar variables, una mayor evidencia de su importancia. Busca modelos más simples pero con una peor capacidad de proporcionar buenas predicciones.

La función `MASS:stepAIC()` nos proporciona ambas medidas.

```
extractAIC(fit4)
```

```
[1] 3.00000 -24.65309
```

El primer valor es el número de parámetros estimados, p, y el valor AIC aparece es el segundo valor. El valor de BIC lo tenemos con la misma función ajustando la penalización.

```
extractAIC(fit4,k=log(length(df0$expression)))
```

```
[1] 3.00000 -21.11893
```

Procedimientos de selección

La evaluación del efecto de añadir variables puede ser evaluado con las funciones `stats::add1()`, `stats::drop1()` y `stats::step()`. Empezamos evaluando el efecto de eliminar una variable cada vez con `stats::drop1()`.

```
drop1(fit4,test="F")
```

```
Single term deletions

Model:
expression ~time + Pi
      Df Sum of Sq RSS AIC F value
<none> 6.692 -24.6531
time 1 39.603 46.295 19.7674 124.287
Pi 1 10.485 17.176 -4.0288 32.904
        Pr(>F)
<none>
time 2.795e-10 ***
Pi 1.078e-05 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Tenemos AIC para el modelo original en la primera fila y luego podemos ver el valor en los modelos que resultan de eliminar cada variable. Tenemos las sumas de cuadrados para cada variable, la residual cuando eliminamos la variable y los correspondientes cocientes y su p-valor.

La función `stats::add1` tiene un objetivo inverso al anterior. Ahora hemos de indicar cuál es el modelo más simple y el modelo más complejo que pretendemos evaluar. En este caso (simplemente como ilustración) vamos a considerar como modelo más simple aquél que tiene solamente la constante y como más complejo el modelo con todas las variables.

```
fit.null = lm(expression ~1,data=df0)
add1(fit.null,fit4,test="F")
```

```
Single term additions

Model:
expression ~1
      Df Sum of Sq RSS AIC F value
<none> 56.779 22.6669
time 1 39.603 17.176 -4.0288 50.7257
Pi 1 10.485 46.295 19.7674 4.9825
        Pr(>F)
<none>
time 3.842e-07 ***
Pi 0.03611 *
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Lo visto simplemente nos muestra el efecto de añadir o eliminar un predictor. Es fácil imaginar la complejidad del problema. Si suponemos que tenemos p variables predictoras incluyendo la constante entonces el total de modelos a evaluar es 2^p y, si mantenemos la constante, 2^{p-1} modelos distintos. No son pocos.

En la selección `forward` empezamos habitualmente con el modelo con la constante y vamos añadiendo en cada paso la función que produce en decremento mayor en el AIC (o BIC). Y lo seguimos haciendo hasta que AIC no mejora.

En la selección `backward` empezamos con todas las variables en el modelo y vamos eliminando las que nos producen un decremento mayor de AIC. Paramos cuando no mejoramos eliminando una variable.

En la selección `both` en una iteración dada tendremos un modelo con unas predictoras incluidas. Se evalúa cuál es la mejor opción, incorporar o eliminar una variable y se realiza. Cuando no mejora AIC paramos.

La función stats::step o MASS::stepAIC nos permite realizar cada una de las tres opciones con el argumento `direction`.

Ejemplo 5.20. *Empezamos con la opción* `forward`.

```
pacman::p_load(MASS)
min.model = fit.null
max.model = fit4
step(min.model,direction="forward",
     scope=list(lower=min.model,upper=max.model))
```

```
Start: AIC=22.67
expression ~1

      Df Sum of Sq RSS AIC
+ time 1 39.603 17.176 -4.0288
+ Pi 1 10.485 46.295 19.7674
<none> 56.779 22.6669

Step: AIC=-4.03
expression ~time

      Df Sum of Sq RSS AIC
+ Pi 1 10.485 6.6915 -24.6531
<none> 17.1762 -4.0288

Step: AIC=-24.65
expression ~time + Pi
```

```
Call:
lm(formula = expression ~time + Pi, data = df0)

Coefficients:
(Intercept) time PiTreatment
    8.6294 -0.1331 -1.3219
```

Ahora realizamos una selección `backward` *empezando por el modelo más complejo que pretendemos evaluar.*

```
step(max.model,direction="backward",
     scope=list(lower=min.model,upper=max.model))
```

```
Start: AIC=-24.65
expression ~time + Pi

      Df Sum of Sq RSS AIC
<none> 6.692 -24.6531
- Pi 1 10.485 17.176 -4.0288
- time 1 39.603 46.295 19.7674
```

```
Call:
lm(formula = expression ~time + Pi, data = df0)

Coefficients:
(Intercept) time PiTreatment
    8.6294 -0.1331 -1.3219
```

Finalmente consideramos la opción `both`.

```
step(max.model,direction="both",
     scope=list(lower=min.model,upper=max.model))
```

```
Start: AIC=-24.65
expression ~time + Pi

      Df Sum of Sq RSS AIC
<none> 6.692 -24.6531
- Pi 1 10.485 17.176 -4.0288
- time 1 39.603 46.295 19.7674
```

```
Call:
lm(formula = expression ~time + Pi, data = df0)

Coefficients:
(Intercept) time PiTreatment
    8.6294 -0.1331 -1.3219
```

En este ejemplo obtenemos el mismo modelo utilizando las tres opciones de selección.

5.6.6 Efectos

En esta sección presentamos cómo ilustrar los efectos de cada una de las variables predictoras sobre la variable respuesta. La idea es sencilla. En el eje de abscisas recogemos la variable predictora de interés y fijamos todas las demás variables predictoras a su valor medio. Notemos que esto tiene sentido para las variables binarias y para las variables dummy con las cuales representamos las variables categóricas y ordinales. En todos estos casos tenemos variables con valores cero y uno y el valor medio simplemente indica la proporción de unos.

Elegimos para ilustrar el modelo `fit4`. Vamos a ilustrarlo con dos paquetes [40, effects] y [60, ggeffects]. Empezamos con [40, effects].

```
pacman::p_load(effects)
```

```
timeEf = predictorEffect("time",fit4)
```

Podemos ver las predicciones.

```
timeEf
```

```
 time predictor effect

 time effect
time
      0 0.49 0.98 1.47 1.96
7.968436 7.903232 7.838027 7.772823 7.707618
   2.45 2.94 3.43 3.92 4.41
7.642414 7.577210 7.512005 7.446801 7.381596
    4.9 5.39 5.88 6.37 6.86
7.316392 7.251187 7.185983 7.120778 7.055574
   7.35 7.84 8.33 8.82 9.31
6.990369 6.925165 6.859961 6.794756 6.729552
    9.8 10.3 10.8 11.3 11.8
6.664347 6.597812 6.531277 6.464742 6.398207
   12.2 12.7 13.2 13.7 14.2
6.344979 6.278443 6.211908 6.145373 6.078838
   14.7 15.2 15.7 16.2 16.7
6.012303 5.945768 5.879232 5.812697 5.746162
   17.1 17.6 18.1 18.6 19.1
5.692934 5.626399 5.559864 5.493329 5.426793
   19.6 20.1 20.6 21.1 21.6
5.360258 5.293723 5.227188 5.160653 5.094118
     22 22.5 23 23.5 24
5.040890 4.974354 4.907819 4.841284 4.774749
```

Está definido un método para la función genérica `plot()` cuyo resultado lo mostramos en la figura 5.18(a).

```
plot(timeEf)
```

El siguiente código muestra las predicciones cuando nos fijamos en distintas variables predictoras. Podemos ver los correspondientes dibujos en la figura 5.18. En cada dibujo las variables no consideradas toman el valor igual a su media.

```
timeEf = predictorEffect("time",fit4)
png(paste0(dirTamiFigures,"timeEf.png"))
plot(timeEf)
dev.off()
PiEf = predictorEffect("Pi",fit4)
png(paste0(dirTamiFigures,"PiEf.png"))
plot(PiEf)
dev.off()
```

Repetimos lo anterior con [60, ggeffects] (ver [61]).

```
pacman::p_load(ggeffects)
timeGGEf = ggpredict(fit4, terms = "time [all]")
```

Las predicciones y los valores en que se fijan las variables no consideradas los obtenemos con

```
timeGGEf
```

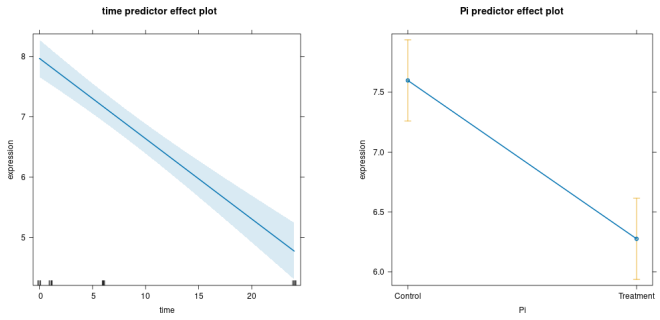


Figura 5.18: Efectos para las distintas variables predictoras: izquierda, **time**; derecha, **Pi**.

```
# Predicted values of expression

time | Predicted | 95% CI
------------------------------
   0 | 8.63 | [8.24, 9.02]
   1 | 8.50 | [8.12, 8.87]
   6 | 7.83 | [7.49, 8.17]
  24 | 5.44 | [4.91, 5.96]

Adjusted for:
* Pi = Control
```

Tenemos también definido un método **`plot()`** que nos proporciona el dibujo correspondiente aplicado al objeto de clase

```
class(timeGGEf)
```

```
[1] "ggeffects" "data.frame"
```

que acabamos de obtener. Lo aplicamos y obtenemos el gráfico correspondiente que se muestra en la figura 5.19(a).

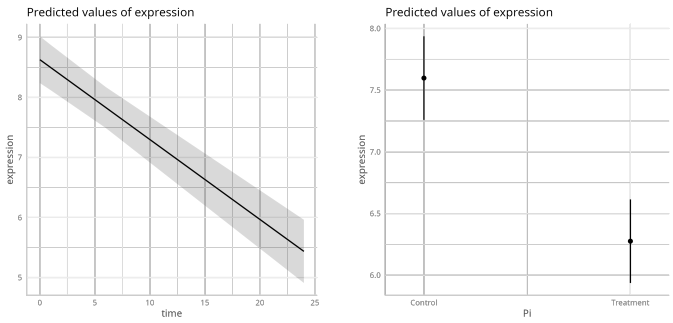
```
plot(timeGGEf)
```

Lo hacemos para cada de las variables predictoras con el siguiente código y los gráficos los mostramos en la figura 5.19.

```
pacman::p_load(ggplot2)
timeGGEf = ggpredict(fit4,terms = "time")
p = plot(timeGGEf)
ggsave(paste0(dirTamiFigures,"timeGGEf.png"),p)
PiGGEf = ggpredict(fit4,terms = "Pi")
p = plot(PiGGEf)
ggsave(paste0(dirTamiFigures,"PiGGEf.png"),p)
```

Hemos estimado las medias fijando todas las variables predictoras salvo una que hemos ido variando. Supongamos que hacemos variar dos variables predictoras y fijamos las demás a su media observada. Elegimos como variables que varían **`time`** y `Pi`. Tenemos las predicciones en la figura 5.20.

```
MHG = ggpredict(fit4,terms = c("time","Pi"))
p = plot(MHG)
ggsave(paste0(dirTamiFigures,"MHG.png"),p)
```



(a) (b)

Figura 5.19: Efectos para las distintas variables predictoras: (a) time, (b) Pi.

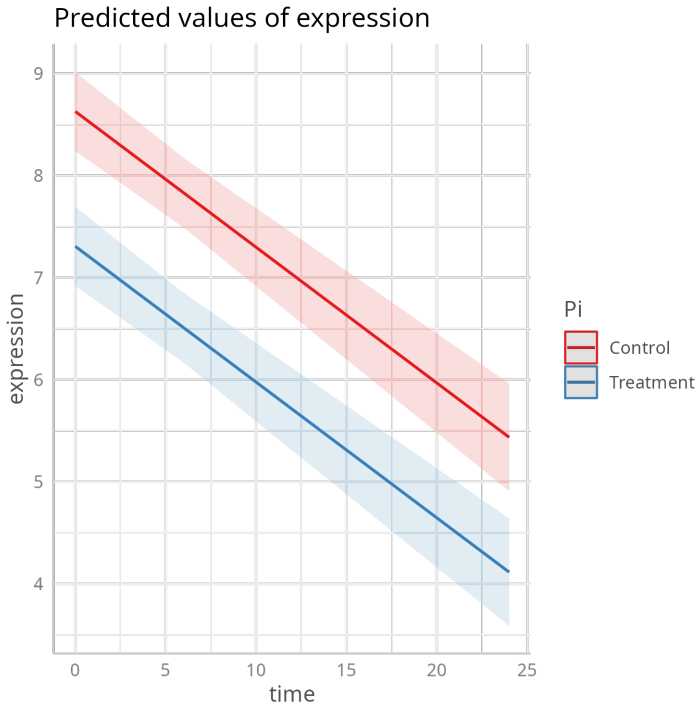


Figura 5.20: Efectos utilizando time y Pi.

Parte III

Modelos lineales generalizados

Capítulo 6

Datos categóricos

En este tema vemos un breve resumen de análisis estadístico de datos categóricos.[49]

Se entiende que tenemos un problema de datos categóricos cuando nuestra variable relevante, lo que hemos llamado variable respuesta, es una variable de tipo categórico: nominal u ordinal.

En este tema nos ocupamos de lo básico cuando trabajamos con este tipo de datos. Las distribuciones de probabilidad fundamentales en este contexto son la binomial, Poisson, binomial negativa y la hipergeométrica.

[49] Utilizamos fundamentalmente los tres primeros capítulos de [2]. El código de R se obtiene de [85].

6.1 Inferencia con la distribución binomial

La verosimilitud de una Bernoulli viene dada por

$$f(y|\pi) = \pi^y(1-\pi)^{1-y},$$

donde $y = 0, 1$. Si consideramos n observaciones $Y_1, \ldots, Y_n$ i.i.d. esto es una muestra aleatoria y los valores observados los denotamos por $y_1, \ldots, y_n$ entonces la función de verosimilitud viene dada por

$$L(\pi) = L(\pi; y_1, \ldots, y_n) = \prod_{i=1}^{n} \pi^{y_i}(1-\pi)^{1-y_i} = \pi^{\sum_{i=1}^n y_i}(1-\pi)^{n-\sum_{i=1}^n y_i},$$

donde $\boldsymbol{y} = (y_1, \ldots, y_n)$. La log-verosimilitud viene dada por

$$l(\pi) = \left(\sum_{i=1}^{n} y_i\right)\log\pi + \left(n - \sum_{i=1}^{n} y_i\right)\log(1-\pi).$$

Sin dificultad vemos que el estimador máximo verosímil es

$$\hat{\pi} = \sum_{i=1}^{n} \frac{y_i}{n}. \tag{6.1}$$

Es claro que $\boldsymbol{Y} = \sum_{i=1}^n Y_i \sim Bi(n, \pi)$.[1]

[1] En § 4.1 se incluye un comentario en donde se discute cómo distintos modelos probabilísticos producen verosimilitudes proporcionales.

6.1.1 Contrastes

Consideremos la hipótesis nula simple consistente en que la probabilidad de éxito, π, toma un valor dado, esto es, el contraste

$$\begin{aligned} H_0 : \pi &= \pi_0 \\ H_1 : \pi &\neq \pi_0. \end{aligned}$$

El estadístico score sería

$$Z_S = \frac{\hat{\pi} - \pi_0}{\sqrt{\pi_0(1-\pi_0)/n}}. \tag{6.2}$$

Bajo la hipótesis nula Z_S se distribuye aproximadamente como una normal estándar.

El estadístico de Wald (§ 4.4.2) viene dado por

$$Z_W = \frac{\hat{\pi} - \pi_0}{\sqrt{\hat{\pi}(1-\hat{\pi})/n}}, \tag{6.3}$$

y tiene aproximadamente una distribución normal estándar. Finalmente si consideramos el test del cociente de verosimilitudes (§ 4.4.1) adopta la expresión

$$-2\log\Lambda = 2\left(y\log\frac{y}{n\pi_0} + (n-y)\log\frac{n-y}{n-n\pi_0}\right), \tag{6.4}$$

donde Λ denota el cociente de verosimilitudes. La expresión dada en 6.4 no es más que

$$2\sum Observado \log\frac{Observado}{Esperado}.$$

Bajo H_0, esto es que la probabilidad de éxito es π_0, el test del cociente de verosimilitudes tiene una distribución asintótica que es una ji-cuadrado con un grado de libertad, χ^2_1.

6.1.2 Intervalos de confianza

A partir de los estadísticos anteriores se obtiene fácilmente el correspondiente intervalo de confianza, con nivel de confianza $1-\alpha$, como los valores π_0 para los cuales no rechazamos la hipótesis nula con un nivel de significación α. El intervalo de confianza de Wald para π viene dado por

$$\{\pi_0 : |z_W| < Z_{1-\alpha/2}\},$$

donde $Z_{1-\alpha/2}$ denota el percentil $1-\alpha/2$ de la normal estándar. El intervalo adopta la siguiente expresión

$$\hat{\pi} \pm Z_{1-\alpha/2}\sqrt{\frac{\hat{\pi}(1-\hat{\pi})}{n}}.$$

No es un buen intervalo cuando n es pequeño fundamentalmente para valores de π próximos a 0 o a 1.

Si utilizamos el estadístico score entonces los extremos los obtenemos resolviendo las ecuaciones

$$\frac{\hat{\pi} - \pi_0}{\sqrt{\pi_0(1-\pi_0)/n}} = \pm Z_{1-\alpha/2}.$$

La expresión final es bastante compleja pues tenemos una función cuadrática en π_0. Finalmente el intervalo de confianza del cociente de verosimilitudes no tiene una expresión sencilla y viene dado por

$$\{\pi_0 : -2\log\Lambda \leq \chi^2_1(\alpha)\}$$

Ejemplo 6.1 (Vegetarianos). *Preguntamos si se es o no vegetariano. Se considera como éxito ser vegetariano y como fracaso no serlo. Supongamos un tamaño muestral de $n = 55$ y ninguna persona se declara vegetariana, esto es, $y = 0$. Veamos cómo construir los intervalos de confianza de Wald, score y del cociente de verosimilitudes con R. Consideremos el intervalo de Wald para el parámetro de una binomial.*

```
pacman::p_load(Hmisc)
Hmisc::binconf(x = 0, n = 25, method = "asymptotic")
```

```
 PointEst Lower Upper
        0 0 0
```

Para el intervalo score lo podemos obtener mediante dos opciones. La primera es

```
res = prop.test(x = 0, n = 25, conf.level = 0.95, correct = FALSE)
res$conf.int
```

```
[1] 0.0000000 0.1331923
attr(,"conf.level")
[1] 0.95
```

La segunda opción sería

```
Hmisc::binconf(x = 0, n = 25, alpha = 0.05, method = "wilson")
```

```
 PointEst Lower Upper
        0 0 0.1331923
```

6.2 Inferencia para la multinomial

Los parámetros a estimar son $(\pi_1, \ldots, \pi_I)$ aunque $\pi_I = 1 - \sum_{i=1}^{I-1} \pi_i$. Tomamos una muestra de tamaño n y suponemos que tenemos y_j en la categoría j. La verosimilitud es proporcional a

$$\prod_{j=1}^{I} \pi_j^{y_j} \text{ donde } \pi_j \geq 0, \sum_j \pi_j = 1.$$

Los estimadores máximo verosímiles se obtienen fácilmente igualando las derivadas parciales a cero y son

$$\hat{\pi}_j = \frac{y_j}{n}.$$

6.2.1 Contrastes

Consideremos el siguiente contraste.

$$\begin{aligned} H_0 &: \pi_j = \pi_{j0} \text{ con } j = 1, \ldots, I \\ H_1 &: \text{No } H_0. \end{aligned}$$

Bajo H_0 *esperamos* observar de la categoría j, $\mu_j = n\pi_j$ (frecuencia esperada) pero observamos y_j (frecuencia observada). El estadístico de ji-cuadrado de Karl Pearson es el siguiente

$$X^2 = \sum_j \frac{(y_j - \mu_j)^2}{\mu_j} = \sum_j \frac{(y_j - n\pi_{j0})^2}{n\pi_{j0}}.$$

Bajo H_0, y para grandes muestras, se tiene la siguiente distribución asintótica (sin prueba)

$$X^2 \sim \chi^2_{I-1}.$$

El p-valor sería

$$p = P(X^2 \geq X_0^2),$$

donde X_0^2 es el valor observado de X^2.

Ejemplo 6.2 (Mendel)**.** *Cruzamos guisantes amarillos puros con guisantes verdes puros. El carácter dominante es el amarillo. De acuerdo con la teoría de Mendel* $3/4$ *tienen que ser amarillos y* $1/4$ *verdes. En uno de los experimentos de Mendel tenemos* $n = 8023$*, de los cuales* $n_1 = 6022$ *amarillos y* $n_2 = 2001$ *verdes. Aplicamos el test ji-cuadrado:*

```
stats::chisq.test(x = c(6022, 2001), p = c(0.75, 0.25))
```

```
        Chi-squared test for given probabilities

data: c(6022, 2001)
X-squared = 0.014999, df = 1, p-value =
0.9025
```

Fisher (y otros) vieron que tuvo demasiada suerte en su experimentación.

```
stats::chisq.test(x = c(6022, 2001), p = c(0.75, 0.25))
```

```
        Chi-squared test for given probabilities

data: c(6022, 2001)
X-squared = 0.014999, df = 1, p-value =
0.9025
```

Consideremos el test del cociente de verosimilitudes. Tenemos el contraste:

$$\begin{aligned} H_0 &: \pi_j = \pi_{j0} \text{ con } j = 1, \ldots, c \\ H_1 &: \text{No } H_0. \end{aligned}$$

El cociente de verosimilitudes viene dado por

$$\Lambda = \frac{\prod_j \pi_{j0}^{n_j}}{\prod_j (n_j/n)^{n_j}}$$

El estadístico del cociente de verosimilitudes es

$$G^2 = -2\log\Lambda = 2\sum_j y_j \log\frac{y_j}{n\pi_{j0}}.$$

Con n grande, bajo H_0, tenemos

$$G^2 \sim \chi^2_{I-1}.$$

Tabla 6.1: Aspirina y ataque cardíaco.

	Ataque fatal	Ataque no fatal	No ataque
Placebo	18	171	10845
Aspirina	5	99	10933

Tabla 6.2: Aspirina y ataque cardíaco. Consideramos si ha habido ataque frente a que no lo ha habido.

	Ataque	No ataque
Placebo	189	10845
Aspirina	104	10933

6.3 Probabilidad y tablas de contingencia

En esta sección nos ocupamos del estudio de la distribución conjunta de dos variables categóricas. Supondremos dos variables categóricas X e Y con I y J categorías respectivamente. Un individuo puede venir clasificado en una de $I \times J$ categorías.

6.3.1 Distribución conjunta y tabla de contingencia

Consideremos los datos en tabla 6.1. Se trata de estudiar el posible efecto preventivo del uso de la aspirina para prevenir ataques al corazón. Se realizó un estudio aleatorizado durante cinco años ([74]) de modo que se asignó al azar a cada persona en dos posibles grupos. En un grupo los participantes tomaron una aspirina cada día. En el otro tomaron un placebo. Los participantes no sabían si tomaban aspirina o placebo. Todos los participantes eran médicos. Obviamente lo primero que hacemos con los datos es construir una tabla en que aparecen las frecuencias conjuntas donde consideramos la variable X que nos indica si el individuo toma aspirina o placebo ($I = 2$) y la variable Y que nos indica si ha tenido ataque cardíaco fatal, un ataque no fatal o bien no ha tenido ninguno ($J = 3$). Esta tabla recibe el nombre de *tabla de contingencia* o *tabla de clasificación cruzada*. También consideraremos la situación simplificada en que agrupamos los valores de ataque fatal y no fatal de modo que la variable Y pasa a tener dos categorías.

Tenemos dos variables discretas. Su **distribución conjunta** viene dada por

$$\pi_{ij} = P(X = i, Y = j),$$

con $i = 1, \ldots, I$ y $j = 1, \ldots, J$. Las **distribuciones marginales** son, para la variable X,

$$\pi_{i+} = P(X = i) = \sum_{j=1}^{J} P(X = i, Y = j) = \sum_{j=1}^{J} \pi_{ij}$$

y para la variable Y

$$\pi_{+j} = P(Y = j) = \sum_{i=1}^{I} P(X = i, Y = j) = \sum_{i=1}^{I} \pi_{ij}$$

Tabla 6.3: Tabla de contingencia 2×2.

	Test positivo	Test negativo	Total
Enfermo	n_{11}	n_{11}	n_{1+}
No enfermo	n_{21}	n_{22}	n_{2+}
Total	n_{+1}	n_{+2}	n

Tabla 6.4: Tabla de frecuencias relativas sobre el total de la tabla.

$\hat{\pi}_{ij}$	Test positivo	Test negativo	Total
Enfermo	n_{11}/n	n_{11}/n	n_{1+}/n
No enfermo	n_{21}/n	n_{22}/n	n_{2+}/n
Total	n_{+1}/n	n_{+2}/n	1

Habitualmente una variable (por ejemplo, Y) es una variable respuesta y la otra (X) es explicativa o predictora. En esta situación no tiene sentido hablar de distribución conjunta. Es natural considerar la distribución condicionada de Y a X viene dada por

$$P(Y = j|X = i) = \pi_{j|i} = \frac{\pi_{ij}}{\pi_{i+}}$$

Se dice que las dos variables son **independientes** si

$$\pi_{ij} = \pi_{i+}\pi_{+j},$$

para todo i, j. En particular, la distribución condicionada es igual a la distribución marginal si las variables son independientes, esto es,

$$\pi_{j|i} = \pi_{+j} \text{ con } j = 1, \ldots, J.$$

Si X e Y son variables respuesta entonces hablamos de *independencia*. Si Y es respuesta y X explicativa hablamos de *homogeneidad*.

En la tabla 6.3 recogemos una tabla cruzada con los conteos en donde consideramos las frecuencias absolutas.

Si dividimos los conteos por el total de la tabla estamos **estimando** la distribución conjunta $\pi_{ij} = P(X = i, Y = j)$, $\hat{\pi}_{ij}$. Lo tenemos en la tabla 6.4.

En la tabla 6.5 tenemos las proporciones por fila que estiman la probabilidad condicionada $P(Y = j|X = i)$.

En la tabla 6.6 tenemos las proporciones por columna. Estamos estimando $P(X = i|Y = j)$.

Si volvemos a revisar la tabla 6.5 tenemos los siguientes conceptos de uso habitual en Epidemiología.

Sensibilidad Proporción de enfermos correctamente diagnósticados.

$$\pi_{1|1} = P(Y = 1|X = 1).$$

Especificidad Proporción de no enfermos correctamente diagnósticados.

$$\pi_{2|2} = P(Y = 2|X = 2).$$

Tabla 6.5: Proporciones por fila.

	Test positivo	Test negativo	Total
Enfermo	n_{11}/n_{1+}	n_{12}/n_{1+}	1
No enfermo	n_{21}/n_{2+}	n_{22}/n_{2+}	1

Tabla 6.6: Proporciones por columna.

$\hat{\pi}_{j\|i}$	Test positivo	Test negativo
Enfermo	n_{11}/n_{+1}	n_{12}/n_{+2}
No enfermo	n_{21}/n_{+1}	n_{22}/n_{+2}
Total	1	1

	1	2
1	$\pi_{1\|1}$	$\pi_{2\|1}$
2	$\pi_{1\|2}$	$\pi_{2\|2}$

6.4 Comparación de dos proporciones

Muchos estudios se diseñan para comparar grupos basándonos en una respuesta binaria, Y. Con dos grupos tenemos una tabla de contingencia 2×2.

$$\begin{aligned} \pi_{1|i} &= \pi_i \\ \pi_{2|i} &= 1 - \pi_{1|i} = 1 - \pi_i \end{aligned}$$

Queremos comparar π_1 con π_2.

¿Cómo comparamos? Podemos estudiar la diferencia de las proporciones $\pi_1 - \pi_2$. O bien el riesgo relativo: $\frac{\pi_1}{\pi_2}$. O bien el cociente de odds (**odds ratio**)

$$\theta = \frac{\pi_1/(1-\pi_1)}{\pi_2/(1-\pi_2)}. \tag{6.5}$$

Si π es la probabilidad de éxito entonces los odds se definen como

$$\Omega = \frac{\pi}{1-\pi}.$$

Equivalentemente

$$\pi = \frac{\Omega}{\Omega + 1}.$$

En una tabla 2×2 tenemos los odds en la fila i

$$\Omega_i = \frac{\pi_i}{1-\pi_i}.$$

El cociente de los odds de las dos filas será el **odds ratio**

$$\theta = \frac{\pi_1/(1-\pi_1)}{\pi_2/(1-\pi_2)}.$$

Se tiene fácilmente que

$$\theta = \frac{\pi_{11}\pi_{22}}{\pi_{12}\pi_{21}}.$$

Por ello también se le llama el **cociente de los productos cruzados**. Los siguientes puntos son de interés sobre el odds ratio.

	Éxito	**Fracaso**
Grupo 1	$\pi_{1\|1}$	$\pi_{2\|1}$
Grupo 2	$\pi_{1\|2}$	$\pi_{2\|2}$

	Éxito	**Fracaso**
Grupo 1	π_1	$1-\pi_1$
Grupo 2	π_2	$1-\pi_2$

- Puede ser cualquier valor positivo.
- $\theta = 1$ significa que no hay asociación entre X e Y.
- Valores de θ alejados de 1 indican una asociación mayor.
- Se suele trabajar con $\log\theta$ pues entonces el valor que tenemos es simétrico respecto a cero.
- El odds ratio no cambia cuando intercambiamos filas y columnas.

Ejemplo 6.3. *Leemos los datos de la tabla 2.1 en [1].*

```
x = c(104, 189)
n = c(11037, 11034)
prop.test(x, n)
```

```
        2-sample test for equality of proportions
        with continuity correction

data: x out of n
X-squared = 24.429, df = 1, p-value =
7.71e-07
alternative hypothesis: two.sided
95 percent confidence interval:
 -0.010814914 -0.004597134
sample estimates:
   prop 1 prop 2
0.00942285 0.01712887
```

```
asp.ataque = prop.test(x, n)
attributes(asp.ataque)
```

```
$names
[1] "statistic" "parameter" "p.value"
[4] "estimate" "null.value" "conf.int"
[7] "alternative" "method" "data.name"

$class
[1] "htest"
```

```
prop.test(x, n, alt = "less")
```

```
        2-sample test for equality of proportions
        with continuity correction

data: x out of n
X-squared = 24.429, df = 1, p-value =
3.855e-07
alternative hypothesis: less
95 percent confidence interval:
 -1.000000000 -0.005082393
sample estimates:
   prop 1 prop 2
0.00942285 0.01712887
```

```
asp.ataque$estimate
```

```
   prop 1 prop 2
0.00942285 0.01712887
```

La diferencia de proporciones viene dada por

```
asp.ataque$estimate[2] − asp.ataque$estimate[1]
```

```
    prop 2
0.007706024
```

El riesgo relativo lo calculamos con

```
asp.ataque$estimate[2]/asp.ataque$estimate[1]
```

```
  prop 2
1.817802
```

Finalmente el odds ratio sería

```
x[2] ∗ (n[1] − x[1])/(x[1] ∗ (n[2] − x[2]))
```

```
[1] 1.832054
```

6.5 Inferencia en tablas de contingencia

6.5.1 Intervalos de confianza para parámetros de asociación

Odds ratio

Consideremos el intervalo para los odds ratio. El estimador del odds ratio es

$$\hat{\theta} = \frac{n_{11}n_{22}}{n_{12}n_{21}}$$

El estimador puede ser 0, infinito o no estar definido (∞) dependiendo de los conteos. Por ello no existe ni la media ni la varianza de $\hat{\theta}$ ni de $\log\hat{\theta}$. Una posibilidad es trabajar con

$$\tilde{\theta} = \frac{(n_{11}+0.5)(n_{22}+0.5)}{(n_{12}+0.5)(n_{21}+0.5)}$$

y con $\log\tilde{\theta}$. Un estimación del error estándar de $\hat{\theta}$ es

$$\hat{\sigma}(\log\hat{\theta}) = \left(\frac{1}{n_{11}} + \frac{1}{n_{12}} + \frac{1}{n_{21}} + \frac{1}{n_{22}}\right)^{1/2}$$

El intervalo de confianza de Wald sería:

$$\log\hat{\theta} \pm z_{\alpha/2}\hat{\sigma}(\log\hat{\theta})$$

Tomando las exponenciales en los extremos tenemos el correspondiente intervalo para $\log\theta$. El test es algo conservador (la probabilidad de cubrimiento es algo mayor que el nivel nominal).

Ejemplo 6.4 (Aspirina e infarto)**.** *Estudio sueco sobre el uso de la aspirina y el infarto de miocardio.*

	Infarto de miocardio		
	Si	*No*	*Total*
Placebo	*28*	*656*	*684*
Aspirina	*18*	*658*	*656*

Veamos el intervalo de confianza para el odds ratio.

```
Drug ←c("Placebo", "Aspirin")
Infarction ←c("yes", "no")
table.3.1 ←expand.grid(drug = Drug, infarction = Infarction)
datos ←c(28, 18, 656, 658)
table.3.1 ←cbind(table.3.1, count = datos)
```

El código siguiente convierte el data frame en una matriz. La siguiente función calcula todos los intervalos de confianza.

```
## Gives two-sided Wald CI's for odds ratio, difference in proportions
##and relative risk.
## Table is a 2x2 table of counts with rows giving the treatment
    ↪ populations
## aff.response is a string like "c(1,1)" giving the cell of the
    ↪ beneficial
## response and the treatment category alpha is significance level
Wald.ci←function(Table, aff.response, alpha=.05){
    pow←function(x, a=−1) x^a
    z.alpha←qnorm(1−alpha/2)
    if(is.character(aff.response))
        where←eval(parse(text=aff.response))
    else where←aff.response
    Next←as.numeric(where==1) + 1
    ## OR
    odds.ratio←
        Table[where[1],where[2]]∗Table[Next[1],Next[2]]/
        (Table[where[1],Next[2]]∗Table[Next[1],where[2]])
    se.OR←sqrt(sum(pow(Table)))
    ci.OR←exp(log(odds.ratio) + c(−1,1)∗z.alpha∗se.OR)
    ## difference of proportions
    p1←Table[where[1],where[2]]/(n1←Table[where[1],Next[2]] +
                                  Table[where[1],where[2]])
    p2←Table[Next[1],where[2]]/(n2←Table[Next[1],where[2]]+
                                  Table[Next[1],Next[2]])
    se.diff←sqrt(p1∗(1−p1)/n1 + p2∗(1−p2)/n2)
    ci.diff←(p1−p2) + c(−1,1)∗z.alpha∗se.diff
    ## relative risk
    RR←p1/p2
    se.RR←sqrt((1−p1)/(p1∗n1) + (1−p2)/(p2∗n2))
    ci.RR←exp(log(RR) + c(−1,1)∗z.alpha∗se.RR)
    list(OR=list(odds.ratio=odds.ratio, CI=ci.OR),
         proportion.difference=list(diff=p1−p2,
                                    CI=ci.diff),
         relative.risk=list(relative.risk=RR,CI=ci.RR))
}
```

Utilizamos la función que acabamos de definir. Observemos que nos devuelve las estimaciones y los intervalos de confianza para los odds ratio, la diferencia de proporciones y el riesgo relativo.

```
Wald.ci(temp, c(1, 1))
```

Diferencia de proporciones

Veamos el intervalo de confianza para la diferencia de proporciones. Suponemos que tenemos muestras de binomiales independientes.

En grupo i tenemos $Y_i \sim Bi(n_i, \pi_i)$. Tenemos

$$\hat{\pi}_i = Y_i/n_i$$

$$E(\hat{\pi}_1 - \hat{\pi}_2) = \pi_1 - \pi_2$$

y el error estándar es

$$\sigma(\hat{\pi}_1 - \hat{\pi}_2) = \left[\frac{\pi_1(1-\pi_1)}{n_1} + \frac{\pi_2(1-\pi_2)}{n_2}\right]^{1/2}$$

Estimamos sustituyendo π_i por $\hat{\pi}_i$. El intervalo de confianza de Wald sería:

$$\hat{\pi}_1 - \hat{\pi}_2 \pm z_{\alpha/2}\hat{\sigma}(\hat{\pi}_1 - \hat{\pi}_2)$$

Usualmente la probabilidad de cubrimiento es menor que el coeficiente de confianza nominal. Especialmente para valores de π_1 y π_2 próximos a 0 o 1.

Intervalo de confianza para el riesgo relativo

El riesgo relativo muestral viene dado por

$$r = \frac{\hat{\pi}_1}{\hat{\pi}_2}$$

Hay una convergencia a la normalidad más rápida trabajando en la escala logarítmica. El error estándar asintótico de $\log r$ es

$$\sigma(\log r) = \left(\frac{1-\pi_1}{n_1\pi_1} + \frac{1-\pi_2}{n_2\pi_2}\right)^{1/2}.$$

Es algo conservador (probabilidad de cubrimiento mayor que el nivel de confianza nominal). El intervalo de confianza de Wald para $\log \pi_1/\pi_2$ es

$$\log r \pm z_{\alpha/2}\hat{\sigma}$$

6.5.2 Contraste de independencia en tablas de doble entrada

Nos planteamos el contraste de:

$$\begin{aligned} H_0 : \pi_{ij} &= \pi_{i+}\pi_{+j} \ \forall\ i,j \\ H_0 : \pi_{ij} &\neq \pi_{i+}\pi_{+j} \text{ para algún } i,j \end{aligned}$$

Los tests que vamos a considerar se pueden aplicar tanto para muestreo multinomial (con $I \times J$ categorías) como para muestreo multinomial independiente (para las distintas filas). En el primer caso contrastamos independencia y en el segundo homogeneidad. Es el test clásico propuesto por K. Pearson. Bajo H_0,

$$En_{ij} = \mu_{ij} = n\pi_{i+}\pi_{+j}.$$

Los MLE son

$$\hat{\mu}_{ij} = n\hat{\pi}_{i+}\hat{\pi}_{+j} = n\frac{n_{i+}}{n}\frac{n_{+j}}{n} = \frac{n_{i+}n_{+j}}{n}$$

Tabla 6.7: Educación y sentimiento religioso.

	Creencias religiosas			
Educación	Fundamentalista	Moderado	Liberal	Total
Menos que secundaria	178	138	108	424
Secundaria	570	648	442	1660
Graduado	138	252	252	642
Total	886	1038	802	2726

Se utiliza el estadístico

$$X^2 = \sum_i \sum_j \frac{(n_{ij} - \hat{\mu}_{ij})^2}{\hat{\mu}_{ij}}. \tag{6.6}$$

Bajo H_0,

$$X^2 \sim \chi^2((I-1)(J-1)). \tag{6.7}$$

El test score produce el estadístico X^2. El test del cociente de verosimilitud produce el test

$$G^2 = 2\sum_i \sum_j n_{ij} \log \frac{n_{ij}}{\hat{\mu}_{ij}}, \tag{6.8}$$

con $\hat{\mu}_{ij} = \frac{n_{i+}n_{+j}}{n}$. Bajo H_0,

$$G^2 \sim \chi^2((I-1)(J-1)). \tag{6.9}$$

Se rechaza para valores grandes de X^2 o G^2. La convergencia a la distribución ji-cuadrado es más rápida para X^2 que para G^2. La aproximación para G^2 es pobre si $n/IJ < 5$. La aproximación para X^2 puede ser razonablemente buena si las frecuencias esperadas son mayores que 1 y la mayor parte son mayores que 5. Si nos los podemos usar hay que utilizar métodos para muestras pequeñas.

Ejemplo 6.5. *Consideremos la tabla de contingencia 6.7.*
Vamos a aplicar los tests X^2 y G^2.

```
religion.counts<-c(178,138,108,570,648,442,138,252,252)
table.3.2=cbind(expand.grid(list(Religious.Beliefs=c("Fund", "Mod", "Lib"),
Highest.Degree=c("<HS","HS or JH", "Bachelor or Grad"))),count=
    ↪ religion.counts)
table.3.2.array=tapply(table.3.2$count,table.3.2[,1:2], sum)
(res=chisq.test(table.3.2.array))
```

```
        Pearson's Chi-squared test

data:  table.3.2.array
X-squared = 69.157, df = 4, p-value =
3.42e-14
```

Los conteos esperados los obtenemos con

```
res$expected
```

```
                Highest.Degree
Religious.Beliefs <HS HS or JH
            Fund 137.8078 539.5304
            Mod 161.4497 632.0910
            Lib 124.7425 488.3786
```

```
                Highest.Degree
Religious.Beliefs Bachelor or Grad
           Fund 208.6618
           Mod 244.4593
           Lib 188.8789
```

Los tests y frecuencias esperadas pueden obtenerse con el paquete vcd.

```
library(vcd)
assocstats(table.3.2.array)
```

```
                    X^2 df P(> X^2)
Likelihood Ratio 69.812 4 2.4869e-14
Pearson 69.157 4 3.4195e-14

Phi-Coefficient : NA
Contingency Coeff.: 0.157
Cramer's V        : 0.113
```

La función chisq.test lleva test de Montecarlo.

```
chisq.test(table.3.2.array, sim=T, B=2000)
```

```

        Pearson's Chi-squared test with simulated
         p-value (based on 2000 replicates)

data:  table.3.2.array
X-squared = 69.157, df = NA, p-value =
0.0004998
```

El test del cociente de verosimilitud G^2 se obtiene como

```
fit.glm = glm(count~Religious.Beliefs+Highest.Degree, data=table.3.2,
              family=poisson)
fit.glm$deviance
```

```
[1] 69.81162
```

```
temp<-predict(fit.glm,type="response")
matrix(temp, nc=3, byrow=T, dimnames=list(c("<HS","HS or JH", "Bachelor
    ↪ or
Grad"),c("Fund", "Mod", "Lib")))
```

```
                     Fund Mod Lib
<HS 137.8078 161.4497 124.7425
HS or JH 539.5304 632.0910 488.3786
Bachelor or\nGrad 208.6618 244.4593 188.8789
```

6.5.3 Más allá del test ji-cuadrado

¿Acaba la vida con el p-valor? Ya sabemos que el test es significativo. Ya tenemos un p-valor maravillosamente pequeño. ¿Y ahora qué? Quizás analizar los residuos. Vamos a comparar las frecuencias observadas con las esperadas. Notemos que, para muestreo de Poisson,

$$\sigma(n_{ij} - \mu_{ij}) = \sqrt{\mu_{ij}}.$$

La desviación estándar de $n_{ij} - \hat{\mu}_{ij}$ es menor que $\sqrt{\mu_{ij}}$ pero todavía proporcional a este valor. Definimos el **residuo de Pearson** como

$$e_{ij} = \frac{n_{ij} - \hat{\mu}_{ij}}{\sqrt{\hat{\mu}_{ij}}}.$$

En particular tenemos que el estadístico X^2 de Pearson es igual a la suma de los cuadrados de los residuos de Pearson.

$$X^2 = \sum_i \sum_j e_{ij}^2.$$

Comparar estos residuos con los percentiles normales da una visión conservadora. Se definen los *residuos de Pearson estandarizados* como

$$\frac{n_{ij} - \hat{\mu}_{ij}}{\left[\hat{\mu}_{ij}(1-\hat{\pi}_{i+})(1-\hat{\pi}_{+j})\right]^{1/2}}$$

que sí tienen una distribución normal estándar. Podemos comparar los residuos de Pearson estandarizados con los percentiles de la normal. Valores superiores (en módulo) a 2 o 3 indican falta de ajuste.

Ejemplo 6.6. *Vamos a calcular los residuos de Pearson para el cruce entre educación y fundamentalismo religioso.*

```
religion.counts<-c(178,138,108,570,648,442,138,252,252)
table.3.2=cbind(expand.grid(list(Religious.Beliefs=c("Fund", "Mod", "Lib"),
Highest.Degree=c("<HS","HS or JH", "Bachelor or Grad"))),
count=religion.counts)
table.3.2.array=tapply(table.3.2$count,table.3.2[,1:2], sum)
(res=chisq.test(table.3.2.array))
```

```
        Pearson's Chi-squared test

data:  table.3.2.array
X-squared = 69.157, df = 4, p-value =
3.42e-14
```

```
fit.glm = glm(count~Religious.Beliefs+Highest.Degree, data=table.3.2,
              family=poisson)
```

```
resid.pear = residuals(fit.glm, type = "pearson")
```

y los residuos de Pearson estandarizados.

```
ni<-rowSums(table.3.2.array) # sumas por filas
nj<-colSums(table.3.2.array) # sumas por columnas
n<-sum(table.3.2.array) # tamaño muestral total
resid.pear.mat<-matrix(resid.pear, nc=3, byrow=T,
                       dimnames=list(c("<HS","HS or JH",
"Bachelor or Grad"),c("Fund", "Mod", "Lib")))
n*resid.pear.mat/sqrt(outer(n-ni,n-nj,"*") ) ## residuos de Pearson
```

```
                    Fund Mod Lib
<HS 4.534918 -3.5921772 -2.086799
HS or JH 1.814033 1.2859224 -3.050211
Bachelor or Grad -6.336271 0.9180244 6.252547
```

```
                                        ## estandarizados
```

6.5.4 Test exacto de Fisher para tablas 2×2

Todos los procedimientos vistos hasta ahora se basan en distribuciones asintóticas. Si tenemos muestras grandes no hay problemas. ¿Y

Tabla 6.8

	Predicción primer servicio		
Primer servicio	Leche	Té	Total
Leche	3	1	4
Té	1	3	4
Total	4	4	

con muestras pequeñas? Rezar es una buena opción. Siempre lo es. La otra es un test exacto. Consideramos una tabla 2×2. La hipótesis nula es de independencia. Condicionamos a los totales marginales de fila y columna. Solamente nos queda libre un conteo (por ejemplo, n_{11}) y

$$p(t) = P(n_{11} = t) = \frac{\binom{n_{1+}}{t}\binom{n_{2+}}{n_{+1}-t}}{\binom{n}{n_{+1}}}$$

donde los valores posibles son

$$m_- \leq n_{11} \leq m_+$$

con $m_- = \max\{0, n_{1+} + n_{+1} - n\}$ y $m_+ = \min\{n_{1+}, n_{+1}\}$. Queremos contrastar independencia. En tablas 2×2 lo podemos formular como

$$H_0 : \theta = 1$$

frente a (alternativa unilateral)

$$H_1 : \theta > 1$$

Para el test anterior, si t_0 es el valor observado de n_{11}, entonces el p-valor sería

$$P(n_{11} \geq t_0)$$

Ejemplo 6.7 (La señora Muriel Bristol tomando té)**.** *Sería más extrema la tabla con* $n_{11} = 4$*. El p-valor sería*

$$P(n_{11} = 3) + P(n_{11} = 4) = 0.243$$

Estamos ordenando las tablas de acuerdo con n_{11}. Podríamos ordenarlas según el odds ratio o la diferencia de las proporciones y obtenemos el mismo test. Esto no será cierto para test bilateral. La definición de p-valor depende de cómo ordenamos las tablas. Lo que suele ir programado en software es (si $p(t) = P(n_{11} = t)$)

$$p = P(p(n_{11}) \leq p(t_0))$$

Sumamos la probabilidad de todas aquellas tablas que son tan probables o menos que la tabla observada. Otra opción es

$$p = P\Big(|n_{11} - E(n_{11})| \geq |t_0 - E(n_{11})|\Big)$$

teniendo en cuenta que para la hipergeométrica

$$E(n_{11}) = n_{1+}n_{+1}/n$$

Este procedimiento equivale con

$$p = P(X^2 \geq X_0^2)$$

siendo X_0^2 el valor observado de X^2.

Ejemplo 6.8. *Vamos a aplicar el test exacto de Fisher para datos de Muriel Bristol.*

```
(test ←fisher.test(matrix(c(3, 1, 1, 3), byrow = T, ncol = 2)))
```

```
        Fisher's Exact Test for Count Data

data:  matrix(c(3, 1, 1, 3), byrow = T, ncol = 2)
p-value = 0.4857
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
   0.2117329 621.9337505
sample estimates:
odds ratio
  6.408309
```

Si queremos el p-valor unilateral.

```
test$p.value/2
```

```
[1] 0.2428571
```

Podemos contrastar la alternativa unilateral directamente.

```
 (fisher.test(matrix(c(3, 1, 1, 3), byrow = T, ncol = 2),
              alternative = "greater"))
```

```
        Fisher's Exact Test for Count Data

data:  matrix(c(3, 1, 1, 3), byrow = T, ncol = 2)
p-value = 0.2429
alternative hypothesis: true odds ratio is greater than 1
95 percent confidence interval:
 0.3135693       Inf
sample estimates:
odds ratio
  6.408309
```

Podemos contrastar la hipótesis nula de que el odds ratio sea igual a otro valor no necesariamente igual a 1 (por ejemplo 0.2).

```
(fisher.test(matrix(c(3, 1, 1, 3), byrow = T, ncol = 2), or = 0.2))
```

```
        Fisher's Exact Test for Count Data

data:  matrix(c(3, 1, 1, 3), byrow = T, ncol = 2)
p-value = 0.02246
alternative hypothesis: true odds ratio is not equal to 0.2
95 percent confidence interval:
   0.2117329 621.9337505
sample estimates:
odds ratio
  6.408309
```

6.5.5 Distribución condicionada exacta

Consideramos tablas con $I \times J$. Suponemos muestreo multinomial independiente (por filas). En consecuencia los totales de fila (n_{i+}) están fijados. Asumimos la hipótesis de que la distribución condicionada de cada fila es la misma

$$H_0 : \pi_{j|1} = \ldots = \pi_{j|I} = \pi_{+j} \quad \text{para } j = 1, \ldots, J$$

y condicionamos ahora a los totales de columna. La distribución condicionada de los conteos es

$$\frac{\prod_i n_{i+}! \prod_j n_{+j}!}{n! \prod_i \prod_j n_{ij}!}$$

la *distribución hipergeométrica múltiple*. Si suponemos una única distribución multinomial. Condicionamos al total de la tabla n. Condicionamos a los totales de fila y columna. La independencia se formula como

$$H_0 : \pi_{ij} = \pi_{i+}\pi_{+j} \forall i, j$$

Obtenemos la misma distribución condicionada.

Test exacto de independencia para tablas $I \times J$. Hemos de establecer un orden entre las tablas. La opción más habitual es considerar la probabilidad de la tabla. El p-valor es la probabilidad de las tablas (con las marginales dadas) que no son más probables que la tabla observada. Otra posibilidad es utilizar un estadístico que mida la distancia de una tabla con la hipótesis nula: Como el estadístico X^2.

Ejemplo 6.9.

```
(table.3.9 ←matrix(c(25, 25, 12, 0, 1, 3), byrow = T, ncol = 3))
```

```
     [,1] [,2] [,3]
[1,]   25   25   12
[2,]    0    1    3
```

Ejercicios

* **Ex. 7** — Demostrar que el estadístico score es el que aparece en la ecuación 6.2.

* **Ex. 8** — Demostrar que el estadístico de Wald es el que aparece en la ecuación 6.3.

* **Ex. 9** — Demostrar que el estadístico del cociente de verosimilitudes es el que aparece en la ecuación 6.4.

* **Ex. 10** — Vamos a considerar los datos `vcdExtra::AirCrash` en los que se tiene información sobre accidentes de aviones comerciales entre 1993 y 2015. Se pide:

1. Leer los datos.
2. Con `base::table()` obtener la distribución de frecuencias absolutas de las variables que indican la fase en que se produjo el accidente y la causa del mismo: `Phase` y `Cause`.
3. Obtener con `base::table` la tabla de contingencia para las variables `Phase` y `Cause`.
4. Repetir el apartado 2 utilizando la tabla de contingencia de apartado 3 y las función `table::margin.table`.
5. Repetir los apartados 2 y 3 utilizando `base::xtabs`.
6. Obtener las proporciones totales, por fila y por columna utilizando la función `base::prop.table`.
7. Utilizando `gmodels::CrossTable` obtener la tabla en que aparezcan, en cada celda: los conteos, las proporciones sobre el total, las proporciones por fila y columna.

8. Constrastar la independencia entre las variables `Phase` y `Cause` con un test ji-cuadrado.

* **Ex. 11** — Consideremos los datos `vcdExtra::Cancer`. Se refieren a cancer de pecho. Son datos (antiguos) de supervivencia a tres años. En este problema parece lógico considerar la posible asociación (marginal) entre las variables `Grade` y `Center` y la variable **respuesta** `Survival`. Se pide:

1. Leer los datos.
2. Constrastar la dependencia de `Survival` respecto de `Grade` en cada uno de los centros del estudio utilizando un test de Fisher.
3. Repetir el apartado 2 con un test ji-cuadrado.

* **Ex. 12** — Consideramos los datos `vcdExtra::Titanicp`. Se pide:

1. ¿Existe asociación entre el sexo y si se sobrevivió o no?
2. ¿Qué sexo sobrevivió en mayor proporción?[50]

[50] Recordemos la película Titanic.

Capítulo 7

Modelos lineales generalizados

En § 5 se ha asumido una distribución normal para la respuesta. Podemos llamar a esto la **componente aleatoria** del modelo. Para ser exactos se ha asumido que la distribución condicionada de Y_i a los predictores (variables fenotípicas en el contexto ómico), $\boldsymbol{x}_i$, asumiendo que $Y_i|\boldsymbol{x}_i \sim N(\boldsymbol{x}_i^T\boldsymbol{\beta}, \sigma^2)$. Estamos **eligiendo** las variables predictoras y **asumiendo** que la dependencia de la variable respuesta respecto de estas variables predictoras es a través de la media de la variable,

$$\mu_i = E[Y_i|\boldsymbol{x}_i] = \boldsymbol{x}_i^T\boldsymbol{\beta}.$$

Es la **componente sistemática** del modelo. Estamos **identificando** la media condicionada de la respuesta con la combinación lineal de los predictores. Si denotamos por g la función identidad, $g(\mu_i) = \mu_i$, entonces **enlazamos** (transformamos) la media μ_i con la componente sistemática mediante la función (**de enlace**) identidad.

En este tema estudiamos los **modelos lineales generalizados** que abreviaremos en **GLM** (generalized lineal model). [51] En esencia, la idea es poder trabajar con otras distribuciones de probabilidad para la variable respuesta sin limitarnos a asumir una distribución normal. En concreto generalizaremos a lo que se conoce como **familia de dispersión exponencial**. Esto incluirá distribuciones discretas como la binomial, Poisson o binomial negativa (asumiendo un parámetro de dispersión conocido). Con la distribución binomial modelizamos respuestas binarias en donde, dados unos predictores, observamos un éxito o fracaso (presencia o no de un atributo). Con las distribuciones Poisson o binomial negativa (con la restricción indicada) podremos modelizar datos de conteo (como números de lecturas alineadas sobre un gen en RNA-Seq). También podemos asumir distribuciones continuas como la gamma (también asumiendo un valor conocido para la dispersión) que nos permite modelizar datos no negativos con distribuciones claramente no normales.

[51] Sin duda, un nombre mal elegido por Wedderburn y Nelder porque se confunde con modelo lineal general que consiste en admitir una matriz de covarianzas genérica en lugar de un múltiplo de la matriz identidad. A estas alturas no parece tener remedio.

La componente sistemática la mantendremos tal cual, esto es, las variables predictoras intervienen conjuntamente mediante una combinación lineal de las mismas.

La función de enlace que nos lleva la media de la variable respuesta a la componente sistemática vamos a generalizarla de modo que no nos limitamos a la función identidad.

7.1 Componentes de un modelo lineal generalizado

Empezamos con una presentación formal y genérica de estos modelos. Un modelo lineal generalizado (GLM de un modo abreviado) consta de las siguientes componentes:

Componente aleatoria Identifica la variable respuesta Y y su distribución de probabilidad.

Componente sistemática Especifica las variables explicativas (independientes, predictoras) utilizadas en la función predictora lineal.

[52] Link function.

Función de enlace [52] Especifica la función de EY que la expresa como una combinación lineal de las variables predictoras.

En lo que sigue vamos a ver qué es cada una de estas componentes y estudiaremos los ejemplos más importantes desde el punto de vista de las aplicaciones.

La componente aleatoria de un GLM consiste de una variable aleatoria Y con observaciones independientes $(Y_1, \ldots, Y_n)$. Suponemos la distribución de Y es de la *familia de dispersión exponencial* cuya forma genérica sería

$$f(y_i; \theta_i, \phi) = e^{\frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi)}. \tag{7.1}$$

En esta función de densidad el parámetro θ_i recibe el nombre de **parámetro natural** y ϕ es el **parámetro de dispersión**. Un caso de particular interés ocurre cuando $a(\phi) = 1$ y $c(y_i, \phi) = c(y_i)$. Esta subfamilia recibe el nombre de **familia exponencial natural** con

$$f(y_i; \theta_i) = h(y_i) \exp\left[y_i\theta_i - b(\theta_i)\right]. \tag{7.2}$$

La función $a(\phi)$ puede ser cualquiera aunque por razones prácticas que veremos después suele adoptar la forma

$$a(\phi) = \frac{\phi}{\omega}, \tag{7.3}$$

[53] No es muy común hoy en día pero en [64] y otros artículos a veces se denota el parámetro de dispersión como σ^2. Utilizaremos ϕ en este manual.

siendo ω un valor conocido y ϕ desconocido. El parámetro ϕ[53] es constante y no depende de las distintas observaciones. El valor ω es un peso **conocido** que sí dependerá de la observación.

Denotamos $\ell_i = \ln f(y_i; \theta_i, \phi)$ la logverosimilitud para la observación y_i. La logverosimilitud total es $l = \sum_{i=1}^{n} \ell_i$. Si suponemos que estamos en la familia de dispersión exponencial tendremos

$$\ell_i = [y_i\theta_i - b(\theta_i)]/a(\phi) + c(y_i, \phi),$$

y

$$\frac{\partial \ell_i}{\partial \theta_i} = [y_i - b'(\theta_i)]/a(\phi),$$
$$\frac{\partial^2 \ell_i}{\partial \theta_i^2} = -b''(\theta_i)/a(\phi),$$

siendo $b'(\theta_i)$ y $b''(\theta_i)$ las derivadas segundas de $b()$ evaluadas en θ_i. Bajo condiciones de regularidad, que verifica la familia de dispersión exponencial, tenemos que

$$E\left(\frac{\partial l}{\partial \theta}\right) = 0, \tag{7.4}$$

y

$$-E\left(\frac{\partial^2 l}{\partial \theta^2}\right) = E\left(\frac{\partial l}{\partial \theta}\right)^2. \tag{7.5}$$

Aplicamos la igualdad 7.4 a la logverosimilitud de una sola observación y tenemos

$$E[Y_i - b'(\theta_i)]/a(\phi) = 0,$$

de donde

$$\mu_i = E[Y_i] = b'(\theta_i). \tag{7.6}$$

De la igualdad 7.5 se deduce

$$b''(\theta_i)/a(\phi) = E[(Y_i - b'(\theta_i))/a(\phi)]^2 = var(Y_i)/[a(\phi)]^2,$$

y, en consecuencia,

$$var(Y_i) = a(\phi)b''(\theta_i)). \tag{7.7}$$

La función $a(\phi)$ puede ser cualquier función de ϕ si es conocido. Si no es conocido el parámetro de dispersión ϕ la cosa se complica y entonces se suele asumir que $a(\phi) = \phi/\omega$ siendo ω conocido. Esto cubre la mayor parte de casos prácticos.

Si asumimos que $a(\phi) = \phi/\omega$ podemos considerar la función $\nu(\mu_i) = b''(\theta_i)/\omega$ a la que a veces de denomina **función varianza**[54] de modo que tenemos

[54] Variance function.

$$var(Y_i) = \phi\nu(\mu_i). \tag{7.8}$$

Veremos que esto es muy importante en lo que sigue porque necesitaremos expresar la varianza de la respuesta como función de la media.

Ejemplos

Veamos algunos ejemplos importantes de distribuciones que pertenecen a la familia de dispersión exponencial.

Ejemplo 7.1 (Distribución binomial)**.** *Suponemos que tenemos n_i pruebas Bernoulli que comparten unos mismos predictores $\boldsymbol{x}_i$. Vamos a considerar como respuesta aleatoria Y_i, en lugar del número de éxitos (como es habitual), la proporción de éxitos. Esto significa que el número total de éxitos vendrá dado por n_iY_i. Si denotamos por π_i la probabilidad de éxito común a las n_i pruebas Bernoulli entonces $n_iY_i \sim Bi(n_i, \pi_i)$ con $EY_i = \pi_i$. Consideramos*

$$\theta_i = \ln \frac{\pi_i}{1-\pi_i}$$

esto es, definimos θ_i como el logit de la probabilidad de éxito. Fácilmente se comprueba que la transformación inversa es

$$\pi_i = \frac{e^{\theta_i}}{1+e^{\theta_i}},$$

y que $\ln(1-\pi_i) = -\ln(1+e^{\theta_i}$. Podemos expresar la función de probabilidad de la proporción muestral como

$$f(y_i; \pi_i, n_i) = \binom{n_i}{n_i y_i} \pi_i^{n_i y_i}(1-\pi_i)^{n_i - n_i y_i} = \exp\left[\frac{y_i\theta_i - \ln[1+\exp(\theta_i)]}{1/n_i} + \ln\binom{n_i}{n_i y_i}\right], \tag{7.9}$$

siendo $b(\theta_i) = \ln[1 + \exp(\theta_i)]$, $a(\phi) = 1/n_i$ *y* $c(y_i, \phi) = \ln \binom{n_i}{n_i y_i}$. *El parámetro natural es* $\theta_i = \ln \frac{\pi_i}{1-\pi_i}$, *el logit de* π_i. *Tenemos:*

$$E(Y_i) = b'(\theta_i) = \frac{\exp(\theta_i)}{1 + \exp(\theta_i)} = \pi_i,$$

$$var(Y_i) = a(\phi)b''(\theta_i) = \frac{\exp(\theta_i)}{[1 + \exp(\theta_i)]^2 n_i} = \frac{\pi_i(1 - \pi_i)}{n_i}.$$

Notemos que $a(\phi) = 1/n_i$ *es conocido y tiene la forma de un parámetro* ϕ *(en este caso la unidad) dividida por un cantidad (un peso) conocido (en este caso el número de observaciones que comparten los predictores* $\boldsymbol{x}_i$*).*

Ejemplo 7.2 (Distribución Poisson)**.** *La función de densidad de la distribución Poisson viene dada por*

$$f(y_i; \mu_i) = \frac{e^{-\mu_i} \mu_i^{y_i}}{y_i!} = \exp y_i \ln \mu_i - \mu_i - \ln(y_i!). \tag{7.10}$$

Si tomamos $\theta_i = \ln \mu_i$, $b(\theta_i) = \exp \theta_i$, $a(\phi) = 1$, $c(y_i, \phi) = -\ln(y_i!)$. *Como hemos visto antes* $E[Y_i] = b'(\theta_i) = \exp \theta_i = \mu_i$ *y* $var(Y_i) = a(\phi)b''(\theta_i)) = b''(\theta_i)) = \exp \theta_i = \mu_i$.

Ejemplo 7.3 (Distribución normal)**.** *Vamos a ver el caso de la normal que hemos tratado previamente desde otro punto de vista. La normal pertenece a la familia de dispersión exponencial.*

$$f(y_i; \mu_i, \sigma^2) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{(y_i - \mu_i)^2}{2\sigma^2} \right] = \exp\left[\frac{y_i \mu_i - \frac{1}{2}\mu_i^2}{\sigma^2} - \frac{1}{2}\ln(2\pi\sigma^2) - \frac{y_i^2}{2\sigma^2}. \right] \tag{7.11}$$

Vemos que estamos en la familia de dispersión exponencial donde

$$\theta_i = \mu_i,\ b(\theta_i) = \frac{1}{2}\mu_i^2 = \frac{1}{2}\theta_i^2,\ a(\phi) = \sigma^2,\ c(y_i, \phi) = -\frac{1}{2}\ln(2\pi\sigma^2) - \frac{y_i^2}{2\sigma^2}.$$

Otra vez vemos que $E[Y_i] = b'(\theta_i) = \theta_i = \mu_i$ *y* $var(Y_i) = a(\phi)b''(\theta_i)) = \sigma^2$. *Notemos que* $a(\phi) = \sigma^2$ *tiene la forma de un parámetro* $\phi = \sigma^2$ *dividido por una cantidad que en este caso es la unidad.*

La componente sistemática de un modelo lineal generalizado es el vector $(\eta_1, \ldots, \eta_n)$ donde cada uno de los η_i es la combinación lineal de los predictores correspondientes a la i-ésima observación, es decir,

$$\eta_i = \sum_{j=1}^{n} \beta_j x_{ij} = \boldsymbol{x}_i' \boldsymbol{\beta},$$

con $i = 1, \ldots, N$ donde x_{ij} es el valor del j-ésimo predictor en el i-ésimo individuo. La combinación lineal $\sum_j \beta_j x_{ij}$ es el *predictor lineal*. Como es habitual, se suele considerar que uno de los predictores x_{ij} vale uno para todos los i de modo que consideramos el término independiente o constante.

La función de enlace (link function) g relaciona las componentes aleatoria y sistemática,

$$g(\mu_i) = \eta_i = \boldsymbol{x}_i' \boldsymbol{\beta}.$$

Tendremos que

$$\mu_i = g^{-1}(\eta_i) = g^{-1}(\boldsymbol{x}_i'\boldsymbol{\beta}).$$

A la función g^{-1} en ocasiones se le denomina la función respuesta.

Hay una función de enlace que tiene un interés especial, es aquella que nos transforma la media μ_i en el parámetro natural y recibe el nombre de **enlace canónico** (canonical link). Con esta función de enlace tendremos

$$\theta_i = \boldsymbol{x}_i'\boldsymbol{\beta}.$$

Notemos que en el caso de la normal $\theta_i = \mu_i$ y allí igualamos la media al predictor lineal. De algún modo θ_i es el parámetro que sustituye a la media de la normal de una forma natural.

Con los ejemplos antes comentados tenemos que el enlace canónico es la función logit para la binomial, el logaritmo natural para la Poisson y la función identidad para la normal.

7.2 Verosimilitud, ajuste y distribución asintótica de los GLMs

Con modelos lineales generalizados vamos a estimar el vector de coeficientes maximizando la verosimilitud. Sus estimadores van a ser los máximo verosímiles del mismo modo que en modelos lineales normales.

7.2.1 Verosimilitud

La función de verosimilitud viene dada por

$$L(\boldsymbol{\beta}) = \prod_{i=1}^{n} \exp\left\{\frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi)\right\} \tag{7.12}$$

de donde obtenemos la función de logverosimilitud como

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{n} \frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + \sum_{i=1}^{n} c(y_i, \phi). \tag{7.13}$$

Las ecuaciones de verosimilitud (en la literatura se habla también de ecuaciones de estimación[55]) las obtenemos considerando las derivadas parciales e igualando a cero, en definitiva buscando el punto donde se anula el vector gradiente.

[55] Estimating equations.

$$\frac{\partial \ell(\boldsymbol{\beta})}{\partial \beta_j} = \sum_{i=1}^{n} \frac{\partial \ell_i(\boldsymbol{\beta})}{\partial \beta_j}. \tag{7.14}$$

¿Cómo resolver estas ecuaciones? ¿Hay un procedimiento simple que nos de una solución directa? La respuesta es que no pues

$$\mu_i = g^{-1}(\sum_{j=1}^{p} \beta_j x_{ij}).$$

7.2.2 Distribución asintótica de los estimadores

La distribución conjunta asintótica o distribución con grandes muestras de los estimadores máximo verosímiles de los coeficientes viene dada por el siguiente resultado.

Teorema 7.1. *Asintóticamente* $\hat{\boldsymbol{\beta}}$*,*

$$\hat{\boldsymbol{\beta}} \sim N_p(\boldsymbol{\beta}, (\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}),$$

siendo $\boldsymbol{W}$ *una matriz diagonal con las entradas*

$$w_i = \left(\frac{\partial \mu_i}{\partial \eta_i}\right)^2 \frac{1}{var(Y_i)}.$$

Teniendo en cuenta el teorema 7.1 podemos estimar la matriz de covarianzas asintótica con

$$\widehat{var(\hat{\boldsymbol{\beta}})} = (\boldsymbol{X}^T\hat{\boldsymbol{W}}\boldsymbol{X})^{-1} \tag{7.15}$$

donde $\hat{\boldsymbol{W}}$ es la matriz $\boldsymbol{W}$ evaluada en $\hat{\boldsymbol{\beta}}$.

7.2.3 Estimación del predictor lineal y las medias

Una vez tenemos la distribución asintótica de $\hat{\boldsymbol{\beta}}$ podemos plantearnos estimar la componente sistemática que viene dada por

$$\boldsymbol{\eta} = \boldsymbol{X}\boldsymbol{\beta} \tag{7.16}$$

por lo que la estimamos con

$$\hat{\boldsymbol{\eta}} = \boldsymbol{X}\hat{\boldsymbol{\beta}}. \tag{7.17}$$

La matriz de covarianzas de estos estimadores sería

$$var(\hat{\boldsymbol{\eta}}) = \boldsymbol{X}var(\hat{\boldsymbol{\beta}})\boldsymbol{X}^T. \tag{7.18}$$

Puesto que podemos aproximar la matriz de covarianzas de $\hat{\boldsymbol{\beta}}$ con $(\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}$ que, a su vez, aproximamos con $(\boldsymbol{X}^T\hat{\boldsymbol{W}}\boldsymbol{X})^{-1}$.

¿Cómo estimamos el vector de medias y la matriz de covarianza de los estimadores? Obviamente $g(\mu_i) = \eta_i$ de donde $g(\hat{\mu_i}) = \hat{\eta_i}$. En resumen las medias estimadas serían $\hat{\mu}_i = g^{-1}(\hat{\eta}_i)$. Utilizando el método delta en el caso multivariante podemos aproximar la matriz de covarianzas de $\hat{\boldsymbol{\mu}}$ con

$$var(\hat{\boldsymbol{\mu}}) \approx \boldsymbol{D}var(\hat{\boldsymbol{\eta}})\boldsymbol{D} \approx \boldsymbol{D}\boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{W}\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{D}. \tag{7.19}$$

Obviamente son aproximaciones. Para construir un intervalo de confianza para μ_i lo que se hace es construirlo para η_i y luego aplicar la transformación g^{-1} a los extremos y de este modo no aplicamos la aproximación del método delta.

¿Qué entendemos por distribución asintótica en este contexto? Tenemos n observaciones. Aquí no entendemos por asintótico que el valor de n tienda a infinito. Aquí se considera un número fijo y para cada componente se incrementa el número de observaciones que dan lugar al valor aleatorio de esa componente. En el caso de binomial trabajamos con la media. Se entiende que el valor de n_i crece. En el caso de Poisson que la media crece en cada componente. Estos casos suelen tener $a(\phi) = \phi/\omega_i$ con un ω_i creciendo.

7.3 Bondad de ajuste

Estamos usando un modelo estocástico. Un modelo estocástico siempre es una visión simplificada del proceso que produce nuestros datos y_i. Utilizando este modelo tendremos unas predicciones o valores ajustados que denotamos por $\hat{\mu}_i$ (o $\hat{y}_i$). La diferencia entre ambos valores es lo que nos da la medida en que el modelo puede ser considerado aproximadamente válido. En esta sección se evalúa la bondad del ajuste. Hasta qué punto el modelo produce predicciones que son razonablemente compatibles con los datos observados.

7.3.1 Desviación

Consideramos un GLM con observaciones $\boldsymbol{y} = (y_1, \ldots, y_n)$ Sea $\ell(\boldsymbol{\mu}; \boldsymbol{y})$ la logverosimilitud expresada como función del vector de medias $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)$. La máxima logverosimilitud será $\ell(\hat{\boldsymbol{\mu}}; \boldsymbol{y})$. Consideremos ahora el modelo en que no utilizamos para nada los predictores y estimamos la media μ_i con la propia observación y_i, esto es, hacemos $\mu_i = y_i$. La logverosimilitud en este caso será $\ell(\boldsymbol{y}; \boldsymbol{y})$. A este modelo con un parámetro por observación se le llama **modelo saturado**. Tenemos un ajuste perfecto completamente inútil. La diferencia entre el valor observado, y_i, y el valor estimado para su media (y para la observación misma),y_i, sería nulo. ¿Y de qué sirve cuando tengamos otros predictores en donde no conocemos la media? No sirve nada más que como modelo base. Los demás modelos los compararemos con este modelo saturado. Son el máximo alcanzable de la logverosimilitud, un valor de referencia.

Si $\hat{\boldsymbol{\mu}}$ es el estimador máximo verosímil bajo el modelo que estamos considerando entonces el estadístico del cociente de verosimilitudes contrastando H_0 frente a un modelo más general sería

$$-2\ln\frac{\text{máxima verosilimitud bajo el modelo}}{\text{máxima verosimilitud bajo el modelo saturado}} = -2[\ell(\hat{\boldsymbol{\mu}}; \boldsymbol{y})) - \ell(\boldsymbol{y}; \boldsymbol{y})]. \quad (7.20)$$

Si $\hat{\theta}_i$ denota el estimador máximo verosímil de θ_i y $\tilde{\theta}_i$ denota el estimador de θ_i cuando tenemos el modelo saturado entonces la expresión del cociente sería

$$-2[\ell(\hat{\boldsymbol{\mu}}; \boldsymbol{y})) - \ell(\boldsymbol{y}; \boldsymbol{y})] = 2\sum_{i=1}^{n}\frac{y_i\tilde{\theta}_i - b(\tilde{\theta}_i)}{a(\phi)} - 2\sum_{i=1}^{n}\frac{y_i\hat{\theta}_i - b(\hat{\theta}_i)}{a(\phi)}. \quad (7.21)$$

En el caso particular y frecuente en que $a(\phi) = \frac{\phi}{\omega_i}$ entonces

$$2\frac{1}{\phi}\sum_{i=1}^{n}\omega_i[y_i(\tilde{\theta}_i - \hat{\theta}_i) - b(\tilde{\theta}_i) + b(\hat{\theta}_i)] = \frac{D(y; \hat{\boldsymbol{\mu}})}{\phi}. \quad (7.22)$$

y recibe el nombre de **desviación escalada** mientras que $D(y; \hat{\boldsymbol{\mu}})$ recibe el nombre de **desviación** del modelo actual.

Definición 7.1 (Desviación)**.**

$$D(y; \hat{\boldsymbol{\mu}}) = 2\sum_{i=1}^{n}\omega_i[y_i(\tilde{\theta}_i - \hat{\theta}_i) - b(\tilde{\theta}_i) + b(\hat{\theta}_i)]. \quad (7.23)$$

Definición 7.2 (Desviación escalada)**.**

$$D^*(y;\hat{\boldsymbol{\mu}}) = \frac{D(y;\hat{\boldsymbol{\mu}})}{\phi}. \tag{7.24}$$

Es obvio que $\ell(\hat{\boldsymbol{\mu}};\boldsymbol{y})) \leq \ell(\boldsymbol{y};\boldsymbol{y}))$ por lo que $D(y;\hat{\boldsymbol{\mu}}) \geq 0$.

¿Qué expresión adopta la desviación en los modelos más habituales?

Ejemplo 7.4 (Desviación para GLM binomial)**.** *Denotamos por y_i la proporción de éxitos en un total de n_i pruebas con los mismos predictores $\boldsymbol{x}_i$. Recordemos que $\theta_i = \ln \frac{\pi_i}{1-\pi_i}$, $b(\theta_i) = \ln[1+\exp(\theta_i)]$, y $a(\phi) = 1/n_i$. Por tanto, $\hat{\theta}_i = \ln \frac{\hat{\pi}_i}{1-\hat{\pi}_i}$ y $\tilde{\theta}_i = \ln \frac{\tilde{\pi}_i}{1-\tilde{\pi}_i}$. La desviación vendría dada por*

$$\begin{aligned} &D(\boldsymbol{y};\hat{\boldsymbol{\pi}}) = \\ &2\sum_i n_i \left[y_i \left(\ln \frac{y_i}{1-y_i} - \ln \frac{\hat{\pi}_i}{1-\hat{\pi}_i} \right) + \ln(1-y_i) - \ln(1-\hat{\pi}_i) \right] = \\ &2\sum_i n_i y_i \ln \frac{n_i y_i}{n_i - n_i y_i} - 2\sum_i n_i y_i \ln \frac{n_i \hat{\pi}_i}{n_i - n_i \hat{\pi}_i} + 2\sum_i n_i \ln \frac{1-y_i}{1-\hat{\pi}_i} = \\ &2\sum_i n_i y_i \ln \frac{n_i y_i}{n_i \hat{\pi}_i} + 2\sum_i (n_i - n_i y_i) \ln \frac{n_i - n_i y_i}{n_i - n_i \hat{\pi}_i}. \end{aligned} \tag{7.25}$$

Si observamos la expresión de la desviación que acabamos de obtener tenemos que $n_i y_i$ y $n_i - n_i y_i$ corresponden con los éxitos y fracasos observados. Por otro lado $n_i \hat{\pi}_i$ y $n_i - n_i \hat{\pi}_i$ nos dan los éxitos y fracasos que el modelo predice que podemos llamar con propiedad éxitos y fracasos ajustados. Por tanto si consideramos los $2n$ conteos correspondientes a éxitos y fracasos en cada una de las observaciones podemos expresar de un modo resumida la expresión de la desviación como

$$D(\boldsymbol{y};\hat{\boldsymbol{\pi}}) = 2\sum Observado \ln \frac{Observado}{Esperado}. \tag{7.26}$$

*Esta situación se conoce como **datos agrupados**. Cuando el tamaño de la muestra crece se asume que el valor de n no crece y lo que crecen son los valores de los distintos n_i.*

Con datos no agrupados asumimos que $n_i = 1$ y la respuesta es 1 o 0. En este caso un incremento de la muestra corresponde con un incremento de n

Ejemplo 7.5 (Desviación con GLM Poisson)**.** *Hemos visto que $\hat{\theta}_i = \ln \hat{\mu}_i$ y $b(\hat{\theta}_i) = \exp \hat{\theta}_i = \hat{\mu}_i$. En el modelo saturado: $\tilde{\theta}_i = \ln y_i$ y $b(\tilde{\theta}_i) = y_i$. Además $a(\phi) = 1$. En consecuencia, tanto la desviación como la desviación escalada vienen dadas por*

$$D(y;\hat{\boldsymbol{\mu}}) = 2\sum_{i=1}^{n} [y_i \ln \frac{y_i}{\hat{\mu}_i} - y_i + \hat{\mu}_i].$$

Si se asumen que utilizamos el enlace logarítmico y que tenemos la constante en el modelo tendremos que $\sum_{i=1}^n y_i = \sum_{i=1}^n \hat{\mu}_i$. La desviación queda como

$$D(\boldsymbol{y};\hat{\boldsymbol{\mu}}) = 2\sum_{i=1}^{n} y_i \ln \frac{y_i}{\hat{\mu}_i}.$$

Si observamos la expresión que hemos obtenido también puede expresarse (como en el caso de la binomial) en la forma simplificada

$$D(\boldsymbol{y};\hat{\boldsymbol{\mu}}) = 2\sum Observado \ln \frac{Observado}{Esperado}. \tag{7.27}$$

La desviación que nos aparece en este caso corresponde con el estadístico G^2 ([2, 3]).

Una de las aplicaciones fundamentales de los GLM Poisson es modelizar conteos en tablas de contingencia. En este caso n, total de conteos es fijo y que crece son los valores esperados en cada celda. Bajo esta condición la desviación converge a una ji-cuadrado con $n-p$ grados de libertad donde p es el número de parámetros del modelo.

Ejemplo 7.6 (Desviación con GLM normal)**.** *Tenemos que $\hat{\theta}_i = \hat{\mu}_i$, $b(\hat{\theta}_i) = (\hat{\theta}_i)/2$. En el caso del modelo saturado tenemos $\theta_i = y_i$ y $b(\tilde{\theta}_i) = y_i^2/2$. La desviación adopta la siguiente expresión*

$$D(\boldsymbol{y};\hat{\boldsymbol{\mu}}) = 2\sum_{i=1}^{n}\left[y_i(y_i-\hat{\mu}_i) - \frac{y_i^2}{2} + \frac{\hat{\mu}_i^2}{2}\right] = \sum_{i=1}^{n}(y_i-\hat{\mu}_i)^2. \tag{7.28}$$

Por tanto la desviación coincide con la suma de cuadrados residual o suma de los cuadrados de los residuos. En este caso, el parámetro de dispersión es σ^2 por tanto la desviación escalada sería $\sum_{i=1}^{n}(y_i - \hat{\mu}_i)^2/\sigma^2$. Asumiendo el modelo lineal normal el teorema de Cochran nos afirma que la desviación escalada sigue una distribución χ^2_{n-p}. Vemos pues cómo los estimadores mínimo cuadráticos de los coeficientes en el modelo lineal normal lo que hacía era minimizar la suma de cuadrados residual.

7.3.2 Residuos

Supongamos un modelo con una función varianza $V(\mu)$ (recordemos que $var(Y) = a(\phi)b''(\theta_i) = a(\phi)V(\mu)$). Una forma natural de evaluar la bondad de ajuste del modelo es considerar la diferencia entre la observación y su predicción (de media u observación que son la misma). Un primer ejemplo es el residuo de Pearson.

Definición 7.3 (Residuo de Pearson)**.** *Siendo la función varianza $\nu(\mu)$ definimos el **residuo de Pearson** para la observación y_i como*

$$e_i = \frac{y_i - \hat{\mu}_i}{\sqrt{\nu(\hat{\mu}_i)}}. \tag{7.29}$$

Ejemplo 7.7 (GLM binomial)**.** *Recordemos que y_i denota la proporción de éxitos en n_i pruebas. Tenemos $V(\pi_i) = \pi_i(1-\pi_i)$ y por lo tanto el residuo de Pearson será*

$$e_i = \frac{y_i - \hat{\pi}_i}{\sqrt{\hat{\pi}_i(1-\hat{\pi}_i)}}. \tag{7.30}$$

Ejemplo 7.8 (GLM Poisson)**.** *En este caso $V(\mu) = \mu$ y por tanto el residuo de Pearson viene dado por*

$$e_i = \frac{y_i - \hat{\mu}_i}{\sqrt{\hat{\mu}_i}}. \tag{7.31}$$

Consideremos la desviación

$$D(\boldsymbol{y};\hat{\boldsymbol{\mu}}) = \sum_{i=1}^{n} d_i,$$

donde

$$d_i = 2\omega_i[y_i(\tilde{\theta}_i - \hat{\theta}_i) - b(\tilde{\theta}_i) + b(\hat{\theta}_i)].$$

Definición 7.4 (Desviación residual)**.** *Se define la* ***desviación residual*** *o* ***residuo de la desviación*** *como*

$$\sqrt{d_i} \times signo(y_i - \hat{\mu}_i).$$

Por la definición que hemos dado desviación residual tenemos que la suma de los cuadrados de la desviación residual es igual a la desviación.

7.4 Estimación del parámetro de escala

[56] Seguimos en esta sección [94, pág. 110-111]

[56] Lo primero es observar que hemos podido estimar el vector de coeficientes $\boldsymbol{\beta}$ sin estimar el parámetro de escala ϕ. Ya lo vimos en modelos lineales normales que estimábamos la varianza del error aleatorio a partir de los residuos una vez teníamos estimado el vector de coeficientes.

Hemos indicado que habitualmente $a(\phi) = \phi/\omega$ siendo ω un valor conocido. Sin embargo, ¿cómo estimamos ϕ? Lo primero es que la estimación de los coeficientes no ha implicado en ningún momento la estimación del parámetro ϕ de escala. La desviación escalada, $D^* = D/\phi$, bajo ciertas condiciones, sigue aproximadamente una distribución χ^2_{n-p}. De donde aproximadamente

$$E\frac{D}{\phi} = n - p,$$

y un posible estimador sería

$$\hat{\phi}_D = \frac{\hat{D}}{n-p}. \tag{7.32}$$

Un segundo estimador utiliza los residuos de Pearson cuya suma de cuadrados constituye el estadístico de Pearson dado por

$$X^2 = \sum_{i=1}^{n} \frac{(y_i - \hat{\mu}_i)^2}{\nu(\hat{\mu}_i)}$$

Se tiene que $\nu(\mu_i) = \phi\nu^*(\mu_i)$

7.5 Respuesta binaria

[1] Nuestra variable respuesta es binaria: presencia o ausencia de un atributo. Nuestra variable respuesta la reducimos a dos posibles resultados. A uno de ellos arbitrariamente lo podemos llamar éxito (codificarlo como 1 es razonable) y el al otro resultado lo llamamos fracaso (0).

[1] Esta sección utiliza fundamentalmente el capítulo 5 de [4] y el capítulo 5 de [2]. El código utilizado para los ejemplos de [2] en parte procede de [85].

7.5.1 Datos

Leemos los datos. Son datos en donde tenemos SNPs, esto es, polimorfismos que afectan a un solo par de bases en el genoma. Son datos anonimizados. Leemos y arreglamos un poco los datos. La variable respuesta será `DM2` que nos indica si la persona tiene diabetes mellitus de tipo 2 o no. Las variables predictoras a considerar son las demás.

```
finput = system.file("extdata","SNPs_RM.csv",
                     package="tamidata3")
df = read.table(file = finput,header = TRUE,sep=",")
sel = c(1:5,8,10)
for(i in sel) df[,i] = as.factor(df[,i])
df[,6] = (df[,6] == "YES")*1.
df = df[complete.cases(df),]
names(df)[6:10] = c("DM2","age","gender","weight",
                    "smoking")
```

Podemos ver las primeras filas de los datos.

```
head(df,n=2)
```

```
  SNP1 SNP2 SNP3 SNP4 SNP5 DM2 age gender weight
2 AG AA AG GT TT 0 46 Female 72
6 GG AG AG GG CT 0 32 Male 69
    smoking
2 ex-smoker
6 ex-smoker
```

Los SNPs aparecen en las primeras columnas. Hemos de transformar estas variables para convertirlas en posibles predictores de acuerdo con distintos modelos genéticos. La siguiente función lo hace.

```
#' Transformation of the SNP to a genetic model
#' @description
#' Transformation of the SNP to a genetic model
#' @param x SNPs
#' @param type Model to be used
#' @param sep Separator
#' @export
  snp2model = function(x,type=c("codominant","dominant",
                                "recessive"),
                       sep=""){
    type = match.arg(type)
    x1 = substr(x,1,1)
    x2 = substr(x,2,2)
    a = table(c(x1,x2))
    recessive = names(which.min(a))
    dominant = names(which.max(a))
    x1 = (x1 == recessive)*1
    x2 = (x2 == recessive)*1
    if(type == "codominant") rs = x1+x2
    if(type == "dominant") rs = (x1+x2 == 0)*1.
    if(type == "recessive") rs = (x1+x2 == 2)*1.
    rs
  }
```

Construimos tres **`data.frame`**s donde los SNPs son codificados según cada uno de los modelos considerados, el mismo para todos ellos. Esto no tiene porqué ser así y podríamos considerar un modelo distinto para cada uno de ellos.

```
snpscol = 1:5
dfc = dfd = dfr = df
dfc[,snpscol] = apply(df[,snpscol],2,snp2model,
                      type="codominant")
```

```
dfd[,snpscol] = apply(df[,snpscol],2,snp2model,
                      type="dominant")
dfr[,snpscol] = apply(df[,snpscol],2,snp2model,
                      type="recessive")
```

Estos datos los usamos en lo que sigue para ilustrar.

7.5.2 Función de enlace

Para un conjunto de predictores dados x_i tendremos n_i pruebas Bernoulli que darán lugar a una observación binomial (bien como número de éxitos bien como una proporción de éxitos). Denotaremos las respuestas $y_i, \ldots, y_N$ y supondremos que denotan las proporciones de éxito, esto es, $n_i Y_i \sim Bi(n_i, \pi_i)$ para $i = 1, \ldots, N$. Cada i denota ahora una situación en la que repetimos n_i pruebas. Normalmente los predictores serán categóricos. En el caso de algún predictor continuo se ha de asumir que repetimos este valor en todas las pruebas. Por tanto:

$$EY_i = \mu_i = \pi_i.$$

El vector $(n_1, \ldots, n_N)$ denota los tamaños muestrales. Tenemos n_i pruebas asociadas a un mismo vector de predictores $\boldsymbol{x}_i$. El total de observaciones es $n = \sum_{i=1}^{N} n_i$.

Podemos considerar los datos *agrupados* y *no agrupados*. ¿Qué significa esto? Podemos considerar los datos originales en donde repetimos las covariables $\boldsymbol{x}_i$ y consideramos como respuesta uno o cero, esto es, la respuesta binaria. Tenemos una distribución Bernoulli de la respuesta. Obviamente, en este contexto, cuando hablamos de resultado asintótico nos referimos a que el tamaño $N = n$ tiende a infinito.

En el segundo caso, consideramos los datos agrupados. Esto corresponde habitualmente a la situación en que **todos** los predictores son categóricos. Si tenemos algún predictor que sea numérico (continuo) raramente tendremos más de una observación. En esta situación el valor de N es fijo y lo que crece es n_i para todos los i. El tamaño de las muestras en cada situación crece pero no el número de situaciones distintas que consideramos. A esto se le llama **small dispersion asymptotics**. Obviamente logramos disminuir la varianza de la proporción de éxitos en cada situación sin incrementar el número de situaciones distintas.

Ambas situaciones dan los mismos estimadores de los coeficientes y sus errores estándar son los mismos. No así la desviación. Es más cómodo trabajar con datos agrupados.

7.5.3 Modelos con variable latente

Supongamos que nuestra variable respuesta, Y^*, es una variable que no podemos observar y que verifica

$$Y_i^* = \sum_{j=1}^{p} \beta_j x_{ij} + \epsilon_i.$$

Suponemos que los errores aleatorios ϵ_i son independientes con media nula y función de distribución común F. Supongamos que hay un punto de corte desconocido τ tal que la variable que realmente estamos observando es $Y_i = 0$ si $Y_i^* \leq \tau$ e $Y_i = 1$ si $Y_i^* > \tau$. Estamos

considerando que lo que vemos es una versión discretizada en dos valores de una variable continua subyacente o latente que no podemos observar realmente. La función de probabilidad de la variable binaria observada se puede expresar en términos de la función de distribución F y del punto de corte desconocido τ.

$$P(Y_i = 1) = \pi_i = P(Y_i^* > \tau) = P\Big(\sum_{j=1}^{p} \beta_j x_{ij} + \epsilon_i > \tau\Big) = \\ 1 - \Big(\epsilon_i \leq \tau - \sum_{j=1}^{p} \beta_j x_{ij}\Big) = 1 - F\Big(\tau - \sum_{j=1}^{p} \beta_j x_{ij}\Big). \quad (7.33)$$

Los datos nunca nos van a permitir conocer τ. Podemos pues considerar de un modo arbitrario que τ es nulo. De modo que

$$P(Y_i = 1) = \pi_i = 1 - F\Big(-\sum_{j=1}^{p} \beta_j x_{ij}\Big). \quad (7.34)$$

Si todos los parámetros son multiplicados por un mismo valor el modelo sería equivalente a dividir por ese mismo valor las predictoras. Realmente hablamos de un modelo equivalente en lo esencial. Podemos asumir también varianza dada. En los modelos habituales se suele considerar una distribución simétrica respecto del origen: $F(x) = 1 - F(-x)$. Nos queda

$$P(Y_i = 1) = \pi_i = F\Big(\sum_{j=1}^{p} \beta_j x_{ij}\Big). \quad (7.35)$$

En consecuencia se tiene que

$$F^{-1}(P(Y_i = 1)) = F^{-1}(\pi_i) = \sum_{j=1}^{p} \beta_j x_{ij}. \quad (7.36)$$

Los modelos más utilizados corresponden a este tipo de modelos.

Modelo probit

Un ejemplo de lo que acabamos de ver correspondería al caso en que Φ es la función de distribución de la normal estándar. Una función de enlace a utilizar es Φ^{-1}. Cuando utilizamos esta función de enlace hablamos de un **modelo probit**. En este caso tenemos, por ecuación 7.36, que

$$\Phi^{-1}(P(Y_i = 1)) = \Phi^{-1}(\pi_i) = \sum_{j=1}^{p} \beta_j x_{ij}, \quad (7.37)$$

o bien que

$$P(Y_i = 1) = \pi_i = \Phi\Big(\sum_{j=1}^{p} \beta_j x_{ij}\Big). \quad (7.38)$$

Este es el modelo de **regresión probit**.

Modelo logit

Sin duda este modelo es el más usado en las aplicaciones y es conocido como *regresión logística*. La función de densidad de una **distribución logística estándar** es

$$f(x) = \frac{e^{-x}}{(1+e^{-x})^2} = \frac{1}{(e^{x/2}+e^{-x/2})^2} \tag{7.39}$$

con $x \in \mathbb{R}$. Su función de distribución viene dada por

$$F(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{1+e^x}. \tag{7.40}$$

En este modelo tendremos que

$$P(Y_i = 1) = \pi_i = F\left(\sum_{j=1}^{p} \beta_j x_{ij}\right) = \frac{e^{\sum_{j=1}^{p} \beta_j x_{ij}}}{1+e^{\sum_{j=1}^{p} \beta_j x_{ij}}} \tag{7.41}$$

Pero

$$F^{-1}(\pi_i) = \ln \frac{\pi_i}{1-\pi_i} = \sum_{j=1}^{p} x_{ij}\beta_j. \tag{7.42}$$

La función **logit** se define precisamente como

$$logit(\pi_i) = \ln \frac{\pi_i}{1-\pi_i}. \tag{7.43}$$

Por lo que en un modelo de **regresión logística** tenemos que el logit de la probabilidad de éxito es igual a la componente sistemática.

$$logit(\pi_i) = \sum_{j=1}^{p} x_{ij}\beta_j. \tag{7.44}$$

¿Qué significan los coeficientes β_j? Obviamente (como en todo modelo lineal o lineal generalizado) si $\beta_j = 0$ indica que la variable respuesta es condicionalmente independiente (dadas el resto de variables predictoras) de la j-ésima variable predictora. Notemos que $\frac{\pi_i}{1-\pi_i}$ son los odds. Nos fijamos en la j-ésima variable predictora. Asumimos que el efecto de las demás y la constante se mantiene constante y comparamos subpoblaciones con un valor dado x_{ij} con otra subpoblación con valor $x_{ij}+1$. Supongamos que los predictores de la primera y segunda subpoblaciones los denotamos $\boldsymbol{x}_i$ y $\boldsymbol{x}_i^*$. Denotamos por π_i y π_i^* las probabilidades correspondientes a los vectores $\boldsymbol{x}_i$ y $\boldsymbol{x}_i^*$. De (7.44) se sigue inmediatamente que

$$logit(\pi_i^*) - logit(\pi_i) = \beta_j, \tag{7.45}$$

pero

$$\beta_j = logit(\pi_i^*) - logit(\pi_i) = \ln\left(\frac{\pi_i^*/(1-\pi_i^*)}{\pi_i/(1-\pi_i}\right), \tag{7.46}$$

y el cociente que tenemos a la parte derecha de (7.46) es el logaritmo del cociente de los odds o log-odds ratio. O si lo preferimos

$$e^{\beta_j} = \frac{\pi_i^*/(1-\pi_i^*)}{\pi_i/(1-\pi_i}, \tag{7.47}$$

es decir, e^{β_j} corresponde con el cociente de los odds. El usuario suele encontrar más fácil interpretar el valor de e^{β_j}. Si es mayor que la unidad indica que el cociente de odds se incrementa en esa cantidad. Menor que la unidad indica un decremento. Obviamente el sentido del cambio viene en función de la codificación de la variable predictora que estamos considerando.

Ejemplo 7.9. *Vamos a considerar dos variables categóricas X e Y con I y J categorías. Un sujeto puede venir clasificado en una de $I \times J$ categorías.*

Nos fijamos en el modelo dominante y consideramos las variables `SNP1` *y* `clinica.DM2`.

```
(tab = table(dfd[,c("SNP1","DM2")]))
```

```
   DM2
SNP1  0  1
   0 351 30
   1 746 50
```

Podemos calcular el cociente de odds estos datos.

```
fisher.test(tab)
```

```

        Fisher's Exact Test for Count Data

data:  tab
p-value = 0.3232
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 0.4796518 1.3014415
sample estimates:
odds ratio
 0.7843749
```

O simplemente

```
DescTools::OddsRatio(tab,conf.level=.95)
```

```
odds ratio lwr.ci upr.ci
 0.7841823 0.4900784 1.2547826
```

Observemos que los intervalos de confianza son distintos pues se han calculado con diferentes métodos.

Podemos ajustar un modelo de regresión logística con respuesta `DM2` *y predictora la variable* `SNP1`.

```
fit = glm(DM2 ~SNP1, family="binomial",data=dfd)
summary(fit)
```

```

Call:
glm(formula = DM2 ~SNP1, family = "binomial", data = dfd)

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept) -2.4596 0.1902 -12.930 <2e-16
SNP1 -0.2431 0.2398 -1.014 0.311

(Intercept) ***
SNP1
---
Signif. codes:
```

```
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 584.63 on 1176 degrees of freedom
Residual deviance: 583.62 on 1175 degrees of freedom
AIC: 587.62

Number of Fisher Scoring iterations: 5
```

Podemos ver el odds ratio con

```
exp(coef(fit)[2])
```

```
     SNP1
0.7841823
```

Obviamente utilizando el modelo de regresión logística podemos considerar otras variables predictoras que nos permitan evitar confundir la influencia que tiene el uso de la aspirina con otras posibles variables predictoras no consideradas. El utilizar el cociente de odds se acaba aquí.

7.5.4 Predicción y error de clasificación

Una vez hemos ajustamo un modelo de regresión logística tenemos, para cada observación, una probabilidad de éxito estimada, $\hat{\pi}_i$. Si queremos una clasificación de la observación en éxito o fracaso podemos fijar una probabilidad de corte, π_0, y fijar una regla de clasificación que sea: si $\hat{\pi}_i > \pi_0$ entonces tomamos $\hat{y}_i = 1$, esto es, clasificamos la i-ésima observación como éxito. Si $\hat{\pi}_i \leq \pi_0$ entonces clasificamos como fracaso, $\hat{y}_i = 0$. Una probabilidad de corte habitualmente usada es $\pi_0 = 0.5$. Obviamente conocemos el grupo al que pertenece la observación. Conocemos y_i y por ello podemos construir una tabla de contingencia en donde consideremos los conteos de $(y_i, \hat{y}_i)$. Esta tabla recibe el nombre de **tabla de clasificación**.

Es habitual cuando aplicamos el procedimiento indicado utilizar el procedimiento leaving-one-out. ¿Cómo se hace? Para cada observación ajustamos el modelo sin utilizar esta observación, la dejamos fuera. Con este modelo ajustado sin la observación hacemos la predicción de la probabilidad π_i que podemos denotar, $\hat{\pi}_{(i)}$. Aplicamos la regla de decisión indicada con estas probabilidades y tendremos las predicciones $\hat{y}_{(i)}$. Construimos la tabla de clasificación con $(y_i, \hat{y}_{(i)})$.

La tabla de clasificación depende de la probabilidad de corte π_0. ¿Qué ocurre cuando queremos evaluar los resultados para distintos valores de π_0? Dado un valor de π_0 podemos considerar la sensibilidad y la especificidad. La sensibilidad sería la proporción de éxitos que son declarados como éxitos. Dicho de otro modo, la tasa de verdaderos positivos. En términos probabilísticos $P(\hat{Y} = 1|Y = 1)$. También podemos considerar la especificidad o proporción de fracasos declarados como fracasos, la tasa de verdaderos negativos: $P(\hat{Y} = 0|Y = 0)$.

Supongamos que vamos tomando valores de π_0 de 1 a 0 decreciendo. Para cada valor de π_0 consideramos uno menos la especificidad en abscisas ($P(\hat{Y} = 1|Y = 0)$)) y la sensibilidad ($P(\hat{Y} = 1|Y = 1)$) en ordenadas. Esta es la curva ROC.[57] Una curva ROC es tanto mejor cuando más rápidamente crece y cuanto más próxima a uno está. El puro azar, la clasificación completamente aleatoria, daría una curva ROC igual a la función identidad.

[57] Receiver operating characteristic.

7.5.5 Desviación y bondad de ajuste

Una forma de evaluar el modelo que utilizamos es compararlo con modelos más complejos y ver que el ajuste no es mejor. De algún modo, estamos viendo que nuestro modelo es razonable, no es un mal modelo.

Otra forma de ver si el modelo falla es utilizar la desviación o estadísticos de Pearson.

7.5.6 Desviación y estadísticos de Pearson

La desviación compara nuestro modelo con un modelo saturado en donde igualamos cada una de las medias con la observación, $\tilde{\pi}_i = y_i$. La desviación viene dada por

$$-2\ln\frac{\prod_{i=1}^{N}\hat{\pi}_i^{n_i y_i}(1-\hat{\pi}_i)^{n_i-n_i y_i}}{\prod_{i=1}^{N}\tilde{\pi}_i^{n_i y_i}(1-\tilde{\pi}_i)^{n_i-n_i y_i}} = 2\sum_{i=1}^{N} n_i y_i \ln\frac{n_i y_i}{n_i\hat{\pi}_i} + 2\sum_{i=1}^{N}(n_i - n_i y_i)\ln\frac{n_i - n_i y_i}{n_i - n_i\hat{\pi}_i}. \quad (7.48)$$

Tenemos N valores distintos de las covariables y $n_i y_i$ es el número de éxitos en la situación i mientras que el número de fracasos viene dado por $n_i - n_i y_i$. Por tanto, $n_i y_i$ y $n_i - n_i y_i$ son las valores **observados** de exitos y fracasos mientras que $n_i\hat{\pi}_i$ y $n_i - n_i\hat{\pi}_i$ son los éxitos y fracasos **ajustado** por el modelo. Por ello, de un modo abreviado, podemos escribir

$$D(\boldsymbol{y};\hat{\boldsymbol{\mu}}) = 2\sum Observado\ln\frac{Observado}{Ajustado}.$$

La desviación es distinta si consideramos las observaciones agrupadas o sin agrupar.

Con datos agrupados se puede utilizar el estadístico de Pearson como medida de bondad de ajuste del modelo.

$$X^2 = \sum_{i=1}^{n}\frac{(n_i y_i - n_i\hat{\pi}_i)^2}{n_i\hat{\pi}_i} + \sum_{i=1}^{n}\frac{(n_i - n_i y_i) - (n_i - n_i\hat{\pi}_i))^2}{n_i - n_i\hat{\pi}_i} = \sum\frac{(Observado\ -\ Ajustado)^2}{Ajustado}. \quad (7.49)$$

7.5.7 Residuos

Cuando tenemos datos agrupados, es útil comparar las proporciones observadas y ajustadas: y_i con $\hat{\pi}_i$. El residuo de Pearson viene dado por

$$e_i = \frac{y_i - \hat{\pi}_i}{\sqrt{\hat{\pi}_i(1-\hat{\pi}_i)/n_i}}.$$

Notemos que por la definición de este residuo tenemos que

$$X^2 = \sum_{i=1}^{N} e_i^2.$$

Podemos usar los residuos de la desviación. O bien utilizar los residuos estandarizados en donde el residuo $y_i - \hat{\pi}_i$ por su error estándar estimado. Si consideramos la matriz

$$\hat{\boldsymbol{H}}_{\boldsymbol{W}} = \hat{\boldsymbol{W}}^{1/2}\boldsymbol{X}(\boldsymbol{X}^T\hat{\boldsymbol{W}}\boldsymbol{X})^{-1}\boldsymbol{X}^T\hat{\boldsymbol{W}}^{1/2}$$

siendo $\hat{\boldsymbol{W}}$ la matriz diagonal con $\hat{w}_{ii} = n_i\hat{\pi}_i(1-\hat{\pi}_i)$. Si $\hat{h}_{ii}$ denota el elemento i-ésimo en la diagonal principal de $\hat{\boldsymbol{H}}_{\boldsymbol{W}}$ entonces el residuo estandarizado viene dado por

$$r_i = \frac{e_i}{\sqrt{1-\hat{h}_{ii}}}.$$

Si el modelo es correcto y n_i es grande entonces los residuos estandarizados tienen aproximadamente una distribución normal estándar.

7.5.8 Analizando SNPs

En esta sección vamos a evaluar cada uno de los modelos genéticos buscando asociación con la respuesta `clinica.DM2` que nos indica si la persona tiene la diabetes mellitus tipo 2.

Vamos a evaluar los modelos genéticos buscando asociación con la respuesta así como otras variables predictoras. Empezamos considerando el modelo codominante. Empezamos ajustando un modelo en el que consideramos como predictoras `SNP1`, `age`, `gender`, `weight` y `smoking`.

```
fit.c1 = glm(DM2 ~SNP1 + age + gender + weight +
                smoking, family="binomial",data=dfc)
fit.c2 = glm(DM2 ~SNP2 + age + gender + weight +
                smoking, family="binomial",data=dfc)
fit.c3 = glm(DM2 ~SNP3 + age + gender + weight +
                smoking, family="binomial",data=dfc)
fit.c4 = glm(DM2 ~SNP4 + age + gender + weight +
                smoking, family="binomial",data=dfc)
fit.c5 = glm(DM2 ~SNP5 + age + gender + weight +
                smoking, family="binomial",data=dfc)
```

Para poder comparar podemos

```
fit.c0 = glm(DM2 ~age + gender + weight + smoking,
             family="binomial",data=dfc)
```

Podemos ver que las desviaciones obtenidas cuando comparamos los modelos son claramente no significativas ya que tenemos unas distribuciones aproximadas de una ji-cuadrado con un grado de libertad.

```
anova(fit.c1,fit.c0)
anova(fit.c2,fit.c0)
anova(fit.c3,fit.c0)
anova(fit.c4,fit.c0)
anova(fit.c5,fit.c0)
```

7.6 Datos de conteo

En este tema se trata la situación en que la variable respuesta son datos de conteo, esto es, contamos díás de estancia en un hospital, número de supervivientes o muertos considerando su posible asociación con covariables.

7.6.1 Modelos loglineales Poisson

La variable respuesta Y es un conteo (número de defectos, conteo en una tabla de contingencia). En principio vamos a asumir que la distribución de Y es Poisson, $Y \sim Po(\mu)$, con media y varianza μ, donde $EY = varY = \mu$. La función de probabilidad es

$$f(y;\mu) = \frac{e^{-\mu}\mu^y}{y!} = e^{-\mu}\frac{1}{y!}e^{y\log\mu}$$

con $y = 0, 1, \ldots$. El parámetro natural sería $Q(\mu) = \ln\mu$ y la función enlace canónica $\eta = \ln\mu$. El modelo loglineal con una sola variable explicativa x es

$$\log\mu = \beta_0 + \beta_1 x.$$

En este modelo

$$\mu = \exp(\beta_0 + \beta_1 x) = e^{\beta_0}\left(e^{\beta_1}\right)^x.$$

Ejemplo 7.10. *Utilizamos para ilustrar los datos* `tamidata2:PRJNA218851` ↪ *. Son datos de cáncer colorrectal en donde tenemos una variable fenotípica con tres niveles.*

```
pacman::p_load(SummarizedExperiment)
data(PRJNA218851,package="tamidata2")
```

```
table(colData(PRJNA218851)[,"Stage"])
```

```
  Cancer Metastasis Normal
      18 18 18
```

Nos fijamos en el gen que ocupa la fila 1000. Construimos un `data` ↪ `.frame` *en donde incluimos la variable* `Stage` *y el conteo correspondiente a este gen.*

```
df = data.frame(count = assay(PRJNA218851)[1000,],
                Stage=colData(PRJNA218851)[,"Stage"])
head(df)
```

```
                          count Stage
SRR975551Aligned.out.sam.bam 539 Cancer
SRR975552Aligned.out.sam.bam 563 Cancer
SRR975553Aligned.out.sam.bam 1018 Cancer
SRR975554Aligned.out.sam.bam 393 Cancer
SRR975555Aligned.out.sam.bam 398 Cancer
SRR975556Aligned.out.sam.bam 672 Cancer
```

Ajustamos un modelo loglineal de Poisson.

```
fit = glm(count ~Stage, family = poisson(link = log),
          data = df)
summary(fit)
```

```

Call:
glm(formula = count ~Stage, family = poisson(link = log), data = df)

Coefficients:
              Estimate Std. Error z value
(Intercept) 6.729957 0.008146 826.14
StageMetastasis -0.306800 0.012512 -24.52
```

```
StageNormal 0.429249 0.010467 41.01
             Pr(>|z|)
(Intercept) <2e-16 ***
StageMetastasis <2e-16 ***
StageNormal <2e-16 ***
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 13236.9 on 53 degrees of freedom
Residual deviance: 8723.7 on 51 degrees of freedom
AIC: 9189.2

Number of Fisher Scoring iterations: 4
```

La desviacion nula es la desviacion para el modelo que tiene solo la constante. La desviacion residual es la desviacion del modelo que tiene la constante y las variables binarias que describen **`Stage`**. *La diferencia entre los valores tiene una distribucion ji-cuadrado con dos grados de libertad y nos permite contrastar si los coeficientes de* **`StageMetastasis`** *y* **`StageNormal`** *pueden considerarse simultáneamente nulos.*

```
fit$null.deviance - fit$deviance
```

```
[1] 4513.206
```

Podemos rechazar confortablemente la hipotesis nula. Hay un aportacion significativa **`Stage`**. *Veamos lo que nos devuelve la funcion genérica* **`summary`** *sobre un objeto de clase* **`glm`**.

```
attributes(summary(fit))
```

```
$names
 [1] "call" "terms"
 [3] "family" "deviance"
 [5] "aic" "contrasts"
 [7] "df.residual" "null.deviance"
 [9] "df.null" "iter"
[11] "deviance.resid" "coefficients"
[13] "aliased" "dispersion"
[15] "df" "cov.unscaled"
[17] "cov.scaled"

$class
[1] "summary.glm"
```

Por ejemplo, podemos extraer los coeficientes.

```
summary(fit)$coefficients
```

```
               Estimate Std. Error z value
(Intercept) 6.7299568 0.008146275 826.13916
StageMetastasis -0.3068000 0.012512160 -24.52015
StageNormal 0.4292487 0.010467369 41.00827
                  Pr(>|z|)
(Intercept) 0.000000e+00
StageMetastasis 9.006867e-133
StageNormal 0.000000e+00
```

Si nos interesa uno dado no hay mas que ver la posicion y recuperarlo.

```
summary(fit)$coefficients[1, 2]
```

```
[1] 0.008146275
```

Veamos los atributos del objeto de clase **glm**.

```
attributes(fit)
```

```
$names
 [1] "coefficients" "residuals"
 [3] "fitted.values" "effects"
 [5] "R" "rank"
 [7] "qr" "family"
 [9] "linear.predictors" "deviance"
[11] "aic" "null.deviance"
[13] "iter" "weights"
[15] "prior.weights" "df.residual"
[17] "df.null" "y"
[19] "converged" "boundary"
[21] "model" "call"
[23] "formula" "terms"
[25] "data" "offset"
[27] "control" "method"
[29] "contrasts" "xlevels"

$class
[1] "glm" "lm"
```

En particular los valores esperados vienen dados por

```
head(fit$fitted.values)
```

```
SRR975551Aligned.out.sam.bam
                    837.1111
SRR975552Aligned.out.sam.bam
                    837.1111
SRR975553Aligned.out.sam.bam
                    837.1111
SRR975554Aligned.out.sam.bam
                    837.1111
SRR975555Aligned.out.sam.bam
                    837.1111
SRR975556Aligned.out.sam.bam
                    837.1111
```

El mismo resultado lo obtenemos mediante

```
head(fitted(fit))
```

```
SRR975551Aligned.out.sam.bam
                    837.1111
SRR975552Aligned.out.sam.bam
                    837.1111
SRR975553Aligned.out.sam.bam
                    837.1111
SRR975554Aligned.out.sam.bam
                    837.1111
SRR975555Aligned.out.sam.bam
                    837.1111
SRR975556Aligned.out.sam.bam
                    837.1111
```

Los coeficientes los tenemos con

```
fit$coefficients
```

```
   (Intercept) StageMetastasis StageNormal
     6.7299568 -0.3068000 0.4292487
```

o bien con

```
coef(fit)
```

```
   (Intercept) StageMetastasis StageNormal
     6.7299568 -0.3068000 0.4292487
```

Podemos predecir la media de la respuesta para el valor de **`Stage`** *que queramos con* **`predict`**.

```
predict(fit,type = "response",
        newdata = data.frame(Stage =c("Cancer","Metastasis","Normal")))
```

```
       1 2 3
837.1111 615.9444 1285.8889
```

Sobredispersión en GLM Poisson

En una distribución de Poisson, la media y la varianza son iguales. Cuando trabajamos con conteos reales no suele ser cierta esta hipótesis. Con frecuencia la varianza es mayor que la media. A esto se le llama *sobredispersión*. No es un problema cuando Y tiene una distribución normal pues la normal tiene un parámetro que la modeliza.

Ejemplo 7.11. *Vamos a estimar la sobre dispersión en un GLM Poisson.*

```
fit1 = glm(count ~Stage, family = quasipoisson(link = log),
                    data = df)
summary(fit1)$dispersion
```

```
[1] 198.0956
```

La estimacion del parametro de dispersion no es mas que la suma de los residuos de Pearson dividida por los grados de libertad residuales.

7.6.2 GLM binomiales negativos

La densidad de la distribución binomial negativa es

$$f(y;k,\mu) = \frac{\Gamma(y+k)}{\Gamma(k)\Gamma(y+1)}\left(\frac{k}{\mu+k}\right)^k\left(1-\frac{k}{\mu+k}\right)^y$$

con $y = 0, 1, 2, \ldots$ donde k y μ son los parámetros. Se tiene que

$$E(Y) = \mu, \quad var(Y) = \mu + \mu^2/k.$$

El parámetro $1/k$ es un *parámetro de dispersión*. Si $1/k \to 0$, entonces $var(Y) \to \mu$ y la distribución binomial negativa converge a una distribución de Poisson. Con k fijo esta densidad está en la familia exponencial natural y podríamos hablar de un GLM binomial negativo.

Ejemplo 7.12.

```
library(MASS)
fit2 = glm(count ~Stage,
           family = negative.binomial(theta = 1,link = "log"),
           data = df, start = coef(fit))
summary(fit2)
```

```
Call:
glm(formula = count ~Stage, family = negative.binomial(theta = 1,
    link = "log"), data = df, start = coef(fit))

Coefficients:
              Estimate Std. Error t value
(Intercept) 6.7300 0.1195 56.334
StageMetastasis -0.3068 0.1690 -1.816
StageNormal 0.4292 0.1689 2.541
              Pr(>|t|)
(Intercept) <2e-16 ***
StageMetastasis 0.0753 .
StageNormal 0.0141 *
---
Signif. codes:
0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for Negative Binomial(1) family taken to be
    ↪ 0.2565866)

    Null deviance: 15.757 on 53 degrees of freedom
Residual deviance: 10.801 on 51 degrees of freedom
AIC: 845.31

Number of Fisher Scoring iterations: 1
```

7.6.3 Ejercicios

Ex. 13 — Repetir el análisis que hemos realizado en § 7.5.8 para el modelo dominante.

Ex. 14 — Repetir el análisis que hemos realizado en § 7.5.8 para el modelo recesivo.

Ex. 15 — Programar una función que nos permita elegir el módelo genético más significativo para cada uno de los SNPs.

Parte IV

Expresión diferencial

Capítulo 8

Expresión diferencial marginal

8.1 Introducción

Para cada uno de los genes (exones, o en general características genómicas) tenemos un valor numérico (expresión) que nos indica abundancia de copias de esta característica. En resumen abundancia de la característica en la que estamos interesados. Tenemos este valor observado para distintas muestras. De cada una de las muestras tenemos a su vez variables que las describen: tiempos, tamaños celulares, temperatura, etc. A estas variables las llamaremos en lo que sigue covariables. Estas covariables pueden ser categóricas, numéricas o tiempos de supervivencia censurados. Si la covariable tiene dos categorías entonces nos define dos grupos (por ejemplo, control y tratamiento). El caso en que queremos comparar dos grupos es el más frecuente y por ello siempre le daremos una mayor atención.

El problema que se conoce como **expresión diferencial** se puede traducir a saber si hay algún tipo de asociación entre las expresiones observadas y los valores de la covariable. Si la covariable define dos categorías entonces la pregunta de la asociación covariable-expresión se puede formular como: ¿Hay diferencias entre la expresión de genes entre dos grupos considerados? Dicho de otro modo, la expresión es distinta bajo los tratamientos que estamos considerando.

En un primer momento vamos a adoptar la aproximación gen-a-gen. Buscamos genes que se expresan diferencialmente sin atender a las interacciones que puedan existir entre los distintos genes, que las hay y las consideraremos más adelante.

Intentaremos ir planteando las cuestiones (muy) lentamente ilustrando continuamente lo que hacemos con R/Bioconductor.

8.2 Algo de notación

En lo que sigue denotaremos la matriz con los datos de expresión de los distintos genes como $\boldsymbol{y} = [y_{ij}]_{i=1,\ldots,N;j=1,\ldots,n}$ donde el valor y_{ij} nos da el nivel de expresión **observado** del gen i-ésimo en la muestra j-ésima. Asociada a la muestra j tenemos unas covariables (o varia-

ble fenotípicas) $\boldsymbol{x}_j \in \mathbb{R}^p$. [1] Como es de uso habitual en Estadística denotaremos

$$y_{\cdot j} = \sum_{i=1}^{N} y_{ij}; \quad y_{i\cdot} = \sum_{j=1}^{n} y_{ij},$$

$$\bar{y}_{\cdot j} = \sum_{i=1}^{N} \frac{y_{ij}}{N}; \quad \bar{y}_{i\cdot} = \sum_{j=1}^{n} \frac{y_{ij}}{n}.$$

En la notación previa hemos indica los valores de expresión o la covariable utilizando letras en minúsculas. Esto es habitual en Probabilidad. Sin embargo cuando consideremos los valores que observaremos **antes** de realizar la experimentación tendremos los valores aleatorios. También como es usual utilizaremos las letras en mayúsculas. De modo que $\boldsymbol{Y} = [Y_{ij}]_{i=1,\ldots,N;j=1,\ldots,n}$ denota la matriz de expresión aleatoria cuando todavía no se ha realizado la experimentación.

8.3 Fold-change

[2] Queremos comparar dos grupos. Denotamos los valores de expresión originales con x_{ij} (respectivamente y_{ij}) para la i-ésima característica en la j-ésima muestra del primer grupo (respectivamente del segundo grupo). A los logaritmos (en base 2 o log2) de los valores originales los denotamos por $u_{ij} = \log_2(x_{ij})$ y $v_{ij} = \log_2(y_{ij})$. Para cada gen, tendremos una expresión media para la i-ésima característica en cada uno de los dos grupos $\bar{x}_{i\cdot} = \sum_{j=1}^{n_1} \frac{x_{ij}}{n_1}$ para la media en el primer grupo. Similarmente definimos $\bar{y}_{i\cdot}$., $\bar{u}_{i\cdot}$., $\bar{v}_{i\cdot}$.. ¿Qué se entiende por **fold-change**? Dos son las interpretaciones de este valor. La primera ([88]) lo define como

$$FC_i^{(1)} = \frac{\bar{x}_{i\cdot}}{\bar{y}_{i\cdot}}. \tag{8.1}$$

Lo definimos como el cociente de las medias de las expresiones en la escala original. La segunda definición (que no es equivalente a la primera) es

$$FC_i^{(2)} = \bar{u}_{i\cdot} - \bar{v}_{i\cdot}, \tag{8.2}$$

tenemos la diferencia de las medias de las log2 expresiones.

[58] Que no es estadístico y que por tanto no vamos a considerar en este manual.

Un procedimiento[58] que se ha utilizado frecuentemente en la literatura de microarrays consiste en tomar el log2 del fold-change en la definición 8.1, el valor

$$\log_2 FC_i^{(1)} = \log_2 \left(\frac{\bar{x}_{i\cdot}}{\bar{y}_{i\cdot}} \right).$$

Si el módulo del cociente anterior es mayor que una constante positiva c ($c > 0$) entonces diríamos que el gen i se sobre expresa en el grupo 1 en relación con el grupo 2 ya que

$$\log_2 \left(\frac{\bar{x}_{i\cdot}}{\bar{y}_{i\cdot}} \right) \geq c.$$

[1] Por ejemplo, si estamos comparando los niveles de expresión en dos grupos (un grupo control y un grupo de enfermos por ejemplo) entonces x_j tomará el valor 1 si está en el primer grupo (control) y valor 2 si está en el segundo grupo (valor 2).

[2] Lo que comentamos en esta sección me ha costado de entender una enfermedad. Lo primero el porqué de utilizarlo y luego que hay una indefinición en el término en la literatura. Vamos a intentar aclarar qué significa antes de que se me olvide. Porque mucho interés en usarlo no tengo.

y se sigue que

$$\frac{\bar{x}_{i\cdot}}{\bar{y}_{i\cdot}} \geq 2^c.$$

Si $\log_2\left(\frac{\bar{x}_1}{\bar{x}_2}\right) < -c$ entonces tendremos que

$$\frac{\bar{x}_{i\cdot}}{\bar{y}_{i\cdot}} \leq -2^c,$$

o bien que

$$\frac{\bar{y}_{i\cdot}}{\bar{x}_{i\cdot}} \geq 2^c.$$

En este segundo caso el gen i se sobre expresa en el grupo 2 en relación con el primer grupo. El término que se utiliza es sobre regulación en el primer caso y de infra regulación en el segundo.[59] Se utiliza con mucha frecuencia. Es sencillo, entendible y fácil de usar. Pero **no** es una buena opción. Lo fundamental, *no se tiene en cuenta la variabilidad de las medias* que estamos comparando. Una opción más correcta la vemos posteriormente en este tema y es la utilización de un test de la t para comparar las medias de las dos poblaciones que estamos comparando. O versiones modificadas del t-test clásico como la propuestas en [88] o en [82] en donde esencialmente se modifica la estimación del error estándar.

[59] Up (down) regulation.

Por último una interpretación en mi opinión inadecuada de la expresión log fold-change es como nombre de los coeficientes en los ajustes lineal o lineal generalizado en procedimientos implementados en algunos paquetes. Por ejemplo, en el método limma [82] implementado en [81]. Asumen que trabajan en escala logarítmica y eso hace que bajo alguna circunstancia se pueda asimilar a alguna de las definiciones previas. No es adecuado en absoluto.

8.4 Expresión diferencial de un solo gen

Trabajamos con los datos `tamidata::gse21942`.

```
library(Biobase)
data(gse21942,package="tamidata")
```

Vamos a comparar la expresión observada de un gen para enfermos (esclerosis múltiple) y para sanos. La variable fenotípica que nos indica si la muestra corresponde a enfermo o sano es

```
y0 = pData(gse21942)[,"FactorValue..DISEASE.STATE."]
```

Pretendemos fijarnos en un gen y ver cómo comparamos los niveles de expresión. En concreto nos fijamos en el gen que ocupa la primera fila de la matriz de expresión. Esta es la información que tenemos del mismo.

```
fData(gse21942)[1,]
```

```
   PROBEID ENTREZID ENSEMBL SYMBOL
1 1007_s_at 780 ENSG00000204580 DDR1
        GO EVIDENCE ONTOLOGY
1 GO:0001558 IEA BP
```

Guardamos los niveles de expresión en *DDR1*.

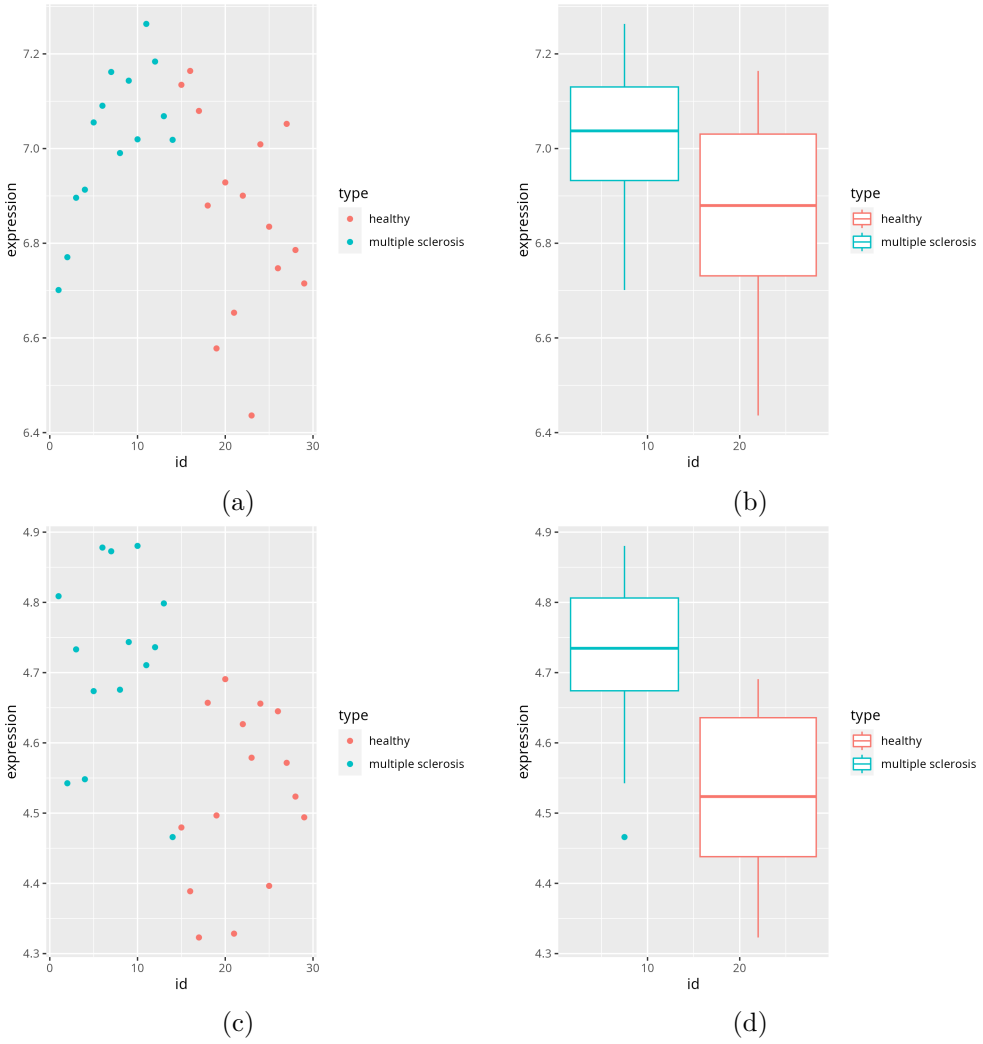


Figura 8.1: a) Perfil de expresión para el gen DDR1 utilizando casos y controles. b) Diagrama de cajas para gen DDR1 en los dos grupos. c) Perfil de expresión para el gen THRA en los dos grupos. d) Diagrama de cajas para gen DDR1 en los dos grupos.

```
DDR1 = exprs(gse21942)[1,]
```

En la figura 8.1(a) mostramos los datos para las distintas muestras (con distintos colores y caracteres distintos).

```
df=data.frame(id = 1:ncol(gse21942),expression = DDR1,type = y0)
ggplot(df,aes(x= id,y = expression,color = type)) + geom_point()
```

Sabemos que hay dos grupos de pacientes. Podemos representar dos diagramas de cajas que nos den una comparación sencilla de los niveles de expresión para los dos grupos. Lo tenemos en figura 8.1(b).

```
df=data.frame(id = 1:ncol(gse21942),expression = DDR1,type = y0)
ggplot(df,aes(x= id,y = expression,color = type)) + geom_boxplot()
```

También trabajaremos con las expresiones de `THRA` que aparece en la fila 7 de la matriz de expresión de `tamidata::gse21942`.

```
fData(gse21942)[7,]
```

```
   PROBEID ENTREZID ENSEMBL SYMBOL
332 1316_at 7067 ENSG00000126351 THRA
          GO EVIDENCE ONTOLOGY
332 GO:0000976 IEA MF
```

Guardamos los datos en `THRA`.

```
THRA = exprs(gse21942)[7,]
```

En las figuras 8.1(c) y 8.1(d) mostramos el perfil de expresión de este gen diferenciando las expresiones según el tipo de leucemia y el correspondiente diagrama de cajas.

```
df=data.frame(id = 1:ncol(gse21942),expression = THRA,type = y0)
ggplot(df,aes(x= id,y = expression,color = type)) + geom_point()
ggplot(df,aes(x= id,y = expression,color = type)) + geom_boxplot()
```

Este es el problema planteado: dos grupos de valores indicando los niveles de expresión bajo dos condiciones y la pregunta fundamental a responder es: ¿son ambos grupos de valores, ambos grupos de niveles de expresión, similares entre sí o difieren claramente uno de otro? Estamos con lo que en Estadística se conoce como la comparación de dos poblaciones: cada población corresponde con una condición.

8.5 Comparamos dos condiciones

Tenemos datos relativos a niveles de expresión bajo dos condiciones experimentales. En nuestro caso, la condición experimental me indica el tipo de leucemia. Denotamos por X el nivel de expresión aleatorio que observamos bajo la primera condición y por Y lo mismo pero con la segunda condición.

Supongamos que asumimos que ambos tienen una distribución normal con medias y varianzas posiblemente distintas. Es decir, asumimos que $X \sim N(\mu_X, \sigma_X^2)$ y que $Y \sim N(\mu_Y, \sigma_Y^2)$. También asumimos que tenemos una muestra aleatoria de X: $X_1 \ldots, X_n$ variables aleatorias independientes y con una misma distribución. Y otra muestra de Y: $Y_1 \ldots, Y_m$ variables aleatorias independientes y con la distribución. Utilizamos los datos `tamidata::gse21942`.

```
data(gse21942,package="tamidata")
```

Utilizamos `genefilter::rowttests()`.

```
tt = genefilter::rowttests(gse21942,pData(gse21942)[,"
    ↪ FactorValue..DISEASE.STATE."])
```

Veamos las primeras filas de la salida.

```
head(tt)
```

```
          statistic dm p.value
1007_s_at -2.29869931 -0.15981906 0.029492008
1053_at 3.44084102 0.20940361 0.001901206
117_at -0.08505071 -0.01110444 0.932848609
121_at -0.53362792 -0.02274832 0.597965094
1255_g_at -1.01536731 -0.04339509 0.318943716
1294_at -1.05030996 -0.07873761 0.302886126
```

Cada fila corresponde a un gen. La primera columna (`statistic`) nos muestra el estadístico t del contraste de hipótesis para la igualdad de las medias (por defecto asume varianzas iguales). La segunda columna (`dm`) nos muestra la diferencia de medias y la última columna (`p.value` ↪) nos da el p-valor del contraste.

Para cada gen tenemos pues un p-valor. ¿Podemos seguir aplicando el criterio de rechazar la hipótesis nula cuando el p-valor es inferior al nivel de significación α previamente elegido?

Si lo hacemos en este ejemplo, ¿cuándo tests mostrarían una expresión diferencial entre ambos grupos?

```
table(tt[,"p.value"] < 0.05)
```

```
FALSE TRUE
37561 17114
```

Como vemos tenemos demasiados genes que muestran expresión diferencial. Y no parece muy razonable esto. ¿Con qué problema nos estamos encontrando? El nivel de significación α es una cota superior del error tipo I cuando realizamos **un solo test**. Pero no estamos aplicando un solo test. En concreto con estos datos acabamos de realizar 3051 tests que además son dependientes entre si. Estamos rechazando muchas hipótesis nulas (no diferencia entre grupos para cada gen) cuando no deberíamos de hacerlo.

Sin embargo, cuando realizamos muchos test: ¿qué es el error tipo I? Ya no es claro cómo cuantificamos los errores que cometemos y, de hecho, tenemos que definir nuevas tasas de error para cuantificar y, sobre todo, controlar los errores que cometemos cuando realizamos *simultáneamente* muchos tests.

8.6 Ejercicios

* **Ex. 16** — Utilizando los datos `tamidata::gse1397` y el factor `pData(gse1397[,"type"]` realizar un análisis de expresión diferencial marginal.

Ex. 17 — Con los datos `multtest::golub` y utilizando las funciones `base::apply()`, `stats::sd()`, `stats::IQR()` se pide:

1. Determinar para cada gen (esto es, las expresiones de una fila dada) el rango intercuartílico.

2. Una vez hemos calculado el rango intercuartílico para gen calcular el percentil de orden 0.50 (o mediana).
3. Determinar las filas correspondientes a los genes cuyo rango intercuartílico supera el percentil que hemos calculado en el apartado anterior.
4. Determinar aquellas filas donde al menos 10 muestras de las 38 superan una expresión de 0.
5. Seleccionar en la matriz de expresión golub las filas que verifican los criterios dados en los puntos 3 y 4.

** **Ex. 18** — Utilizando el paquete [48, genefilter] se pide diseñar un filtrado no específico para los datos `multtest::golub` y que seleccione los genes atendiendo a los siguientes criterios:

1. Determinamos para cada gen el coeficiente de variación.
2. Determinamos el percentil 0.90 del los coeficientes de variación para todos los genes.
3. El coeficiente de variación ha de superar el percentil 0.9 de los coeficientes de variación observados.
4. Repetimos los puntos anteriores sustituyendo el coeficiente de variación por el nivel de expresión medio.
5. Conservamos los genes que verifican los dos criterios de selección.

** **Ex. 19** — Realizar una selección no específica de los datos [59, ALL] utilizando el paquete [48, genefilter]. Los criterios de selección son los siguientes:

1. La mediana de los niveles de expresión del gen ha de superar el percentil 0.9 de las medianas observadas para todos los genes.
2. El rango intercuartílico de los niveles de expresión del gen ha de superar el percentil 0.9 de los rangos intercuartílicos observados para todos los genes.

Capítulo 9

Comparaciones múltiples

9.1 Introducción

Empezamos con algunos comentarios bibliográficos. Esta sección se basa fundamentalmente en [30]. Sin embargo, un artículo a consultar es [76].

Supongamos que nuestras muestras corresponden a dos grupos que de un modo genérico podemos llamar casos y controles. Nuestro interés no está en evaluar si uno o dos genes concretos se expresan de un modo distinto entre las dos condiciones consideradas. Se pretende una visión global. Pretendemos responder a una pregunta como: ¿qué genes se expresan de un modo distinto (o diferencial en la jerga habitual en esta literatura) en los dos grupos que consideramos? Tenemos miles de genes: ¿cuáles de ellos tienen realmente una expresión diferencial? Y, lo importante, pretendemos responder la pregunta de modo que controlemos, de algún modo, las veces que admitimos una expresión diferencial cuando no la tiene. ¿Qué contrastes de hipótesis estamos evaluando? Si numeramos los genes con $i = 1, \ldots, N$ entonces para el i-ésimo gen estamos considerando el contraste de hipótesis siguiente:

- H_i: El gen i **no tiene** una expresión diferencial entre las condiciones consideradas.
- K_i: El gen i **tiene** una expresión diferencial entre las condiciones consideradas.

El contraste anterior se puede reformular como

- H_i: La expresión del gen i no tiene asociación con la condición.
- K_i: La expresión del gen i tiene asociación con la condición.

[1]

Estos contrastes nos lo planteamos para cada uno de los N genes que evaluamos. Denotemos por $G = \{1, \ldots, N\}$ el conjunto de hipótesis nulas que estamos evaluando. El número de hipótesis que vamos a contrastar es conocido a priori ya que corresponde con el número de

[1] De este modo, creo que englobamos de un modo más natural el caso en que consideramos asociada a las muestras una covariable que no sea necesariamente un factor experimental con dos niveles.

Hipótesis nula	No rechazadas	Rechazadas	Total
Verdadera	U	V	N_0
Falsa	T	S	$N - N_0 = N_1$
Total	N - R	R	N

Tabla 9.1: Errores tipo I y II en contraste de múltiples hipótesis

genes que estamos evaluando. Denotamos por G_0 (con $G_0 \subset G$) las hipótesis nulas que son ciertas. Denotamos por $N_0 = |G_0|$ el cardinal del conjunto G_0. Notemos que el conjunto G_0 es desconocido para nosotros. No sabemos ni cuántas ni cuáles son las hipótesis nulas ciertas. Esto supondría que conocemos qué genes se expresan diferencialmente y esto es precisamente nuestro objetivo. La situación en la en que nos encontramos viene descrita en la tabla 9.1 (de [14]).

En esta tabla conocemos N, esto es, el número de contrastes de hipótesis. Una vez hemos realizado todos los contrastes y tomado una decisión sobre si rechazamos o no cada hipótesis nula podemos **observar** R que nos indica cuántas hipótesis nulas hemos rechazado. Obviamente R es una variable aleatoria. Distintos datos nos darán distintos valores de R. Los valores de S, T, U, V son también aleatorios. Sin embargo, estas variables no son observables. La variable aleatoria V nos está dando el número (desconocido) de falsos positivos o errores tipo I (no hay expresión diferencial pero decidimos que la hay de un modo erróneo) mientras que T nos da el número de falsos negativos o error tipo II (genes que se expresan diferencialmente pero no admitimos que no lo hacen). Ambas variables indican error y son importantes.

Cuando realizamos un solo contraste el procedimiento consiste en fijar una cota al error tipo I, el nivel de significación, con un valor determinado que solemos denotar por α. Supongamos que denotamos por p_i el p-valor asociado al i-ésimo contraste. Entonces si aplicamos la regla de rechazar H_i cuando $p_i \leq \alpha$ y en otro caso aceptarla (o no rechazarla como se prefiera). Con este procedimiento sabemos que tenemos controlado el error tipo I **para el contraste i-ésimo** a un nivel α. Pero, y esto es fundamental, solamente para el ese contraste. No sabemos nada de lo que ocurre **simultáneamente** para todos los contrastes. Supongamos que tenemos un estadístico T_i que utilizamos para contrastar H_i y supongamos además que rechazamos la hipótesis nula para valores grandes de $|T_i|$[2] En este caso tendremos un valor c_α tal que rechazamos la hipótesis H_i cuando $\{|T_i| > c_\alpha\}$ y se tendrá que

$$P(|T_i| > c_\alpha | H_i) = \alpha.$$

Esto es, la probabilidad de rechazar cuando es cierta la hipótesis nula es α. O la probabilidad de no rechazar cuando es cierta la hipótesis nula H_i será de $1 - \alpha$, es decir,

$$P(|T_i| \leq c_\alpha | H_i) = 1 - P(|T_i| > c_\alpha | H_i) = 1 - \alpha.$$

Cuando realizamos simultáneamente muchos contrastes nos podemos equivocar rechazando la hipótesis nula cuando no lo es en más de una ocasión. De hecho, el número de hipótesis nulas falsamente rechazadas

[2]Una situación así la tenemos cuando observamos las muestras bajo dos condiciones distintas y utilizamos un test de la t de comparación de medias.

es una variable aleatoria que denotamos por V y nos da el número de veces que rechazamos la hipótesis nula erróneamente. Nuestro interés es que la variable aleatoria V tome valores pequeños con probabilidades altas.

Las distintas medidas de error que se utilizan esencialmente cuantifican valores asociados a las variables que hemos definido en la tabla 9.1. Se habla de *tasas de error tipo I* abandonando la expresión de error tipo I utilizada con un solo test. Son extensiones del único error tipo I que se nos plantea cuando realizamos un solo contraste.

FWER: Tasa de error global [3] Esta medida de error es la que tiene una mayor tradición en Estadística. Quizás es la forma natural de pensar: no quiero equivocarme nunca rechazando una hipótesis nula cuando es cierta. La definimos como

$$FWER = P(V > 0) = P(V \geq 1).$$

Sin duda es la medida ideal. Ningún error. Esto tiene un pago. Es un criterio muy exigente si tenemos un número de hipótesis N muy grande. Notemos que nos fijamos en cometer al menos un error cuando con frecuencia tendremos decenas de miles de contrastes. Esto puede parecer algo bueno pero también estaremos construyendo tests que procurarán no rechazar una hipótesis nula salvo que tengan una evidencia contra ella muy grande. Procedimientos para contrastar miles de hipótesis nula y que hagan pequeña esta tasa de error tenderán a ser muy conservadores. Como (mala) contrapartida muchos genes que tendrán una expresión diferencial realmente no serán detectados. O dicho de un modo más técnico perderemos potencia en los contrastes donde la potencia es la probabilidad de rechazar H_i cuando realmente es falsa.

FDR: Tasa de falsamente rechazados [4] Se propuso en [14]. De alguna manera se vió que la tasa FWER era demasiado exigente y se trataba de conseguir procedimientos más potentes sin que por ello dejaramos de tener una tasa de error razonable controlada. Definimos la variable aleatoria $Q = V/R$ si $R > 0$ y $Q = 0$ en otro caso. Esta variable simplemente nos da la proporción de test que rechazamos erróneamente. Rechazamos R de los cuales V no debieran de ser rechazados, por tanto, nuestra proporción de hipótesis nulas falsamente rechazadas es Q. Tenemos el problema de si no rechazamos ninguna hipótesis. En este caso tendremos $0/0$. Lo que se hace es definir el cociente como cero. Otra vez, Q es algo aleatorio, es una proporción aleatoria. ¿Qué queremos controlar de este valor aleatorio? La conocida como tasa de falsamente rechazados o *false discovery rate* denotada como FDR y que se define como

$$\begin{aligned} FDR = E(Q) = \\ E\left(\frac{V}{R}\middle| R > 0\right) P(R > 0) + 0 \times P(R = 0) = \\ E\left(\frac{V}{R}\middle| R > 0\right) P(R > 0), \end{aligned} \quad (9.1)$$

[3] Familywise error rate.
[4] False discovery rate.

que nos da la proporción esperada de hipótesis erróneamente rechazadas entre aquellas hipótesis que hemos rechazado. A la tasa FDR la hemos llamada tasa de falsamente rechazados. Hay una cierta confusión con la tasa de falsos positivos. Esta tasa sería la proporción de hipótesis nulas que son ciertas y que son rechazadas. Esto es, sería el valor medio de la variable V/N_0. Por ejemplo, si tenemos una FDR de 5% entonces el 5% de las hipótesis nulas que rechazamos son realmente ciertas. Si tenemos una tasa de falsos positivos del 5% entonces una media del 5% de las hipótesis nulas ciertas son erróneamente rechazadas. Si tenemos p-valores p_i y aplicamos simplemente la regla de rechazar H_i cuando $p_i \leq \alpha$ entonces tenemos una tasa de falsos positivos α.

pFDR: Tasa de falsamente rechazados modificada [5] Es una modificación de la anterior. Se define como

$$pFDR = E\left(\frac{V}{R}|R > 0\right).$$

Cuando tenemos un gran número de hipótesis nulas N entonces la probabilidad de que rechacemos al menos una es prácticamente segura, es decir, $P(R > 0) \approx 1$ y por lo tanto ambas tasas de error son también prácticamente iguales $FDR \approx pFDR$. Definimos la tasa pFDR porque está muy relacionada con el concepto de q-valor que definimos más adelante.

En lo que sigue vamos a considerar distintos procedimientos para contrastar muchas hipótesis. Un procedimiento concreto se dice que controla una tasa de error tipo I al nivel α cuando su tasa de error es menor o igual que α cuando aplicamos el procedimiento para producir una lista de R hipótesis nulas rechazadas o de genes declarados como significativos. La tasa de error que utilizaremos fundamentalmente en lo que sigue será la tasa FDR.

9.2 Relación entre las tasas de error tipo I

En general se verifican las siguientes desigualdades:

$$PCER \leq FDR \leq FWER.$$

Esto significa que si un procedimiento controla la FWER entonces tenderá a rechazar menos hipótesis nulas, tenderá a ser más conservador por tanto y, posiblemente, nos estaremos perdiendo genes que tienen expresión diferencial y no lo apreciamos.

La aproximación clásica al problema de las contrastes múltiples se basa en el control fuerte de la FWER. Una aproximación más reciente propuesta en [14] consiste en controlar la FDR en un sentido débil.

[5] Positive false discovery rate.

9.3 p valores y p valores ajustados

Para cada contraste H_i tendremos un p-valor p_i. En el test de la t para comparar medias tenemos

$$p_i = P(|T_i| \geq t_i | H_i).$$

donde t_i es el i-ésimo estadístico observado. A los p-valores p_i los llamaremos p-valores originales. Podemos contrastar el contraste H_i con un nivel de significación α_i rechazando la hipótesis nula H_i si $p_i \leq \alpha_i$ y no rechazando en otro caso. Cuando consideramos simultáneamente todos los tests los valores α_i serán distintos y por ello tendríamos que ir comprobando si la desigualdad $p_i \leq \alpha_i$ se verifica o no. La idea del p-valor ajustado es transformar el p-valor p_i en otro valor $\tilde{p}_i$, el i-ésimo p-valor ajustado, de modo que sean equivalentes: $p_i \leq \alpha_i$ y $\tilde{p}_i \leq \alpha$ siendo α el valor que especificamos para controlar alguna de las tasas de error tipo I.

9.4 Métodos que controlan la FWER

Veremos procedimientos que actúan en un solo paso y procedimientos por pasos bien de bajada o de subida.

9.4.1 Los métodos en un solo paso

Veamos el ejemplo más conocido, es el método de Bonferroni. Supongamos que tenemos el p-valor p_i asociado al contraste H_i. Si solamente estuvieramos contrastando la hipótesis H_i entonces rechazamos con un error tipo I α si $p_i < \alpha$ y aceptamos o no rechazamos en otro caso. La modificación de Bonferroni es simple. Ahora tenemos en cuenta que hay m contrastes y rechazamos H_i si

$$p_i \leq \frac{\alpha}{N} \tag{9.2}$$

o, equivalentemente, si

$$Np_i \leq \alpha. \tag{9.3}$$

Si tenemos en cuenta que $0 < \alpha \leq 1$ entonces la desigualdad 9.3 es equivalente a que

$$\min\{Np_i, 1\} \leq \alpha. \tag{9.4}$$

De este modo el p-valor ajustado será $\tilde{p}_i = \min\{Np_i, 1\}$.

Método de Bonferroni En este método rechazamos la hipótesis nula H_i si el correspondiente p-valor sin ajustar es menor o igual a $\frac{\alpha}{N}$. El p-valor ajustado sería:

$$\tilde{p}_i = \min\{Np_i, 1\}$$

En general, los métodos de un solo paso son simples de implementar pero tiende a ser conservadores, esto es, tienden a no rechazar la hipótesis nula. No son pues muy potentes en el sentido de detectar genes con expresión diferencial. Los métodos por pasos son más potentes.

9.5 Métodos que controlan el FDR

Los métodos de la sección anterior controlan el FWER. En resumen estamos controlando el no cometer ningún error tipo I (admitir un gen con expresión diferencial cuando no la tiene). Esto produce procedimientos conservadores. En [14] se propuso la idea de que cuando contrastamos muchas hipótesis podemos tolerar algunos errores tipo I, siempre que el número de estos errores sea pequeño en relación con el número de hipótesis que se rechazan. Esta es la idea de controlar el FDR. Dos son los procedimientos que vamos a ver para controlar esta tasa.

Benjamini y Hochberg Este procedimiento fue propuesto en [14]. Siendo $p_{r_1} \leq \ldots \leq p_{r_N}$ son los p-valores originales ordenados entonces consideramos

$$i^* = \max\{i : p_{r_i} \leq \frac{i}{N}\alpha\}$$

y rechazamos H_{r_i} para $i = 1, \ldots, i^*$. Si no existe i^* entonces no rechazamos ninguna hipótesis. Los p-valores ajustados se definen como

$$\tilde{p}_{r_i} = \min_{k=i,\ldots,N} \left\{ \min \left\{ \frac{N}{k} p_{r_k}, 1 \right\} \right\}.$$

Benjamini y Yekutieli Propuesto en [15]. Como en el anterior, $p_{r_1} \leq \ldots \leq p_{r_N}$ son los p-valores originales ordenados. En este caso los p-valores ajustados se definen como

$$\tilde{p}_{r_i} = \min_{k=i,\ldots,N} \left\{ \min \left\{ \frac{m \sum_{j=1}^{N} 1/j}{k} p_{r_k}, 1 \right\} \right\}.$$

9.6 Utilizando genefilter y p.adjust

No vamos a realizar en principio ninguna selección no específica y nos planteamos trabajar con todos los genes.

9.6.1 Cálculo de p-valores ajustados

Calculamos los estadísticos t (test de la t con varianzas distintas) y los p-valores asociados.

Utilizamos los datos tamidata::gse1397.

```
data(gse1397,package ="tamidata")
eset = gse1397; y = pData(gse1397)[,"type"]
```

Aplicamos los t-tests para cada gen.

```
tt = genefilter::rowttests(eset,y)
```

Los p-valores originales los guardamos en `p0`.

```
p0 = tt[,"p.value"]
```

Supongamos que vamos a utilizar el método de Benjamini-Hochberg. Entonces pasamos de los p-valores originales (raw p-values) a los p-valores ajustados con la función `stats::p.adjust()`.

```
p.BH = p.adjust(p0, method = "BH")
```

Podemos incorporar los p-valores ajustados al **data.frame** original.

```
tt1 = data.frame(tt,p.BH)
```

9.6.2 Genes con expresión diferencial

Lo primero es fijar una tasa de error α. En concreto podemos considerar el valor $\alpha = 0.05$.

```
alpha = 0.05
```

Consideremos los p-valores ajustados utilizando Benjamini-Hochberg manteniendo el orden original de la matriz de expresión. Observad que la segunda columna tiene los p-valores ajustados por el método Benjamini-Hochberg.

```
p1 = p.adjust(p0, "BH")
```

Los genes significativos, esto es, aquellos que tienen un p-valor ajustado menor a la tasa especificada ocupan las siguientes filas de la matriz de expresión.

```
(significativos.BH = which(p.BH < alpha))
```

```
[1] 346 1170 1853 6303 7889
```

¿Cuántos eran significativos con los p-valores originales?

```
significativos.p0 = which(p0 < alpha)
length(significativos.p0)
```

```
[1] 902
```

9.7 Ejercicios

* **Ex. 20** — Utilizando los datos `tamidata::gse20986` se pide:

1. Seleccionar las muestras correspondientes a iris y huvec.
2. Aplicamos, a cada gen, un test de la t (asumiendo una misma varianza). Obtener los t-estadísticos y los p-valores correspondientes. Utilizad `genefilter::rowttests()`.
3. Utilizando el método de corrección de Benjamini-Hochberg obtener los p-valores ajustados.[60]
4. Si utilizamos un valor de FDR igual a 0.05: ¿qué genes declaramos como significativos?
5. Repetir los apartados anteriores comparando los muestras correspondientes a retina y huvec.
6. Repetir los apartados anteriores comparando las muestras correspondientes a coroides y huvec.
7. Utilizando la función `base::intersect()` y la función `Biobase ↪ ::featureNames()` determinar los genes comunes a cada par de comparaciones y los comunes a las tres comparaciones.

[60] Utilizad [73, multtest].

7.* **Ex. 21** — Vamos a utilizar los datos `tamidata::gse1397`. Vamos a realizar un estudio de expresión diferencial marginal comparando las personas con y sin la trisomía del cromosoma 21.

1. Aplicamos, a cada gen, un test de la t (asumiendo una misma varianza). Obtener los t-estadísticos y los p-valores correspondientes.
2. Si aplicamos la regla de decisión consistente en admitir una expresión diferencial cuando el p-valor es menor que $\alpha = 0.05$: ¿cuántos y qué genes son declarados significativos?
3. Utilizando los métodos de corrección de Bonferroni y de Benjamini-Hochberg obtener los p-valores ajustados para cada uno de los procedimientos.
4. Representar, en un mismo dibujo, los p-valores originales y los ajustados obtenidos en el paso anterior. Utilizad tipos de línea y colores diferentes para cada conjunto de p-valores.
5. Si utilizamos un valor de FDR igual a 0.05: ¿qué genes declaramos como significativos utilizando la corrección de Benjamini-Hochberg? ¿Y utilizando la corrección de Bonferroni?
6. ¿Todos los significativos según Bonferroni lo son con Benjamini-Hochberg? Comparar los grupos de genes significativos utilizando las funciones `intersect()` y `Biobase::featureNames()`.

Capítulo 10

Expresión diferencial con respuesta continua

Aunque inicialmente el tipo de dato más habitual para análisis de transcripción eran los DNA microarrays con el tiempo no es así. Sin embargo, la metodología de análisis que se ha desarrollado para este tipo de análisis es aplicable, con mínimas variaciones, a cualquier tipo de dato ómico donde la respuesta, la variable de interés, es de tipo continuo, esto es, un número real. De este tipos son los DNA microarrays, los de proteínas, los datos de metilación, etc. Obviamente según la técnica cambia el preprocesado que realizamos pero el tratamieno estadístico posterior es el mismo. En esencia el problema que vamos a abordar consiste en considerar las variables fenotípicas (o metadatos) como las variables predictoras. La variable respuesta para cada sonda (gen, proteína, etc.) serán las expresiones cuantificadas (correspondiendo a una fila de la matriz de expresión). Vamos a ajustar un modelo lineal para cada perfil de expresión. El problema es que el número de observaciones es pequeño (número de muestras que coincide con el número de columnas de la matriz de expresión). Esto supone que la estimación que hacemos de la varianza del error aleatorio es mala. ¿Cómo mejorar esta estimación? Agregando, basándonos en algún modelo estocástico, los estimadores de los distintos ajustes correspondiendo con las distintas sondas. Una idea sencilla pero muy efectiva.

10.1 Limma

Hemos estudiado la posible asociación entre el perfil de expresión de un gen y las variables fenotípicas. Para ello hemos seleccionado una sonda y ajustado un modelo lineal. Incluso cuando comparamos dos grupos (caso frente a control, cepa mutante frente a salvaje) estamos ajustando un modelo lineal. Pero tenemos muchas sondas y muchos modelos a ajustar y cada uno de ellos con poca muestra.

En § 5 damos una breve introducción a modelos lineales que se recomienda leer antes de este capítulo. En los ejemplos que tratamos en § 5 se elige una sola fila de la matriz de expresión. Esta fila constituye la variable respuesta y consideramos como variables predictoras en el modelo lineal que ajustamos las variables fenotípicas.

El paquete básico en esta sección es [81, limma] siendo su esplén-

dida viñeta, en sí misma, la referencia básica.

10.1.1 El modelo

La referencia original a consultar es [82].

Se ajusta un modelo lineal para cada fila de la matriz de expresión. Obviamente las muestras son condicionalmente independientes. muestras.

Con datos de microarrays de DNA asumimos que se ha preprocesado corrigiendo fondo y normalizado y que los datos están en escala logaritmo en base 2 o $\log_2$ de la expresión original. Como es habitual la Estadística suele empezar en ómicos con datos preprocesados. En las siguientes secciones vamos a considerar distintos diseños experimentales y los analizaremos con [81, limma]. Este paquete es uno de los fundamentales en el proyecto Bioconductor y de los más antiguos.

10.2 Limma aplicado a gse25171

Este ejemplo es utilizado en § 5 para ilustrar cómo ajustar un modelo lineal. Hacemos una presentación más ligada al desarrollo teórico del modelo limma. Pretendemos evaluar cómo influye la presencia de fosfatos en la arabidopsis. Leemos los datos `tamidata2::gse25171`

```
pacman::p_load(Biobase)
data(gse25171,package="tamidata2")
```

Nuestras variables fenotípicas son

```
head(pData(gse25171),n=2)
```

```
              time time2 Pi replication
GSM618324.CEL.gz 0 Short Treatment 1
GSM618325.CEL.gz 0 Short Control 2
```

Vamos a plantearnos cómo pueden influir el tiempo de observación de la muestra (**`time`** y la presencia o no de fósforo `Pi` en la expresión de cada una de las sondas que tenemos en el microarray. Con objeto de ilustrar vamos a ir considerando modelos cada vez más complejos.

Empezamos con un modelo que considera simplemente el tiempo. Para ello hemos de construir previamente la matriz de modelo.

```
design = model.matrix(~ pData(gse25171)[,"time"])
```

Si vemos las primeras filas de dicha matriz

```
head(design)
```

```
  (Intercept) pData(gse25171)[, "time"]
1 1 0
2 1 0
3 1 1
4 1 1
5 1 6
6 1 6
```

podemos ver que la primera columna nos da la constante del modelo mientras que la segunda columna nos da el tiempo de observación. Podemos ver que, a diferencia del uso habitual de **`lm()`** no indicamos la variable respuesta ya que cada fila será la variable respuesta en cada uno de los ajustes lineales que aplicamos. Cambiamos los nombres de la columnas por razones estéticas.

```
colnames(design) = c("constante","time")
```

Vamos a ajustar todos los modelos lineales que podemos construir utilizando siempre las mismas variables predictoras (tenemos una misma matriz de modelo previamente construida) y donde la variable respuesta va cambiando correspondiendo a cada una de las filas de la matriz de expresión. La función está preparada para trabajar con `ExpressionSet`.

```
fit = limma::lmFit(gse25171,design)
```

Podemos el vector de coeficientes para cada uno de los ajustes con cualquiera de las siguientes opciones (vemos que están definidos los métodos **`coefficients`** y **`coef`**).

```
fit$coefficients
coef(fit)
coefficients(fit)
```

Vemos los primeros coeficientes y errores estándar estimados.

```
head(coef(fit))
```

```
        constante time
244901_at 5.086076 0.003844458
244902_at 4.843428 -0.005471131
244903_at 6.909896 0.009452352
244904_at 5.439306 -0.001831236
244905_at 4.038044 0.001469313
244906_at 6.249748 0.001488755
```

```
head(fit$sigma)
```

```
244901_at 244902_at 244903_at 244904_at 244905_at
0.2840462 0.2291651 0.4448048 0.3357318 0.1306061
244906_at
0.2972859
```

Es interesante ver un estimador de densidad (figura 10.1) de los distintos errores estándar observados, los valores s_i definidos arriba.

```
pacman::p_load(ggplot2)
df = data.frame(sigma = fit$sigma^2)
p = ggplot(df,aes(x=sigma)) + geom_density()
```

Realmente la hipótesis distribucional era que $1/\sigma_i^2$ multiplicada por una constante sigue una distribución ji-cuadrado. En la figura 10.2 mostramos un estimador de la densidad de $1/s_i^2$.

```
pacman::p_load(ggplot2)
df = data.frame(sigma = 1/fit$sigma^2)
p = ggplot(df,aes(x=sigma)) + geom_density()
```

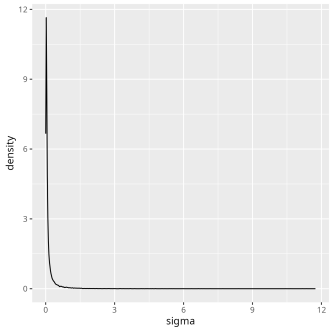
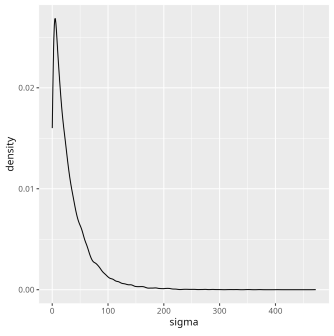


Figura 10.1: Errores estándar estimados para las distintas sondas.

En principio, podríamos simplemente contrastar, sin utilizar el modelo limma, si cada uno de los coeficientes estimados para el predictor **`time`** puede ser considerado nulo como hacemos en § 5.2 ya que realmente estamos realizando un ajuste de regresión lineal simple para cada sonda.

```
t0 = coef(fit)[,2] / fit$sigma
```



Estos valores tienen, aproximadamente, una distribución `t` de Student con $n-2$ grados de libertad. Si nos fijamos en el primer coeficiente, el p-valor correspondiente al test de si podemos considerar dicho coeficiente nulo sería dos veces el área de la cola (la distribución t es simétrica). Lo tenemos con

```
2*(1-pt(abs(t0[1]),df = ncol(gse25171)-2))
```

```
244901_at
0.9893233
```

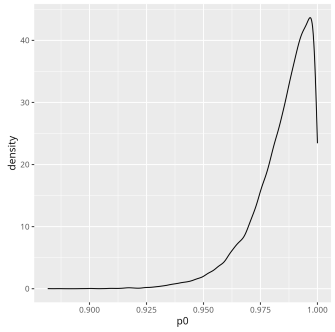


Figura 10.3: p-valores correspondiente al coeficiente del tiempo en un modelo que considera la constante y dicha variable. Todos son p-valores muy grandes.

Todos los p-valores los calculamos con

```
p0 = 2*(1-pt(abs(t0),df = ncol(gse25171)-2))
```

Un diagrama de densidad no viene mal.

```
pacman::p_load(ggplot2)
df = data.frame(p0)
p = ggplot(df,aes(x=p0)) + geom_density()
```

Podemos ver en la figura 10.3 que son p-valores muy grandes. ¿Cuántos son menores que 0.05?

```
table(p0 <0.05)
```

```

FALSE
22746
```

Como vemos ninguno lo es. No tenemos ningún test significativo.

Aplicamos ahora el método limma.

```
fit1 = limma::eBayes(fit)
```

Tenemos los t-estadísticos moderados con

```
head(fit1$t)
```

```
         constante time
244901_at 70.20698 0.6569507
244902_at 81.97642 -1.1463402
244903_at 61.65966 1.0441666
244904_at 63.89556 -0.2663003
244905_at 112.93409 0.5087080
244906_at 82.57484 0.2435055
```

Los p-valores con

```
head(fit1$p.value)
```

```
           constante time
244901_at 1.901507e-29 0.5174219
244902_at 4.564429e-31 0.2628810
244903_at 4.305353e-28 0.3067434
244904_at 1.829925e-28 0.7922674
244905_at 2.024017e-34 0.6155709
244906_at 3.831131e-31 0.8096694
```

Los valores B o logaritmo natural del cociente de odds a posteriori.

```
head(fit1$lods)
```

```
        constante time
244901_at 57.59049 -7.604303
244902_at 61.33907 -7.161476
244903_at 54.42776 -7.272309
244904_at 55.29720 -7.789878
244905_at 68.93538 -7.692872
244906_at 61.51406 -7.795904
```

El valor estimado de s_o^2 y de d_0.

```
fit1$s2.prior
```

```
[1] 0.03494128
```

```
fit1$df.prior
```

```
[1] 2.161468
```

Tenemos los estimadores de los errores estándar a posteriori.

```
head(fit1$s2.post)
```

```
 244901_at 244902_at 244903_at 244904_at
0.07659027 0.05094436 0.18327749 0.10575819
 244905_at 244906_at
0.01865779 0.08359841
```

La función `limma::topTable()` nos hace un resumen de los ajustes en donde ordena (podemos indicar que no lo haga con distintos criterios o que no lo haga con el argumento **`sort.by`**) las sondas por su p-valor original o, equivalentemente, por el p-valor ajustado.

```
limma::topTable(fit1,coef=2,adjust ="BH",number = 3)
```

```
          PROBEID ENTREZID GO EVIDENCE
261892_at 261892_at 844423 GO:0000976 IDA
246253_at 246253_at 829880 GO:0000976 IPI
262590_at 262590_at 838073 GO:0004842 IDA
        ONTOLOGY TAIR logFC
261892_at MF AT1G80840 -0.13307030
246253_at MF AT4G37260 -0.04069718
262590_at MF AT1G15100 0.04440937
          AveExpr t P.Value
261892_at 6.937141 -7.447518 1.048025e-07
246253_at 8.829745 -7.002851 2.947146e-07
262590_at 10.156525 6.418349 1.191073e-06
          adj.P.Val B
261892_at 0.002383838 7.132046
246253_at 0.003351789 6.078721
262590_at 0.007960638 4.658987
```

Vemos en la salida anterior los descriptores que hemos incluido en el slot `ExpressionSet::fData`. Tenemos el coeficiente estimado en el ajuste (`logFC`)[61], luego nos incluye la expresión media (en logaritmo en base 2), el valor del estadístico del test de la t moderado, el p-valor original, el p-valor ajustado según el método que elijamos (por defecto, Benjamini-Hochberg) y el valor estimado del logaritmo de los odds a posteriori (**??**). ¿Cuántas sondas tienen un coeficiente significativamente no nulo?

[61] De verdad que esto no induce más a confusión.

```
padj = limma::topTable(fit1,coef=2,adjust ="BH",
                       number=nrow(gse25171))[,"adj.P.Val"]
table(padj < 0.05)
```

```
FALSE TRUE
22707 39
```

Supongamos que ahora consideramos un modelo en que incorporamos la variable `Pi` de caracter categórico (binario). Consideramos un modelo con los efectos principales.

```
design = model.matrix(~ pData(gse25171)[,"time"] +
                        pData(gse25171)[,"Pi"])
colnames(design) = c("constante","time","Pi")
head(design)
```

```
  constante time Pi
1 1 0 1
2 1 0 0
3 1 1 1
4 1 1 0
5 1 6 1
6 1 6 0
```

En la matriz de modelo hemos añadido una columna en donde recogemos la presencia (1) o no (0) de fosfatos. Ajustamos los modelos.

```
fit = limma::lmFit(gse25171,design)
fit1 = limma::eBayes(fit)
```

Ahora tenemos tres coeficientes correspondiendo con cada una de las columnas de la matriz de modelo y podemos evaluar aquellas sondas para las cuales el correspondiente coeficiente es significativamente no nulo. Observemos que no ordenamos (**`sort.by="none"`**) ni seleccionamos (**`number=nrow(gse25171)`**) las sondas. Primero evaluamos si los coeficientes correspondientes a la columna 2 (**`coef=2`** son nulos. Notemos que la columna 2 corresponde con la variable **`time`**.

```
tt2 = limma::topTable(fit1,coef=2,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
```

Ahora evaluamos si los coeficientes correspondientes a la columna 3 (**`coef=3`** son nulos. Notemos que la columna 3 corresponde con la variable `Pi`.

```
tt3 = limma::topTable(fit1,coef=3,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
```

¿Cualés y cuántas sondas tienen el coeficiente correspondiente a **`time`** significativamente no nulo ajustando por el método de Benjamini-Hochberg?

```
tt2.row = which(tt2[,"adj.P.Val"] < .05)
length(tt2.row)
```

```
[1] 146
```

Es importante notar que el modelo que tenemos ahora no es el mismo que cuando tenemos solamente **`time`** y por ello los resultados no tienen porqué ser los mismos. Dicho con propiedad, estamos ajustando por las variables **`time`** y `Pi`. Y ahora: ¿Cualés y cuántas sondas tienen el coeficiente correspondiente a `Pi` significativamente no nulo ajustando por el método de Benjamini-Hochberg?

```
tt3.row = which(tt3[,"adj.P.Val"] < .05)
length(tt3.row)
```

```
[1] 34
```

¿Y cuántas sondas tienen coeficiente significativamente no nulo tanto para una como para la otra variable?

```
intersect(tt2.row,tt3.row)
```

```
 [1] 1118 2308 3025 3894 5776 6740 8384
 [8] 9258 12851 13892 14979 16243 16293 16992
[15] 17038 20376 21552
```

¿Qué sondas son?

```
tt2[intersect(tt2.row,tt3.row),"PROBEID"]
```

```
 [1] "246018_at" "247208_at" "247925_at"
 [4] "248794_at" "250676_at" "251640_at"
 [7] "253284_at" "254158_at" "257751_at"
[10] "258792_at" "259879_at" "261143_at"
[13] "261193_at" "261892_at" "261938_at"
[16] "265276_at" "266452_at"
```

¿A qué genes corresponden utilizando su código `ENTREZID`?

```
tt2[intersect(tt2.row,tt3.row),"ENTREZID"]
```

```
 [1] "830934" "836610" "835860" "834768"
 [5] "830520" "28719408" "829563" "828540"
 [9] "821400" "819622" "843998" "838565"
[13] "28717292" "844423" "838857" "817388"
[17] "818933"
```

¿Hay interacción entre las variables predictoras **`time`** y `Pi`? Observemos que sustituimos `+` por `*`.

```
design = model.matrix(~ pData(gse25171)[,"time"] *
                        pData(gse25171)[,"Pi"])
colnames(design) = c("constante","time","Pi","time:Pi")
head(design)
```

```
  constante time Pi time:Pi
1 1 0 1 0
2 1 0 0 0
3 1 1 1 1
4 1 1 0 0
5 1 6 1 6
6 1 6 0 0
```

La nueva columna de la matriz de modelo se define como el producto de las columnas 2 y 3 correspondientes a las variables originales **`time`** y `Pi`. El modelo es distinto, los coeficientes estimados para la nueva variable y para las ya existentes son distintos y no necesariamente tendremos las mismas sondas. Este sería el modelo correcto puesto que incorpora la posible interacción. ¿Qué sondas tienen interacciones significativas?

```
fit = limma::lmFit(gse25171,design)
fit1 = limma::eBayes(fit)
tt2 = limma::topTable(fit1,coef=2,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt3 = limma::topTable(fit1,coef=3,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt4 = limma::topTable(fit1,coef=4,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
```

Ahora podemos evaluar los que tienen interacciones significativas.

```
tt4.row = which(tt4[,"adj.P.Val"] < .05)
tt4[tt4.row,c("PROBEID","ENTREZID")]
```

```
          PROBEID ENTREZID
246071_at 246071_at 832137
246576_at 246576_at 840052
248545_at 248545_at 835091
253386_at 253386_at 829440
263851_at 263851_at <NA>
266132_at 266132_at 819120
267611_at 267611_at 817207
```

Cuando una sonda tiene una interacción positiva se suele asumir que los efectos principales no se deben eliminar del modelo. Si hay interacción quiere decir que influyen significativamente y además de un modo distinto.

Hemos utilizado la variable **`time`** como numérica (y lo es). Sin embargo, podemos ver que en las variables fenotípicas hay una variable `time2` categórica con dos niveles.

```
pData(gse25171)[,"time2"]
```

```
 [1] Short Short Short Short Medium Medium
 [7] Medium Medium Short Short Short Short
[13] Medium Medium Medium Medium Short Short
[19] Short Short Medium Medium Medium Medium
Levels: Short Medium
```

Vamos a realizar un análisis en donde se considera en lugar del tiempo original esta nueva variable. Ajustamos directamente el modelo con las dos variables categóricas, `time2` y `Pi` y con la posible interacción.

```
design = model.matrix(~ pData(gse25171)[,"time2"] *
                        pData(gse25171)[,"Pi"])
colnames(design) = c("constante","time2","Pi","time2:Pi")
head(design)
```

```
  constante time2 Pi time2:Pi
1 1 0 1 0
2 1 0 0 0
3 1 0 1 0
4 1 0 0 0
5 1 1 1 1
6 1 1 0 0
```

Podemos repetir el análisis previo en el cual analizábamos la interacción.

```
fit = limma::lmFit(gse25171,design)
fit1 = limma::eBayes(fit)
tt2 = limma::topTable(fit1,coef=2,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt3 = limma::topTable(fit1,coef=3,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt4 = limma::topTable(fit1,coef=4,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt4.row = which(tt4[,"adj.P.Val"] < .05)
tt4[tt4.row,c("PROBEID","ENTREZID")]
```

```
          PROBEID ENTREZID
245571_at 245571_at 827120
245579_at 245579_at 827137
247477_at 247477_at 836355
```

```
247965_at 247965_at 835755
248618_at 248618_at 835024
248733_at 248733_at <NA>
249847_at 249847_at 832385
250172_at 250172_at 831283
251232_at 251232_at 825453
251770_at 251770_at 824763
251914_at 251914_at 824560
252312_at 252312_at 824100
254294_at 254294_at 828406
254313_at 254313_at 828341
255782_at 255782_at 838573
256319_at 256319_at 840493
256783_at 256783_at 820572
257007_at 257007_at 820638
257686_at 257686_at 820462
257697_at 257697_at 820452
258054_at 258054_at 820870
258807_at 258807_at 819558
259106_at 259106_at 28718845
259173_at 259173_at 821201
259351_at 259351_at 819677
259828_at 259828_at 843554
260203_at 260203_at 841722
260386_at 260386_at 843739
260784_at 260784_at 837127
261466_at 261466_at 837282
261745_at 261745_at 837371
261772_at 261772_at 843957
263680_at 263680_at 839583
263876_at 263876_at 816724
264200_at 264200_at 838871
264672_at 264672_at 837504
264752_at 264752_at 838909
265049_at 265049_at 841635
265707_at 265707_at 814868
266743_at 266743_at 814828
267457_at 267457_at 817946
```

Podemos ir evaluando aquellos que son significativos en la interacción (y consecuentemente asumimos que los efectos principales de cada variable también lo son). Podemos evaluar aquellas sondas que no tienen interacción significativa pero sí que significativa alguna de los efectos principales. Por ejemplo, no hay interacción significativa pero sí lo es el efecto de `time2`.

```
sel = which(tt4[,"adj.P.Val"] ≥.05 & tt2[,"adj.P.Val"] < .05)
length(sel)
head(tt4[sel,c("PROBEID","ENTREZID")])
```

No hay interacción significativa pero sí lo es el efecto de `Pi`.

```
sel = which(tt4[,"adj.P.Val"] ≥.05 & tt3[,"adj.P.Val"] < .05)
length(sel)
```

```
[1] 0
```

Esta forma de plantear el análisis supone que sabemos interpretar que la presencia de la constante hace que cada uno de los coeficientes se ha de interpretar como la modificación respecto de un grupo control que en este caso sería cuando las variables categóricas toman el primer nivel.

```
levels(pData(gse25171)[,"time2"])
```

```
[1] "Short" "Medium"
```

```
levels(pData(gse25171)[,"Pi"])
```

```
[1] "Control" "Treatment"
```

Por tanto, el grupo de referencia corresponde a la situación en que `time2` es `Short` y `Pi` es `Control`.

Vamos a adoptar una aproximación basada en constrates, esto es, combinaciones lineales de los coeficientes para realizar el mismo análisis. Primero vamos a construir una variable categórica que combina ambas variables categóricas.

```
time2Pi = vector("list",ncol(gse25171))
for(i in seq_along(time2Pi))
  time2Pi[[i]] = paste0(pData(gse25171)[,"time2"][i],
                        pData(gse25171)[,"Pi"][i])
(time2Pi = factor(unlist(time2Pi)))
```

```
 [1] ShortTreatment ShortControl
 [3] ShortTreatment ShortControl
 [5] MediumTreatment MediumControl
 [7] MediumTreatment MediumControl
 [9] ShortTreatment ShortControl
[11] ShortTreatment ShortControl
[13] MediumTreatment MediumControl
[15] MediumTreatment MediumControl
[17] ShortTreatment ShortControl
[19] ShortTreatment ShortControl
[21] MediumTreatment MediumControl
[23] MediumTreatment MediumControl
4 Levels: MediumControl ... ShortTreatment
```

Ahora vamos a ajustar un modelo en donde no vamos a incorporar una constante sino que las variables predictoras nos van a indicar cada una de las categorías que tenemos.

```
design = model.matrix(~ 0 + time2Pi)
colnames(design) = levels(time2Pi)
head(design)
```

```
  MediumControl MediumTreatment ShortControl
1 0 0 0
2 0 0 1
3 0 0 0
4 0 0 1
5 0 1 0
6 1 0 0
  ShortTreatment
1 1
2 0
3 1
4 0
5 0
6 0
```

Ahora para evaluar los efectos de los dos factores experimentales hemos de construir los contrastes. Todo el planteamiento que hemos visto del método limma se formulaba directamente para contrastes. Ajustamos los modelos lineales.

```
fit = limma::lmFit(gse25171,design)
```

[62] Un detalle: los nombres de los niveles no pueden empezar por un número.

Construimos los contrastes en que estamos interesados.[62] Los nombre indican claramente lo que estamos evaluando.

```
cont.matrix = limma::makeContrasts(
    dif1 = (MediumControl + MediumTreatment)−
        (ShortControl + ShortTreatment),
    dif2 = (MediumControl + ShortControl)−
        (MediumTreatment + ShortTreatment),
    dif3 = (MediumControl − ShortControl),
    dif4 = (MediumTreatment − ShortTreatment),
    dif5 = (MediumControl − ShortControl) −
        (MediumTreatment − ShortTreatment),
    levels = design)
```

Hemos construido la siguiente matriz de contrastes.

```
cont.matrix
```

```
              Contrasts
Levels dif1 dif2 dif3 dif4 dif5
  MediumControl 1 1 1 0 1
  MediumTreatment 1 -1 0 1 -1
  ShortControl -1 1 -1 0 -1
  ShortTreatment -1 -1 0 -1 1
```

```
fit2 = limma::contrasts.fit(fit,cont.matrix)
fit2 = limma::eBayes(fit2)
```

Ahora podemos evaluar de un modo análogo a lo previo lo que es significativo para cada uno de los contrastes. Cada uno de los coeficientes corresponde ahora a uno de los contrastes en el orden que ocupan en la matriz de contrastes.

```
tt1 = limma::topTable(fit2,coef=1,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt2 = limma::topTable(fit2,coef=2,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt3 = limma::topTable(fit2,coef=3,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt4 = limma::topTable(fit2,coef=4,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
tt5 = limma::topTable(fit2,coef=1,adjust ="BH",
                      sort.by="none",number=nrow(gse25171))
```

Del mismo modo que antes podemos buscar sondas que sean significativas para cada uno de los contrastes. Nos fijamos como ilustración en el contraste `dif5` que evalua la interacción.

```
sel = which(tt5["adj.P.Val"] < .05)
length(sel)
```

```
[1] 1111
```

```
head(tt4[sel,c("PROBEID","ENTREZID")])
```

```
           PROBEID ENTREZID
245052_at 245052_at 817184
245076_at 245076_at 816849
245084_at 245084_at 816861
245096_at 245096_at 818685
245100_at 245100_at 818691
245107_at 245107_at 818767
```

10.3 Limma aplicado a gse44456

Leemos los datos `tamidata::gse44456`. Se trata de evaluar el efecto del alcohol sobre el hipocampo. En https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE44456 tenemos información adicional del problema tratado.

```
pacman::p_load(Biobase,limma)
data(gse44456,package="tamidata")
```

Tenemos las siguientes variables fenotípicas para analizar su influencia en la expresión de las sondas.

```
names(pData(gse44456))
```

```
[1] "case" "gender"
[3] "age" "cirrhosis"
[5] "smoker" "postmortenint"
[7] "pH" "batch"
```

Vamos a considerar la variable que nos da el tiempo desde el fallecimiento hasta la toma de la muestra.

```
pData(gse44456)[,"postmortenint"]
```

Determinamos la matriz de modelo.

```
design = model.matrix(~ pData(gse44456)[,"postmortenint"])
colnames(design) = c("intercept","postmortenint")
head(design,n=2)
```

```
  intercept postmortenint
1 1 16.75
2 1 27.00
```

Vemos que en la matriz tiene la columna de unos correspondiente a la constante y la columna correspondiente a la variable predictora considerada que es numérica. Ajustamos los modelos lineales.

```
fit = limma::lmFit(gse44456,design)
```

Hemos realizado una regresión lineal simple para cada una de las sondas. Podemos ver los dos coeficientes (constante y el coeficiente del predictor `postmortenint` para cada uno de los ajustes.

```
head(coefficients(fit),n=2)
```

```
       intercept postmortenint
7892501 2.955131 0.001088853
7892502 4.283626 0.003912042
```

Los errores estándar estimados serían

```
head(fit$sigma)
```

```
  7892501 7892502 7892503 7892504 7892505
0.5416383 0.5042429 0.6797924 0.4761044 0.3454714
  7892506
0.5016761
```

```
fit1 = limma::eBayes(fit)
```

Si la constante por la que multiplicamos `postmortenint` es nula indicará que no hay una dependencia del nivel de expresión respecto de la tasa de crecimiento celular. Veamos los resultados.

```
topTable(fit1,coef=2,adjust ="BH",number=3)
```

```
           logFC AveExpr t
7898750 -0.01150852 6.747570 -6.443278
8170468 0.01311588 6.012464 5.689956
7969651 -0.01098933 7.051056 -5.687741
          P.Value adj.P.Val B
7898750 1.016754e-07 0.003385485 6.398649
8170468 1.196673e-06 0.013378383 3.937214
7969651 1.205368e-06 0.013378383 3.930008
```

Aunque alguno de los p-valores originales pudieran (marginalmente) considerarse como significativos cuando corregimos por comparaciones múltiples por el método de Benjamini-Hochberg no lo son salvo tres de ellos si trabajamos con una FDR de 0.05.

Analizamos la posible dependencia con el alcoholismo recogida en la covariable **`case`**.

```
design = model.matrix(~ 0 + pData(gse44456)[,"case"])
colnames(design) = c("control","alcoholic")
head(design)
```

```
  control alcoholic
1 1 0
2 0 1
3 1 0
4 0 1
5 1 0
6 0 1
```

Ajustamos el modelo.

```
fit = limma::lmFit(gse44456,design)
```

Podemos definir un contraste como la diferencia de las medias entre los dos grupos.

```
(contrast.matrix = makeContrasts(dif = control - alcoholic,
                                 levels = design))
```

```
         Contrasts
Levels dif
  control 1
  alcoholic -1
```

Estimamos.

```
fit2 = contrasts.fit(fit,contrast.matrix)
fit2 = eBayes(fit2)
```

Veamos cuáles son significativos.

```
topTable(fit2,coef=1,adjust="BH",number=3)
```

```
           logFC AveExpr t
7927186 -0.5424921 7.826424 -5.914411
8125919 -1.1358866 8.313619 -5.628792
8021081 -1.2885800 8.593822 -5.481851
          P.Value adj.P.Val B
7927186 5.718213e-07 0.01903993 5.029195
8125919 1.455678e-06 0.02236997 4.307423
8021081 2.351231e-06 0.02236997 3.935029
```

En el segunda análisis nos fijamos en dos covariables, el alcoholismo y el sexo de la persona. Es un diseño factorial 2×2. Veamos cuántos datos tenemos en cada celda.

```
table(pData(gse44456)[,"case"],pData(gse44456)[,"gender"])
```

```
           male female
  control 13 6
  alcoholic 14 6
```

En un diseño como este las preguntas habituales son las siguientes:

1. ¿Qué genes muestran un comportamiento diferenciado o relacionado con el alcoholismo?
2. ¿Qué genes se comportan de un modo distinto para hombres y mujeres?
3. ¿Para qué genes los cambios en su expresión según el alcoholismo son distintos en cada uno de los sexos?

Una aproximación simple y efectiva es construir un solo factor con todas las combinaciones de los factores.

```
casegender = vector("list",ncol(gse44456))
for(i in seq_along(casegender))
  casegender[[i]] = paste0(pData(gse44456)[,"case"][i],
                           pData(gse44456)[,"gender"][i])
casegender = factor(unlist(casegender))
```

Consideremos la siguiente matriz de diseño.

```
design = model.matrix(~ 0 + casegender)
colnames(design) = levels(casegender)
head(design)
```

```
  alcoholicfemale alcoholicmale controlfemale
1 0 0 0
2 0 1 0
3 0 0 0
4 0 1 0
5 0 0 0
6 0 1 0
  controlmale
1 1
2 0
3 1
4 0
5 1
6 0
```

Podemos ver que cada coeficiente corresponde con la expresión media para la correspondiente combinación de factores.

```
fit = limma::lmFit(gse44456,design)
```

Construimos los contrastes en que estamos interesados.

```
cont.matrix = makeContrasts(
    dif1 = controlmale - alcoholicmale,
    dif2 = controlfemale - alcoholicfemale,
    dif12 = (controlmale - alcoholicmale)-
        (controlfemale - alcoholicfemale),
    levels = design)
fit2 = contrasts.fit(fit,cont.matrix)
fit2 = limma::eBayes(fit2)
```

Podemos ejecutar el siguiente código y podemos comprobar que ningún contraste es significativo.

```
topTable(fit2,coef=1,adjust="BH")
topTable(fit2,coef=2,adjust="BH")
topTable(fit2,coef=3,adjust="BH")
```

10.4 Ejercicios

* **Ex. 22** — Utilizando los datos `tamidata::gse20986` se pide:

1. Seleccionar las muestras correspondientes a iris y huvec.
2. Aplicar el método SAM y obtener el grupo de genes significativos.
3. Comparar con el grupo obtenido en el apartado 4 del ejercicio 20.

* **Ex. 23** — Utilizando los datos `tamidata::gse20986` se pide determinar los genes significativos cuando comparamos los cuatro grupos considerados en el estudio.

Capítulo 11

Expresión diferencial con datos RNASeq

11.1 Introducción

Tenemos datos de expresión de gen obtenidos mediante la técnica conocida como RNASeq. Pero, ¿qué tenemos realmente? ¿Con qué vamos a trabajar?

Ya hemos realizado todo el preprocesado. Ya hemos alineado y contado lecturas asociadas **a gen o a transcritos**. Nuestra información muestral es una matriz de expresión **observada** $\boldsymbol{y} = [y_{ij}]_{i=1,\ldots,N;j=1,\ldots,n}$ donde hemos considerado N genes y tenemos n muestras. Además tendremos covariables, variables fenotípicas o metadatos. [63] Estas covariables son otra matriz $n \times p$ donde n es el número de muestras y p es el número de covariables de las que disponibles sobre las muestras. Tendremos $\boldsymbol{x} = [x_{jk}]_{j=1,\ldots,n;k=1,\ldots,p}$. Habitualmente $p = 1$ y $\boldsymbol{x} = (x_1, \ldots, x_n)^T$ es simplemente un vector.

Como indicamos el estudio se puede realizar a nivel de transcrito o a nivel de gen. Si utilizamos sus abreviaturas en inglés podemos hablar de DTE (differential transcript expression) y DGE (differential gene expression). En lo que sigue abreviamos de este modo.

[63] Un estadístico dice covariable, un bioinformático dice variable fenotípica y un moderno de los de los Big Data diría cualquier cosa.

11.2 edgeR

El título de la sección hace referencia al paquete (de largo desarrollo) [25, edgeR]. Con frecuencia se habla de método `edgeR` sin más indicación. Este paquete lleva un cuerpo metodológico importante y lo correcto es referenciar las funciones que se están utilizando y las referencias bibliográficas en donde se explican. Hay aproximaciones muy distintas contenidas en este paquete.[64]

[64] Un comentario similar se puede aplicar a muchos otros paquetes de R. Una costumbre incorrecta es denotar el método de análisis por el paquete que lo incluye y no por la referencia bibliográfica que lo propone. Muchos paquetes incluyen métodos muy diversos.

11.2.1 edgeR clásico

En esta sección tratamos un procedimiento propuesto en [78, 77][65] y que está implementado en [25].

[65] Hay que leer primero [78] y luego [77] pues es la secuencia temporal de los trabajos aunque se publicaron en orden invertido.

Estimación de una dispersión común

Consideramos una característica en n muestras (o librerías) de tamaños distintos.[66] Sea m_j el tamaño de la j-ésima librería (total de

[66] Son réplicas de una mis-

lecturas). Sea λ la proporción que hay en una librería cualquiera de la característica en que estamos interesados. *Vamos a asumir* que Y_j tiene una distribución binomial negativa con media $\mu_j = m_j\lambda$ y con dispersión ϕ. Y tiene una distribución binomial negativa con media μ y dispersión ϕ, $Y \sim NB(\mu, \phi)$, si su función de probabilidad es $P(Y = y|\mu, \phi) = \frac{\Gamma(y+\phi^{-1})}{\Gamma(\phi^{-1})\Gamma(y+1)}\left(\frac{1}{1+\mu\phi}\right)^{\phi^{-1}}\left(\frac{\mu}{\phi^{-1}+\mu}\right)^{y}$ para $y = 0, 1, \ldots$. Su media es $E(Y) = \mu$ y su varianza $var(Y) = \mu + \phi\mu^2$.

Supongamos el caso ideal (e irreal) en que todas las librerías tienen el mismo tamaño: $m_j = m$ para $j = 1, \ldots, n$. Bajo esta condición tenemos que $Z = \sum_{i=j}^{n} Y_j \sim NB(nm\lambda, \phi/n)$. Además la logverosimilitud condicionada al valor de $Z = z$ no depende de λ y podemos estimar el valor de ϕ. Se estimaría ϕ maximizando la logverosimilitud condicionada dada por

$$\ell_{\mathbf{y}|z} = \left[\sum_{j=1}^{n} \log\Gamma(y_j + 1/\phi)\right] + \log\Gamma(n/\phi) - \log\Gamma(z + n/\phi) - n\log\Gamma(1/\phi), \quad (11.1)$$

donde $\mathbf{y} = (y_1, \ldots, y_n)$ son los conteos observados. Si los tamaños de las librerías son distintos la verosimilitud no tiene una expresión simple. La idea en [78] es modificar los valores observados y_i y generar unos nuevos datos en que los tamaños de las librerías coincidan, los *pseudodatos*. El tamaño común será la media geométrica de los tamaños observados, $m^* = \sqrt[n]{\prod_{j=1}^{n} m_j}$. ¿Cómo transformamos los conteos y_j? El procedimiento propuesto es:

1. Inicializamos ϕ (por ejemplo, con el estimador maxímo verosímil condicionado sin realizar ningún ajuste).
2. Dado el valor estimado de ϕ, estimamos λ maximizando la verosimilitud para el valor estimado de la dispersión.
3. Suponemos que cada conteo y_j es un valor observado de una distribución binomial negativa con media $m_j\lambda$ y parámetro de dispersión ϕ. Calculamos los percentiles
$$p_j = P(Y < y_j|m_j\lambda, \phi) + \frac{1}{2}P(Y = y_j|m_j\lambda, \phi), \quad (11.2)$$
para $j = 1, \ldots, n$.
4. Suponemos ahora una distribución binomial negativa con media $m^*\lambda$ y dispersion ϕ. Determinamos qué valor sería el percentil de orden p_j en la nueva distribución. Ya no tiene porqué ser un dato entero e incluso puede ser negativo. Los pseudodatos para un mismo gen tienen aproximadamente la misma distribución.
5. Estimamos la dispersión ϕ con los pseudodatos utilizando la veromilitud condicionada a la (pseudo) suma total de un gen.
6. Repetimos desde 2 hasta 5 hasta que converja ϕ.

A este procedimiento de estimación de ϕ y λ los autores lo denominan *máxima verosimilitud condicionada ajustada por cuantiles*.[67] Lo fundamental es entender que estamos modificando los datos para conseguir un tamaño común de las librerías, una misma profundidad de secuenciación.

[67] Quantile-adjusted conditional maximum likelihood (qCML).

Un test exacto

Tenemos dos condiciones a comparar o las comparamos de dos en dos.

Estimación de la dispersión común Vamos a asumir en un primer momento que el parámetro de dispersión es común. Tenemos y_{ijk} el conteo para la muestra k en la condición j del gen i donde $j = 1, 2$ y $k = 1, \ldots, n_j$. El tamaño de la librería de la condición (j, k) en el gen i es m_{jk}. Supongamos que aplicamos el método qCML dentro de cada condición j de modo que por condición tenemos un tamaño de librería común. Con los pseudodatos que denotamos, por simplicidad notacional, del mismo modo como y_{ijk} podemos condicionar a la suma total por gen dentro de cada clase. **Como el parámetro de dispersión se asume el mismo para todos los genes y condiciones** podemos considerar la logverosimilitud condicionada a $z_{ij} = y_{ij\cdot} = \sum_{k=1}^{n_j} y_{ijk}$ para la dispersión ϕ dada por

$$l_i(\phi) = \sum_{j=1}^{2} \left(\sum_{k=1}^{n_j} \log\Gamma(y_{ijk} + \phi^{-1}) + \log\Gamma(n_j\phi^{-1}) - \right.$$
$$\left. \log\Gamma(z_{ij} + n_j\phi^{-1}) - n_j \log\Gamma(\phi^{-1}) \right). \quad (11.3)$$

En consecuencia la logverosimilitud común es[68]

$$l(\phi) = \sum_{i=1}^{N} l_i(\phi). \quad (11.4)$$

[68] Sería más correcto hablar de pseudoverosimilitud ya que los distintos genes son dependientes.

La estimación de ϕ en 66 se obtiene maximizando la función dada en la ecuación 11.4. Denotamos a este estimador común por $\hat{\phi}_C$ y es el usado en lo que sigue.

Contraste de hipótesis Tenemos dos condiciones a comparar y estamos asumiendo

$$EY_{ijk} = m_{jk}\lambda_{ij},$$

para cualquier i, j, k siendo Y_{ijk} el conteo aleatorio del gen i en la condición j y en la muestra k. Si consideramos la hipótesis nula de que no hay diferencia entre los valores de λ,

$$H_i: \quad \lambda_{i1} = \lambda_{i2},$$
$$K_i: \quad \lambda_{i1} \neq \lambda_{i2}.$$

Bajo la hipótesis nula el valor λ no depende de la condición. Aplicamos el método qCML a **todas** las muestras (tenemos $n = n_1 + n_2$). Como es habitual denotamos los nuevos pseudodatos como los originales. El tamaño de las librerías es la media geométrica de los tamaños originales y la denotamos por m^*.[69] Estos nuevos psedodatos tienen una distribución común. En concreto para el gen i denotamos por y_{ijk} el conteo para la muestra k en la condición j donde $j = 1, 2$ y $k = 1, \ldots, n_j$. El tamaño común de todas las librerías es m^*. Si consideramos la variable aleatoria Y_{ijk} (de la cual y_{ijk} sería el valor observado) entonces

$$EY_{ijk} = m^*\lambda_{ij},$$

[69] Importante, los pseudodatos no son los mismos utilizados para estimar la dispersión común.

para cualquier i, j, k. Bajo la hipótesis nula H_0 no tendríamos diferencia en el valor de λ entre las condiciones y sería un valor común $\lambda_i = \lambda_{i1} = \lambda_{i2}$ para el gen i. Utilizando el estimador previamente calculado $\hat{\phi}_C$ y los pseudatos (que obtienen un tamaño de librería común) y_{ijk} con $k = 1, \ldots, n_j$ podemos estimar λ_i. Simplemente el estimador máximo verosímil no es más que el cociente de la suma de los conteos dividido por la suma de los tamaños de las librerías. Tenemos las estimaciones $\hat{\lambda_i}$ y $\hat{\phi}_C$.

Bajo la hipótesis nula de no diferencia entre grupos tendríamos que $Y_{ij\cdot} = \sum_{k=1}^{n_j} Y_{ijk} \sim NB(n_j m^* \hat{\lambda}_i, \hat{\phi}_C/n_j)$ para $j = 1, 2$. Además $Y_{i1\cdot}$ e $Y_{i2\cdot}$ son independientes. La suma $Y_{i1\cdot} + Y_{i2\cdot}$ también tiene una distribución binomial negativa: $Y_{i1\cdot} + Y_{i2\cdot} = \sum_{i=1}^{2} \sum_{k=1}^{n_j} Y_{ijk} \sim NB((n_1 + n_2)m^* \hat{\lambda}_i, \hat{\phi}_C/(n_1 + n_2))$. Podemos considerar la distribución condicionada del vector aleatorio $(Y_{i1\cdot}, Y_{i2\cdot})$ a la suma $Y_{i1\cdot} + Y_{i2\cdot}$ y considerar las probabilidades de los conteos conjuntos que *son menos probables que el observado*. La suma de estas probabilidades nos daría el p-valor del test.[70]

[70] Es un test similar al test de Fisher bilateral en donde las probabilidades hipergeométricas son sustituidas por probabilidades por la distribución binomial negativa. Ver [1, Página 93].

Dispersiones posiblemente distintas

En el test exacto que hemos visto en 68 hemos asumido una dispersión común ϕ y se ha estimado su valor utilizando los conteos asociados a todos los genes y condiciones bajo las cuales se han observado a estos genes. Bajo la hipótesis nula de un misma media, y asumiendo la dispersión común esto tenía sentido. Es, sin duda, una hipótesis bastante exigente. Una misma dispersión sobre una cantidad grande de genes no es muy razonable.[71] En [77] proponen no asumir un mismo valor para cada gen y considerar que las dispersiones dependen del gen. Tenemos ahora, en lugar del valor común ϕ, un valor (posiblemente) distinto para el gen i, ϕ_i. ¿Y cómo lo estimamos? Proponen un compromiso entre la contribución que hacen los conteos del i-ésimo gen, l_i (ecuación 11.3), a la logverosimilitud global, l (ecuación 11.4), y la propia logverosimilitud global. En concreto hablan de la logverosimilitud condicionanda ponderada definida como

[71] Buena desde el punto de vista del modelo y su estimación pero nos aleja de los datos.

$$WL(\phi_i) = l_i(\phi_i) + \alpha l(\phi_i), \tag{11.5}$$

siendo α el peso que se da a la verosimilitud global. Es una función que propone un compromiso entre considerar l como función a maximizar considerando la misma contribución a todos los genes y l_i en donde solamente consideramos los conteos del propio gen. El problema fundamental a considerar ahora es la elección del valor de α. Un mayor α nos aproxima a un valor común para la dispersión y un menor valor nos dará estimaciones distintas de la dispersión para cada gen. ¿En qué punto nos quedamos? En [77, Sección, 3.2, pág. 2883] dan una motivación que no es un criterio de optimalidad. Simplemente argumentan lo razonable de su elección. Para explicar cómo lo hacen hemos de considerar la **función score** definida como

$$S_i(\phi) = \frac{\partial l_i(\phi)}{\partial \phi} \tag{11.6}$$

y la información esperada definida como

$$I_i(\phi) = E(J_i), \text{ siendo } J_i = -\frac{\partial^2 l_i(\phi)}{\partial \phi^2}. \tag{11.7}$$

El procedimiento de estimación de α y la posterior estimación de ϕ_i es el siguiente:

1. Estimamos la dispersión común maximizando la verosimilitud global l: $\hat{\phi}_0$.
2. Evaluamos para cada gen i: $S_i(\hat{\phi}_0)$ y $I_i(\hat{\phi}_0)$.
3. Estimamos τ_0 resolviendo la ecuación

$$\sum_{i=1}^{N}\left[\frac{S_i(\hat{\phi}_0)}{I_i(\hat{\phi}_0)(1+I_i(\hat{\phi}_0)\tau_0^2)}-1\right]=0.$$

   Si $\sum_{i=1}^{N} S_i^2(\hat{\phi}_0)/I_i(\hat{\phi}_0) < N$ entonces $\tau_0 = 0$.
4. Fijamos

$$1/\alpha = \tau_0^2 \sum_{i=1}^{N} I_i(\hat{\phi}_0).$$

5. Una vez estimado α en el paso anterior, maximizamos la función $WL(\phi_i)$ definida en la ecuación 11.5.

Finalmente los autores utilizan en lugar de la información esperada $I_i(\hat{\phi}_0)$ una aproximación que consiste en utilizar la observada (entendiendo por observada la correspondiente a los conteos dados). Es decir, sustituyen $I_i(\hat{\phi}_0)$ por $J_i(\hat{\phi}_0)$. Además comentan que trabajan con $\delta = \phi/(\phi+1)$.[72]

[72] Lo cual es irrelevante. Estimamos δ estimamos ϕ pues hay una correspondencia 1-1.

PRJNA297664

En esta sección analizamos los datos `tamidata::PRJNA297664`. Utilizamos parcialmente código de [24].

```
data(PRJNA297664,package="tamidata")
```

Construimos el objeto de clase `edgeR::DGEList`.

```
pacman::p_load(edgeR,SummarizedExperiment)
dge = edgeR::DGEList(counts=assay(PRJNA297664),
                     group=colData(PRJNA297664)[,"treatment"])
```

Podemos ver la variable de agrupación `group` y los tamaños de las librerías (número total de lecturas por muestra).

```
head(dge$samples,n=2)
```

```
       group lib.size norm.factors
Sample1 Wild 4788536 1
Sample2 Wild 9387986 1
```

Añadimos al objeto `dge` la anotación que ya tenemos definida en el `slot SummarizedExperiment::rowData`.

```
dge$genes = rowData(PRJNA297664)
```

Podemos ver que no de todos los genes tenemos su anotación.

```
sapply(rowData(PRJNA297664),function(x) sum(is.na(x)))
```

```
    ORF SGD ENTREZID ENSEMBL
      0  10      794    1250
```

Vamos a eliminar genes con conteos bajos. Esto podría deberse a la no expresión del gen o bien indica una actividad baja y estadísticamente no sería muy fiable su análisis. En ambos casos parece razonable eliminarlo del estudio. La selección la haremos básandonos en los conteos por millón. Si y_{ij} es el conteo de la j-ésima muestra correspondiente al i-ésimo y $y_{\cdot j} = \sum_{i=1}^{N} y_{ij}$, la profundidad de secuenciación de la muestra j, entonces el conteo por millón para la muestra j-ésima del gen i-ésimo se define como

$$y_{ij}^{*} = cpm(y_{ij}) = 10^{6} \frac{y_{ij}}{y_{\cdot j}}. \tag{11.8}$$

Los conteos por millón se pueden calcular con `edgeR::cpm()`.

```
head(edgeR::cpm(dge),n=2)
```

```
        Sample1 Sample2 Sample3 Sample4 Sample5
15S_rRNA 0 0 0 0 0
21S_rRNA 0 0 0 0 0
        Sample6
15S_rRNA 0
21S_rRNA 0
```

¿Qué criterio utilizar? Siguiendo indicaciones de [24] podríamos considerar que los conteos por millón superen al menos 0.5 en un número de muestras (que obviamente han de depender de nuestro tamaño muestral). Tenemos 6, podemos pedir que en al menos un par de muestras se verifique la condición (en la referencia indicada son menos exigentes y piden 2 de 12).

```
to_keep1 = rowSums(cpm(dge) > 0.5) > 2
table(to_keep1)
```

```
to_keep1
FALSE TRUE
  948 6178
```

No perdemos más que 948. También podemos basar la selección de genes en valores absolutos. Por ejemplo, podemos ver los conteos por fila. Buscamos el percentil de orden 0.1.

```
quantile(rowSums(dge$counts),probs=.1)
```

```
10%
  6
```

Y podemos quedarnos con genes con una expresión absoluta superior al valor anterior redondeando por exceso que sería 0.

```
to_keep2 = rowSums(dge$counts) > 6
table(to_keep2)
```

```
to_keep2
FALSE TRUE
  714 6412
```

No hay un criterio claro. Elegimos la primera opción en nuestro caso.

```
dge = dge[to_keep1,keep.lib.sizes=FALSE]
dge$samples$lib.size
```

```
[1] 4788094 9387257 9599214 8895385 9003074
[6] 9001313
```

La opción `keep.lib.sizes=FALSE` obliga a que se recalculen los tamaños de las librerías después de la selección que acabamos de hacer.

Se estima el parámetro de dispersión que asumimos un valor común para todos los genes con el siguiente código.

```
dge.c = estimateCommonDisp(dge)
dge.c$common.dispersion
```

```
[1] 0.01170845
```

Podemos estimar parámetros de dispersión por gen. Previamente necesitamos haber estimado el parámetro de dispersión común.

```
dge.t = estimateTagwiseDisp(dge.c)
head(dge.t$tagwise.dispersion)
```

```
[1] 0.01875437 0.01033898 0.05391892 0.02503803
[5] 0.02012587 0.01766447
```

Se realizan los test exactos asumiendo un parámetro de dispersión común.

```
et.c = exactTest(dge.c)
```

Y ver los genes significativos (mostramos los primeros).

```
topTags(et.c,n=2)
```

```
Comparison of groups: SEC66 deletion-Wild
DataFrame with 2 rows and 8 columns
             ORF SGD ENTREZID
       <character> <character> <character>
YBR171W YBR171W S000000375 852469
YCR021C YCR021C S000000615 850385
          ENSEMBL logFC logCPM
       <character> <numeric> <numeric>
YBR171W YBR171W -10.15022 6.11504
YCR021C YCR021C -1.92879 8.46385
           PValue FDR
         <numeric> <numeric>
YBR171W 2.0977e-258 1.29596e-254
YCR021C 4.8495e-47 1.49801e-43
```

Calculamos los tests exactos y mostramos los significativos asumiendo un parámetro de dispersión distinto por gen.

```
et.t = exactTest(dge.t)
topTags(et.t,n=2)
```

```
Comparison of groups: SEC66 deletion-Wild
DataFrame with 2 rows and 8 columns
             ORF SGD ENTREZID
       <character> <character> <character>
YBR171W YBR171W S000000375 852469
YGL255W YGL255W S000003224 852637
          ENSEMBL logFC logCPM
       <character> <numeric> <numeric>
YBR171W YBR171W -10.15052 6.11504
YGL255W YGL255W -1.84605 7.51858
           PValue FDR
         <numeric> <numeric>
YBR171W 1.20767e-297 7.46099e-294
YGL255W 1.00554e-29 3.10613e-26
```

Si queremos obtener los resultados para todos los genes manteniendo su orden original lo podemos hacer con

```
tt1 = topTags(et.t,n=nrow(PRJNA297664),sort.by="none")
```

Y si queremos obtener un `DFrame` para seguir trabajando los datos podemos hacerlo con

```
head(tt1$table,n=3)
```

```
DataFrame with 3 rows and 8 columns
           ORF SGD ENTREZID
    <character> <character> <character>
ICR1 ICR1 S000132612 9164906
LSR1 LSR1 S000006478 9164871
NME1 NME1 S000007436 9164967
       ENSEMBL logFC logCPM PValue
    <character> <numeric> <numeric> <numeric>
ICR1 NA 0.448839 3.511993 0.0290942
LSR1 NA -0.254297 3.382420 0.1519951
NME1 NA 0.417201 0.883312 0.2929631
         FDR
    <numeric>
ICR1 0.0915547
LSR1 0.2835223
NME1 0.4482234
```

TCGA-COAD

Utilizamos los datos obtenidos en § 3.3.2.

```
pacman::p_load(edgeR,SummarizedExperiment)
load(paste0(dirTamiData,"tcga_coad.rda"))
```

Nos centramos en la variable fenotípica `tissue_or_organ_of_origin` que tiene 9 niveles con etiquetas y conteos

```
table(colData(tcga_coad)[,"tissue_or_organ_of_origin"])
```

```
       Ascending colon Cecum
                  72 70
           Colon, NOS Descending colon
                  59 15
Hepatic flexure of colon Rectosigmoid junction
                  12 3
         Sigmoid colon Splenic flexure of colon
                  76 6
      Transverse colon
                  13
```

Eliminamos aquellas muestras que tienen algún dato faltante. Las funciones que siguen necesitan que no haya dato faltante.

```
toremove = which(is.na(colData(tcga_coad)[,"tissue_or_organ_of_origin"]))
tcga_coad = tcga_coad[,-toremove]
```

Contruimos el objeto de clase `DGEList`.

```
dge = DGEList(counts=assay(tcga_coad),
              group=colData(tcga_coad)[,"tissue_or_organ_of_origin"])
```

Eliminamos genes con conteos bajos.

```
to_keep = rowSums(cpm(dge) > 0.5) > 2
dge = dge[to_keep,keep.lib.sizes=FALSE]
```

Estimamos las dispersiones tanto en el que asumimos una común para todos los genes como las que consideramos distintas por gen.

```
dge.c = estimateCommonDisp(dge)
dge.t = estimateTagwiseDisp(dge.c)
```

Comparamos dos de las categorías que nos define nuestro factor experimental.

```
et.c = exactTest(dge.c,pair=c("Ascending colon",
                              "Descending colon"))
et.t = exactTest(dge.t,pair=c("Ascending colon",
                              "Descending colon"))
```

Determinamos los genes significativos.

```
topTags(et.c,n=3)
```

```
Comparison of groups: Descending colon-Ascending colon
         logFC logCPM PValue
PRAC1 4.450539 3.7744837 4.666555e-70
IGFN1 4.212251 0.1655621 5.306039e-59
CALCB -9.164919 3.0768624 6.055306e-58
              FDR
PRAC1 8.007809e-66
IGFN1 4.552581e-55
CALCB 3.463635e-54
```

```
topTags(et.t,n=3)
```

```
Comparison of groups: Descending colon-Ascending colon
        logFC logCPM PValue
ANO4 3.068582 -1.5921044 7.418709e-17
IGFN1 4.210518 0.1655621 1.585287e-15
ACTL8 3.639962 1.9367208 6.320150e-11
              FDR
ANO4 1.273050e-12
IGFN1 1.360177e-11
ACTL8 2.885746e-07
```

11.2.2 edgeR utilizando modelo lineal generalizado

Esta sección se basa en [63]. Se asume un cierto conocimiento de modelos lineales generalizados. Para una introducción rápida y simple podemos consultar [5, capítulo 5, sección 7.5]. Una introducción más completa a un nivel básico es [3] (con código en R) y, por último, una presentación amplia la encontramos en [2].

Siguiendo la notación del manual denotaremos por Y_{ij} el conteo aleatorio (número de lecturas alineadas) para el gen i en la muestra j. Denotamos por $m_j = \sum_{i=1}^{N} y_{\cdot j}$ la profundidad de secuenciación o total de lecturas de la muestra j. Utilizamos como función de enlace el logarítmo natural y consideramos la profundidad de secuenciación como offset (un modelo de tasas sobre la profundidad de secuenciación). El modelo para la media es

$$\ln \mu_{ij} = \boldsymbol{x}_j^T \boldsymbol{\beta}_i + \ln m_j. \tag{11.9}$$

En el modelo que se propone en 11.9 hay que tener en cuenta que las variables predictoras (o variables fenotípicas en este contexto) son comunes a todos los genes. Por ello no aparece en el vector de predictoras $\boldsymbol{x}_j$ la dependencia del gen i. Asumiendo que la componente

aleatoria del modelo sigue una distribución binomial negativa (con el parámetro de dispersión conocido porque de lo contrario no estamos en la familia de dispersión exponencial) entonces la varianza de la respuesta es

$$var(Y_{ijk}) = \mu_{ij} + \phi_i \mu_{ij}^2, \tag{11.10}$$

siendo ϕ_i el parámetro de dispersión que hemos de asumir conocido o, de otro modo, tenemos que estimarlo previamente. En [63, pág. 4290] muestran cómo estimar por máxima verosimitud el vector de coeficientes $\boldsymbol{\beta}_i$. Utilizan una modificación de los mínimos cuadrados iterativamente reponderados (IRWLS). El parámetro de dispersión se estima maximizando la logverosimilitud penalizada definida como

$$APL_i(\phi_i) = \ell(\phi_i; \boldsymbol{y}_i, \hat{\boldsymbol{\beta}}_i) - \frac{1}{2} \ln |\mathbb{I}_i| \tag{11.11}$$

siendo $\boldsymbol{y}_i$ los conteos para el gen i, $\hat{\boldsymbol{\beta}}_i$ el vector de coeficientes, $l()$ es la función de logverosimilitud y $|\mathbb{I}_i|$ el determinante de la matriz de información de Fisher para el i-ésimo gen.

TCGA-COAD

Esta sección utiliza material de [62]. Vamos a utilizar como variables predictoras `age_at_diagnosis` y `tissue_or_organ_of_origin`.

```
pacman::p_load(edgeR,SummarizedExperiment)
load(paste0(dirTamiData,"tcga_coad.rda"))
```

Hemos de eliminar aquellas muestras que tienen las variables predictoras con datos faltantes ya que las funciones que siguen no los admiten.

```
torm1 = which(is.na(colData(tcga_coad)$"age_at_diagnosis"))
torm2 = which(is.na(colData(tcga_coad)$ "tissue_or_organ_of_origin"))
toremove = union(torm1,torm2)
tcga_coad = tcga_coad[,-toremove]
```

Construimos el objeto `DGEList` sin indicar ninguna variable `group` ni ninguna matriz de modelo y eliminamos genes con conteos bajos.

```
dge = DGEList(counts=assay(tcga_coad))
to_keep = rowSums(cpm(dge) > 0.5) > 20
dge = dge[to_keep,keep.lib.sizes=FALSE]
```

Construimos la matriz de modelo con las dos variables predictoras, una de caracter categórico y la otra numérica. También cambiamos los nombres de las columnas de la matriz de modelo.

```
design0 = model.matrix(~ 0 + colData(tcga_coad)$"
    ↪ tissue_or_organ_of_origin"
                    + colData(tcga_coad)$"age_at_diagnosis")
y = levels(colData(tcga_coad)$"tissue_or_organ_of_origin")
y = sapply(y,function(x) gsub(" ","_",x)) ## Eliminamos espacios
y = sapply(y,function(x) gsub(",","_",x)) ## Eliminamos las comas
colnames(design0) = c(y,"age_at_diagnosis")
```

Estimamos las dispersiones por tres métodos distintos: asumiendo una dispersión común, una por gen y con una relación media-varianza. Los métodos con los que se estiman en las dos primeras opciones son distintos al caso del método edgeR clásico de 73.

```
dge = estimateDisp(dge,design=design0)
```

Si solo queremos una de las tres opciones podemos usar las funciones `estimateGLMCommonDisp()`, `estimateGLMTagwiseDisp()` y `estima` ↪ . Podemos ver los estimadores de los parámetros de dispersión en la figura 11.1 generada del siguiente modo.

```
png(paste0(dirTamiFigures,
           "tcga_coad_dge_design0_dispersion.png"))
plotBCV(dge)
dev.off()
```

Ajustamos los modelos lineales generalizados.

```
fit = glmFit(dge, design=design0)
```

Veamos si influye en los conteos observados la variable `age_at_` ↪ `diagnosis`. Si observamos la matriz de modelo `design0` definida previamente corresponde con la columna 10 de la matriz de modelo. Se realiza un test del cociente de verosimilitudes.

```
lrt1 = glmLRT(fit,coef="age_at_diagnosis")
lrt1 = glmLRT(fit,coef=10) ## Equivalente a la línea anterior
topTags(lrt1,n=3)
```

Podemos evaluar toda la variable `tissue_or_organ_of_origin`.

```
lrt2 = glmLRT(fit,coef=1:9)
topTags(lrt2,n=3)
```

Y elegir los contraste que queramos. Mostramos una comparación entre dos grupos.

```
AD = makeContrasts(contrast1 = Ascending_colon - Descending_colon,
                   levels=design0)
lrt3 = glmLRT(fit,contrast = AD)
topTags(lrt3,n=3)
```

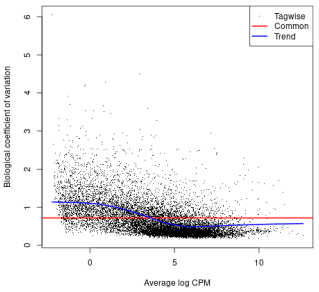


Figura 11.1: Estimadores de los parámetros de dispersión según los tres métodos.

11.3 Ejercicios

Ex. 24 — Para los datos `tamidata::PRJNA297664` se pìde:

1. Aplicar el procedimiento de comparación de proporciones utilizando edgeR::binomTest a las muestras 1 y 3.
2. Ajustar los p-valores obtenidos en el punto anterior por el método de Benjamini-Hochberg.
3. Una vez ajustados los p-valores determinar cuantos y cuales tienen un p-valor ajustado menor a 0.01.
4. Repetir los tres pasos anteriores comparando las muestras 2 y 5.
5. ¿Cuántos genes han sido detectados como significativos simultáneamente en los puntos 3 y 4?

Ex. 25 — Repetir el análisis realizado en 73 intentando comparar todos los pares dos a dos de la variable `colData(tcga_coad)[,"` ↪ `tissue_or_organ_of_origin"]`.

Ex. 26 — Utilizar el análisis contenido en 73. Se pide comparar los conteos originales con los normalizados. Para ello hacer la diferencia de los mismos y al vector resultante aplicarle el método `summary()`.

Parte V

Análisis de grupos de genes

Capítulo 12

Grupos de genes

12.1 Introducción

Hasta ahora hemos trabajado habitualmente a nivel de gen. Cada gen en cada muestra nos da un nivel de expresión. Obviamente esto supone que por muestra tenemos una muy alta dimensión. Un análisis de expresión diferencial supone que tenemos un contraste por gen. Muchos miles de contrastes supone esto. Si pretendemos clasificar la muestra en distintos fenotipos entonces la dimensión alta con la que trabajamos produce un sobreajuste y modelos que no son generalizables a otros datos. En ambos problemas no es bueno tener una dimensión tan alta. Podemos utilizar componentes principales para resumir el perfil de expresión de la muestra (**??**). Otra opción puede ser trabajar no con genes individuales sino con *grupos de genes*. Si nos interesa la expresión diferencial entre condiciones entonces podemos valorar si el grupo de genes que nos interese, como grupo, se expresa diferencialmente. Si vamos a clasificar en fenotipos, podemos resumir la expresión de los genes en una muestra dentro de cada grupo. Reducimos el número de hipótesis a contrastar (expresión diferencial) o bien la dimensión del vector que utilizamos para clasificar (clasificación en fenotipo).

Obviamente podemos construir el grupo de genes que nos apetezca y evaluarlo. No parece algo muy lógico. Es de sobra conocido en la literatura estadística que un contraste de hipótesis es útil en la misma medida en que está correctamente formulado. Y formular correctamente el contraste supone conocimiento sobre el problema que abordamos. En este caso la comunidad científica ha propuesto y propone y propondrá clasificaciones que se han de utilizar para definir estos grupos.

Tres fuentes vamos a utilizar a la hora de considerar grupos de genes.

Gene Ontology: Gene Ontology Podemos utilizar las tres ontologías consideradas en Gene Ontology. El grupo de genes viene definido por todos aquellos genes que tienen este término.

KEGG: Kyoto Encyclopedia of Genes and Genomes Tenemos las rutas de señalización para muchos organismos.

MSigDB: Molecular Signatures Database En MSigDB [73] consideran hasta ocho formas distintas de construir estos grupos.

[73] The Molecular Signatures Database

DO: Disease ontology

DisGeNET: Disease Gene Network

wikiPathways

En http://www.genesetdb.auckland.ac.nz/haeremai.html tenemos una herramienta web que contiene a su vez a otras muchas bases de datos. Realiza análisis de sobre representación y está orientada a humanos, ratones y ratas.

Hay muchas formas de definir estos grupos. En cualquier caso en su definición interviene un conocimiento previo. No utilizamos los propios datos de expresión con los que posteriormente vamos a trabajar. No son grupos obtenidos por un análisis cluster previo.

En este tema tratamos de cómo construir grupos de genes utilizando R/Bioconductor. Es un tema de carácter muy técnico previo al que sigue en donde analizaremos la posible expresión diferencial de estos grupos.[74] Vamos a definir conjuntos de genes o colecciones de conjuntos de genes utilizando el paquete [66, GSEABase].

[74] Siempre nuestro interés principal.

```
library(GSEABase)
```

12.2 Homo sapiens

La opción más simple sería definirnos nuestro propio conjunto de genes. Para ello podemos utilizar la función `GSEAbase::GeneSet()`. Tomamos como ejemplo GSE20986 (**??**).

```
data(gse20986,package="tamidata")
```

Supongamos que queremos construir un conjunto de genes formado por aquellos que ocupan las filas de la 345 a la 405 (sin ningún sentido práctico). Los índices los podemos obtener con

```
345:405
```

```
 [1] 345 346 347 348 349 350 351 352 353 354 355
[12] 356 357 358 359 360 361 362 363 364 365 366
[23] 367 368 369 370 371 372 373 374 375 376 377
[34] 378 379 380 381 382 383 384 385 386 387 388
[45] 389 390 391 392 393 394 395 396 397 398 399
[56] 400 401 402 403 404 405
```

Construimos el grupo y le damos nombre.

```
(egs = GeneSet(gse20986[345:405, ], setName = "Burjasot"))
```

Los genes tienen sus identificadores obtenidos de su anotación (mostramos los primeros).

```
head(geneIds(egs))
```

```
[1] "1552739_s_at" "1552740_at" "1552742_at"
[4] "1552743_at" "1552745_at" "1552747_a_at"
```

En este caso, como se obtuvo con Affymetrix GeneChip, los identificadores son los proporcionados por el fabricante y corresponde a los nombres de los conjuntos de sondas (§ 2). Podemos tener más información con

```
details(egs)
```

```
setName: Burjasot
geneIds: 1552739_s_at, 1552740_at, ..., 1552822_at (total: 61)
geneIdType: Annotation (hgu133plus2)
collectionType: ExpressionSet
setIdentifier: debian:10411:Mon Feb 26 18:03:12 2024:8574
description:
organism: Homo sapiens
pubMedIds:
urls:
contributor:
setVersion: 0.0.1
creationDate:
```

En el ejemplo que acabamos de ver hemos creado un conjunto de genes a partir de un ExpresionSet pero podemos hacerlo de otros modos.

```
showMethods("GeneSet", inherited = FALSE)
```

```
Function: GeneSet (package GSEABase)
type="BroadCollection"
type="character"
type="ExpressionSet"
type="GeneIdentifierType"
type="GOCollection"
type="missing"
```

Podemos obtener la correspondencia de nuestros identificadores con otros tipos de identificadores, por ejemplo, los EntrezId:

```
mapIdentifiers(egs, EntrezIdentifier(), verbose = TRUE)
```

También podemos tener una colección de conjuntos, `GSEABase::GeneSetCollection()`. Quizás la mejor manera de definir conjuntos utilizando un `ExpresionSet` con su anotación es hacerlo utilizando Gene Ontology u otra base de datos. Por ejemplo, en el siguiente ejemplo construimos los conjuntos para nuestro ExpressionSet.

```
gsc = GeneSetCollection(gse20986, setType = GOCollection())
```

Podemos ver un resumen de los conjuntos.

```
gsc
```

```
GeneSetCollection
  names: GO:0000002, GO:0000003, ..., GO:2001070 (18426 total)
  unique identifiers: 1555591_at, 201917_s_at, ..., 216045_at (38345
      total)
  types in collection:
    geneIdType: AnnotationIdentifier (1 total)
    collectionType: GOCollection (1 total)
```

El correspondiente a `GO:0000122` sería

```
gsc["GO:0000122"]
```

```
setName: GO:0000122
geneIds: 1316_at, 1552338_at, ..., AFFX-HUMISGF3A/M97935_MB_at (total:
    2280)
geneIdType: Annotation (hgu133plus2)
collectionType: GO
  ids: GO:0000122 (1 total)
  evidenceCode: EXP IDA IPI IMP IGI IEP HTP HDA HMP HGI HEP ISS ISO ISA
      ISM IGC IBA IBD IKR IRD RCA TAS NAS IC ND IEA
  ontology: CC MF BP
details: use 'details(object)'
```

Ahora vamos convertir estos mismos conjuntos a identificadores Entrez.

```
gsc = mapIdentifiers(gsc, EntrezIdentifier())
```

Podemos ver el primero de ellos.

```
head(geneIds(gsc),n=1)
```

```
$`GO:0000002`
 [1] "80119" "55186" "291" "4358" "1890"
 [6] "4205" "9361" "4976" "10000" "84275"
[11] "92667"
```

¿Cuántos elementos tiene cada uno de los conjuntos que hemos construido?

```
head(sapply(geneIds(gsc), length))
```

Podemos quedarnos con los dos grupos que tengan un cardinal mínimo, por ejemplo, por encima de 10.

```
gsc.filt = gsc[sapply(geneIds(gsc), length) > 10]
```

Son los siguientes grupos.

```
geneIds(gsc.filt)
```

12.3 Grupos con levadura

Veamos cómo construir grupos de genes utilizando los términos Gene Ontology en la Saccharomyces cerevisiae. Empezamos cargando el fichero de anotación.

```
pacman::p_load(org.Sc.sgd.db,GSEABase)
```

```
frame = toTable(org.Sc.sgdGO)
goframeData = data.frame(frame$go_id, frame$Evidence, frame$
    ↪ systematic_name)
goFrame = GOFrame(goframeData, organism = "Saccharomyces cerevisiae")
goAllFrame = GOAllFrame(goFrame)
gscSc_ORF = GeneSetCollection(goAllFrame, setType = GOCollection())
```

Habitualmente cuando queramos utilizar estos grupos tendremos que hacer dos cosas:

1. Quedarnos con aquellos genes que aparecen en nuestra plataforma.
2. Quedarnos posiblemente con grupos de genes con un tamaño mínimo.

Para ello vamos a utilizar la siguiente función.

```
subsettingGeneSet = function(gs0, fn0){
    geneIds(gs0) = geneIds(gs0)[is.element(geneIds(gs0), fn0)]
    gs0
}
```

Y ahora podemos modificar el GeneSetCollection de modo que nos quedamos con los genes que tenemos en nuestros datos `tamidata::gse6647`.

```
data(gse6647,package="tamidata")
```

```
gsc1 = sapply(gscSc_ORF, subsettingGeneSet, fn0 = featureNames(gse6647))
gsc2 = GeneSetCollection(gsc1)
```

Vamos a expresar la base de datos en identificadores ENTREZID. Empezamos considerando el total de genes que pertenecen a alguno de los grupos.

```
idsORF = unique(unlist(geneIds(gscSc_ORF)))
head(idsORF)
```

```
[1] "YDR150W" "YHR194W" "YLL001W" "YOL009C"
[5] "YOR147W" "YNL304W"
```

Obtenemos para todos estos identificadores `ORF` la correspondencia con los identificadores `ENTREZID`.

```
pacman::p_load(AnnotationDbi)
df = AnnotationDbi::select(org.Sc.sgd.db,keys=idsORF,
                           columns=c("ENTREZID"),
                           keytype="ORF")
```

Eliminamos posibles correspondencias múltiples entre los identificadores que hemos elegido: `ORF` de origen y `ENTREZID` de destino. Para ello definimos la siguiente función.

```
#' Choosing a pairs of correspondences
#' @description
#' Different gene identifiers (or proteins or ...) and we require
#' a unique correspondence
#' @param x \code{data.frame} with the different identifiers
#' @param coln \code{coln[,1]} is the original identifier and
#' \code{coln[,2]} is the new identifier
#' @export
multcorrespond = function(x,coln = c("PROBEID","ENTREZID")){
    x = x[,coln]
    x = na.omit(x)
    x = x[match(unique(x[,coln[1]]),x[,coln[1]]),]
    x = x[match(unique(x[,coln[2]]),x[,coln[2]]),]
    x
}
```

La aplicamos a nuestras correspondencias.

```
uc = multcorrespond(x=df,coln= c("ORF","ENTREZID"))
```

Y ahora, para cada grupo de genes, transformamos los identificadores.

```
coln= c("ORF","ENTREZID")
gscSc = lapply(1:length(gscSc),function(i)
    uc[match(geneIds(gscSc)[[i]],uc[,coln[1]]),coln[2]])
names(gscSc) = names(gscSc_ORF)
```

```
load(paste0(dirTamiData,"gscSc.rda"))
```

Podemos ver el primer grupo.

```
gscSc[1]
```

```
$`GO:0000001`
 [1] "851727" "856601" "850686" "854153" "854318"
 [6] "855412" "851249" "851359" "850887" "855318"
[11] "854079" "854504" "856249" "851558" "851532"
[16] "852825" "853033" "853146" "854748" "854867"
[21] "855094" "856198" "854668" "855645"
```

12.4 Grupos para GSE1397

Leemos los datos.

```
data(gse1397,package="tamidata")
```

Cargamos la anotación del `ExpressionSet` para poder formar grupos.

```
library(annotate)
annotation(gse1397)
```

```
[1] "hgu133a"
```

```
library(hgu133a.db)
```

Buscamos los conjuntos de genes utilizando Gene Ontology.

```
library(GSEABase)
```

```
gse1397.gsc = GeneSetCollection(gse1397,setType=GOCollection())
names(gse1397.gsc) = unlist(lapply(gse1397.gsc,setName))
```

¿Cuántos grupos tenemos definidos?

```
gsc = gse1397.gsc
length(gsc)
```

```
[1] 17492
```

Como vemos muchos. Sin embargo, la mayor parte de ellos tiene muy pocos genes. Podemos ver información del primer grupo.

```
(g1 = gsc[[1]])
```

```
setName: GO:0000002
geneIds: 201917_s_at, 201918_at, ..., 219393_s_at (total: 18)
geneIdType: Annotation (hgu133a)
collectionType: GO
  ids: GO:0000002 (1 total)
  evidenceCode: EXP IDA IPI IMP IGI IEP HTP HDA HMP HGI HEP ISS ISO ISA
      ↪  ISM IGC IBA IBD IKR IRD RCA TAS NAS IC ND IEA
  ontology: CC MF BP
details: use 'details(object)'
```

Y sus identificadores Affymetrix son

```
geneIds(g1)
```

```
 [1] "201917_s_at" "201918_at" "201919_at"
 [4] "202825_at" "203466_at" "204858_s_at"
 [7] "208328_s_at" "209017_s_at" "212213_x_at"
[10] "212214_at" "212535_at" "212607_at"
[13] "212609_s_at" "214306_at" "214684_at"
[16] "214821_at" "217497_at" "219393_s_at"
```

Sus identificadores Entrez o Ensembl son

```
unlist(mget(geneIds(g1),hgu133aENTREZID))
```

```
201917_s_at 201918_at 201919_at 202825_at
    "55186" "55186" "55186" "291"
  203466_at 204858_s_at 208328_s_at 209017_s_at
     "4358" "1890" "4205" "9361"
212213_x_at 212214_at 212535_at 212607_at
     "4976" "4976" "4205" "10000"
```

```
212609_s_at 214306_at 214684_at 214821_at
   "10000" "4976" "4205" "291"
  217497_at 219393_s_at
    "1890" "10000"
```

```
unlist(mget(geneIds(g1),hgu133aENSEMBL))
```

```
     201917_s_at 201918_at
"ENSG00000114120" "ENSG00000114120"
       201919_at 202825_at
"ENSG00000114120" "ENSG00000151729"
       203466_at 204858_s_at
"ENSG00000115204" "ENSG00000025708"
     208328_s_at 209017_s_at
"ENSG00000068305" "ENSG00000196365"
     212213_x_at 212214_at
"ENSG00000198836" "ENSG00000198836"
       212535_at 212607_at1
"ENSG00000068305" "ENSG00000117020"
      212607_at2 212609_s_at1
"ENSG00000275199" "ENSG00000117020"
    212609_s_at2 214306_at
"ENSG00000275199" "ENSG00000198836"
       214684_at 214821_at
"ENSG00000068305" "ENSG00000151729"
       217497_at 219393_s_at1
"ENSG00000025708" "ENSG00000117020"
    219393_s_at2
"ENSG00000275199"
```

La mayor parte de estos grupos son muy pequeños. Lo siguiente nos da el tamaño de estos grupos.

```
head(table(sapply(geneIds(gsc),length)))
```

```
  1 2 3 4 5 6
3057 2156 1710 1333 1015 757
```

De tamaño uno tenemos

```
sum(sapply(geneIds(gsc),length) == 1)
```

```
[1] 3057
```

Nos quedamos con aquellos grupos que, al menos, tienen 50 genes.

```
gse1397.gsc.filt = gsc[which(sapply(geneIds(gsc),length) > 50)]
```

12.5 Grupos utilizando anotación

12.5.1 Grupos GO para levadura

```
library(org.Sc.sgd.db)
frame = toTable(org.Sc.sgdGO)
goframeData = data.frame(frame$go_id, frame$Evidence, frame$
    ↪ systematic_name)
goFrame = GOFrame(goframeData, organism = "Saccharomyces cerevisiae")
goAllFrame = GOAllFrame(goFrame)
gscSc = GeneSetCollection(goAllFrame, setType = GOCollection())
save(gscSc,file=paste0(dirTamiData,"gscSc.rda"))
```

En lo que sigue será frecuente que no queramos utilizar todos los grupos que hemos construido. ¿Cómo quedarnos con aquellos que tienen al menos un número dado de genes? Por ejemplo, quedarnos con los grupos con 10 o más genes. El siguiente código lo hace.

```
gruposGrandes = which(sapply(geneIds(gscSc),length) > 10)
```

```
Error in h(simpleError(msg, call)): error in evaluating the argument '
    ↪ x' in selecting a method for function 'which': error in
    ↪ evaluating the argument 'X' in selecting a method for function
    ↪ 'sapply': unable to find an inherited method for function '
    ↪ geneIds' for signature '"list"'
```

```
gsc1 = gscSc[gruposGrandes]
```

```
Error in eval(expr, envir, enclos): objeto 'gruposGrandes' no
    ↪ encontrado
```

Podemos ver el primer grupo.

```
gsc1[[1]]
```

```
setName: GO:0000001
geneIds: YDR150W, YHR194W, ..., YNL079C (total: 24)
geneIdType:
collectionType: GO
  ids: GO:0000001 (1 total)
  evidenceCode: EXP IDA IPI IMP IGI IEP HTP HDA HMP HGI HEP ISS ISO ISA
      ↪  ISM IGC IBA IBD IKR IRD RCA TAS NAS IC ND IEA
  ontology: CC MF BP
details: use 'details(object)'
```

12.5.2 Grupos GO para humanos

En esta sección construimos los grupos de genes según Gene Ontology para humanos. Para otros organismos sería similar reemplazando el paquete [22] por el correspondiente al organismo. El código es el siguiente.

```
library("org.Hs.eg.db")
frame = toTable(org.Hs.egGO)
goframeData = data.frame(frame$go_id, frame$Evidence, frame$gene_id)
goFrame = GOFrame(goframeData, organism = "Homo sapiens")
goAllFrame = GOAllFrame(goFrame)
gscHs = GeneSetCollection(goAllFrame, setType = GOCollection())
save(gscHs,file = paste0(dirTamiData,"gscHs.rda"))
```

12.5.3 Grupos para Arabidopsis thaliana

Utilizamos el paquete [17].

```
pacman::p_load("ath1121501.db")
frame = toTable(org.At.tairGO)
goframeData = data.frame(frame$go_id, frame$Evidence, frame$gene_id)
goFrame = GOFrame(goframeData, organism = "Arabidopsis")
goAllFrame = GOAllFrame(goFrame)
gscAt = GeneSetCollection(goAllFrame, setType = GOCollection())
gscAt = geneIds(gscAt)
save(gscAt,file = paste0(dirTamiData,"gscAt.rda"))
```

12.6 Utilizando EnrichmentBrowser

Vamos a utilizar para obtener colecciones de genes la función `EnrichmentBrowser::getGenesets()`.

```
pacman::p_load(EnrichmentBrowser)
```

Podemos bajarnos todos las rutas de Gene Ontology para un organismo dado. Empezamos con humanos.

```
hsaGO = getGenesets(org="hsa",onto="BP")
save(hsaGO,file=paste0(dirTamiData,"hsaGO.rda"))
```

¿Qué tenemos?

```
class(hsaGO)
```

```
[1] "list"
```

Podemos ver sus nombres.

```
head(names(hsaGO))
```

```
[1] "GO:0000002_mitochondrial_genome_maintenance"
[2] "GO:0000003_reproduction"
[3] "GO:0000012_single_strand_break_repair"
[4] "GO:0000017_alpha-glucoside_transport"
[5] "GO:0000018_regulation_of_DNA_recombination"
[6] "GO:0000019_regulation_of_mitotic_recombination"
```

Y los genes que componen uno determinado con su código Entrez.

```
hsaGO[[3]]
```

```
 [1] "1161" "2074" "3981"
 [4] "7141" "7515" "23411"
 [7] "54840" "54840" "55775"
[10] "55775" "55775" "200558"
[13] "100133315"
```

```
names(hsaGO[3])
```

```
[1] "GO:0000012_single_strand_break_repair"
```

O bien con

```
hsaGO$"GO:0000012_single_strand_break_repair"
```

```
 [1] "1161" "2074" "3981"
 [4] "7141" "7515" "23411"
 [7] "54840" "54840" "55775"
[10] "55775" "55775" "200558"
[13] "100133315"
```

```
hsaGO["GO:0000012_single_strand_break_repair"]
```

```
 [1] "1161" "2074" "3981"
 [4] "7141" "7515" "23411"
 [7] "54840" "54840" "55775"
[10] "55775" "55775" "200558"
[13] "100133315"
```

12.7 Utilizando DOSE

```
pacman::p_load(clusterProfiler,DOSE,org.Hs.eg.db)
data(geneList, package="DOSE")
gene = names(geneList)[abs(geneList) > 2]
head(gene)
ggo = groupGO(gene = gene,
              OrgDb = org.Hs.eg.db,
              ont = "CC",
              level = 3,
              readable = TRUE)
```

```
head(ggo)
```

```
               ID Description
GO:0000133 GO:0000133 polarisome
GO:0000408 GO:0000408 EKC/KEOPS complex
GO:0000417 GO:0000417 HIR complex
GO:0000444 GO:0000444 MIS12/MIND type complex
GO:0000808 GO:0000808 origin recognition complex
GO:0000930 GO:0000930 gamma-tubulin complex
           Count GeneRatio geneID
GO:0000133 0 0/207
GO:0000408 0 0/207
GO:0000417 0 0/207
GO:0000444 0 0/207
GO:0000808 0 0/207
GO:0000930 0 0/207
```

12.8 Ejercicios

Ex. 27 — Construir los grupos basados en Gene Ontology para el ratón (Mus musculus).

Ex. 28 — Construir los grupos basados en Gene Ontology y en KEGG para las cepas de la E. coli que puedas.

Ex. 29 — Pretendemos ver cómo construir un informe en html para un grupo de genes. Se pide:

1. Utilizando `EnrichmentBrowser::getGenesets` construir los grupos correspondientes a humanos en la ontología de Gene Ontology correspondiente a funciones moleculares.
2. Utilizando los nombres de la lista que nos devuelve construir un **`data.frame`** que tenga por primera columna los identificadores del grupo y por segunda columna la descripción.
3. Generar las direcciones URL a partir de los identificadores de grupo.
4. Utilizando [55] generar un informe en html en donde la primera columna del informe contenga los URL correspondientes a los grupos y la segunda columna sea la descripción de los mismos.
5. Repetir los apartados del 1 al 4 para las ontologías de Gene Ontology correspondientes a procesos biológicos y a componentes celulares.

Ex. 30 — Repetir los apartados del 1 al 4 del ejercicio 30 para los grupos KEGG.

Capítulo 13

Test de Fisher unilateral

En este tema mostramos la aplicación del test de Fisher a lo que se conoce como análisis de sobre representación. Una presentación más técnica la tenemos en § 6.5.4.

13.1 Test de Fisher

Por alguna razón (que desconozco) a este procedimiento se le llama en numerosas publicaciones de perfil biológico *test hipergeométrico*. [1] En lo tratado hasta este momento hemos obtenido una ordenación de los genes. Esta lista la hemos estudiado pretendiendo que cuando mayor sea la expresión diferencial del gen este aparezca antes en la lista. De este modo el primer gen es el *marginalmente* (o si se prefiere individualmente) tiene una mayor expresión diferencial y así sucesivamente. De hecho, el p-valor no es más que una medida de esa diferenciación. Una expresión muy utilizada en la literatura es que que hay una asociación entre el gen (su expresión) y el fenotipo.[75] De alguna forma el p-valor que obtenemos en cada test es una cuantificación de la asociación gen-fenotipo. Luego modificamos estos p-valores de forma que se tiene en cuenta todos los genes que simultáneamente se están estudiando. Obtenemos de este modo unos p-valores ajustados. Finalmente, tanto los p-valores originales como los ajustados no dejan de ser cuantificaciones marginales de la asociación gen-fenotipo. Cuando hemos fijado una tasa de error (FDR o FWER) lo que hacemos es fijar un punto de corte. En esa lista que hemos construido decidimos (con algún criterio de error) en qué punto de la lista cortamos. Los genes que están antes del punto de corte se consideran significativos y los que siguen no. Reducimos nuestra información a un si (gen significativo o que tiene expresión diferencial o que hay asociación gen-fenotipo[2] o un no (no es significativo, no hay expresión diferencial o no hay asociación gen-fenotipo).

[75] La expresión fenotipo se usa de un modo muy amplio porque podemos estar hablando de características fenotípicas o simplemente un diseño experimental con factores temporales o diferentes temperaturas.

Ya tenemos ese conjunto de genes significativos. Incluso hemos visto cómo generar unos enlaces para que gen a gen examinemos alguna base de datos online. Es claro que el investigador puede ir generando hipótesis sobre qué indica esta lista. Pero claramente no es una labor simple. Una posible ayuda puede ser utilizar información previamente

[1] La distribución del estadístico de contraste es la distribución hipergeométrica. Hasta ahí llego. Lo que no entiendo es el cambio de nombre ya que la denominación test de Fisher está más que consolidada y debiera de utilizarse.

[2] A gusto del consumidor.

Tabla 13.1: S_0 indica el grupo de genes significativos (grupo predefinido) y $G \setminus S_0$ su complementario respecto del universo de genes considerado G. S_1 indica el conjunto de genes contra el cual comparamos.

	S_1	$S_1^c = G \setminus S_1$	
S_0	n_{11}	n_{12}	$n_{1\cdot}$
$S_0^c = G \setminus S_0$	n_{21}	n_{22}	$n_{2\cdot}$
	$n_{\cdot 1}$	$n_{\cdot 2}$	N

generada por la comunidad científica sobre grupos de genes. Grupos que indican que tienen relación con una misma función, o que están localizados próximos en un mismo cromosoma. En fin, grupos con sentido (biológico).

Un poco de notación que nunca es mala. Sea G es el conjunto de genes considerado. ¿Quién es este conjunto? En un estudio con microarrays uno diría que es el conjunto de genes que se está explorando. Sin embargo, un chip suele estar diseñado para observar tanto genes como se pueda y no hay una selección previa. Quizás no sea muy razonable considerar el conjunto total de genes este. En otros casos no es así. Lo fundamental es darse cuenta que lo que hacemos depende de un modo esencial del conjunto total de genes considerado o universo. S_0 ($\subset G$) el conjunto de genes que *nuestro* estudio ha indicado como significativo y S_1 un conjunto de genes predefinido (misma función, misma localización). Un primer problema es definir los conjuntos S_1 con los que comparar.[76] Una vez definidos *nuestro conjunto* (S_0), el conjunto con el que queremos comparar (S_1) y el conjunto total de genes considerado (G) la situación que se presenta la tenemos reflejada en la tabla 13.1. En esta tabla n_{11} indica el número de genes que están tanto en S_0 como en S_1, también tendremos n_{12} genes que están en S_0 pero no en S_1, n_{21} en S_1 pero no en S_0 y, finalmente, n_{22} que no están ni en S_0 ni en S_1. Suponemos que el total de genes (nuestro universo de genes) G tiene un total de N genes.

[76] § 12.

En la tabla 13.1 consideramos dados los totales de la fila y la columna, en otras palabras, consideramos fijos los valores de $n_{1\cdot}$ y $n_{2\cdot}$ (totales de fila) así como los valores de $n_{\cdot 1}$ y $n_{\cdot 2}$ (totales de columna). Asumiendo fijos estos totales de fila y columna: ¿cuál es la probabilidad de observar la tabla 13.1? Utilizando argumentos combinatorios la respuesta es la siguiente: Si denotamos N_{11} el número *aleatorio* de genes en común entonces, bajo la hipótesis de que no hay ningún tipo de asociación entre fila y columna, entonces la probabilidad sería

$$P(N_{11} = n_{11}) = \frac{\binom{n_{1\cdot}}{n_{11}}\binom{n_{2\cdot}}{n_{\cdot 1}-n_{11}}}{\binom{N}{n_{\cdot 1}}}.$$

Supongamos que estamos contrastando la posible sobrerepresentación entonces, bajo la hipótesis de independencia (condicionada a las marginales), rechazaríamos la hipótesis de independencia para un valor mayor o igual al observado por lo que el p-valor sería la suma de las probabilidades siguientes

$$p = P(N_{11} \geq n_{11}) = \sum_{t=n_{11}}^{\min\{n_{1\cdot},n_{\cdot 1}\}} \frac{\binom{n_{1\cdot}}{t}\binom{n_{2\cdot}}{n_{\cdot 1}-t}}{\binom{N}{n_{\cdot 1}}}.$$

Tabla 13.2: S_0 (S) indica el grupo de genes significativos (grupo predefinido) y S_0 (S^c) su complementario. S_1 indica el conjunto de genes contra el cual comparamos.

	S_1	S_1^c	
S_0	30	40	70
S_0^c	120	156	276
	150	196	346

Supongamos que hemos observado la tabla 13.2 y pretendemos saber si el solapamiento entre ambos conjuntos de genes es mayor que el esperable por el puro azar aunque no estén asociados ambos conjuntos.

Podemos utilizar el test exacto de Fisher direccional (o unilateral o de una cola).

```
conteos = matrix(c(30,120,40,156),ncol=2)
fisher.test(conteos,alternative = "greater")
```

```
      Fisher's Exact Test for Count Data

data:  conteos
p-value = 0.589
alternative hypothesis: true odds ratio is greater than 1
95 percent confidence interval:
 0.6018802       Inf
sample estimates:
odds ratio
 0.9750673
```

¿Y si valoramos una baja representación?

```
fisher.test(conteos,alternative = "less")
```

```
      Fisher's Exact Test for Count Data

data:  conteos
p-value = 0.5179
alternative hypothesis: true odds ratio is less than 1
95 percent confidence interval:
 0.000000 1.571751
sample estimates:
odds ratio
 0.9750673
```

Observamos que en ambas salidas nos muestra el cociente de los odds. Un valor del cociente de odds mayor que la unidad indica sobre expresión mientras que un valor menor a la unidad indica una expresión menor a la esperable bajo independencia.

13.2 Sobre la elección del universo de genes

Pretendemos evaluar el efecto que tiene la elección del universo de genes. ¿Qué efecto tiene en nuestro análisis el universo de genes, el conjunto G total de genes que estamos utilizando? Consideremos los datos de la tabla 13.3. ¿Qué describe la tabla? Supongamos que tenemos dos conjuntos de genes dados, S_0 y S_1. Dado este par de

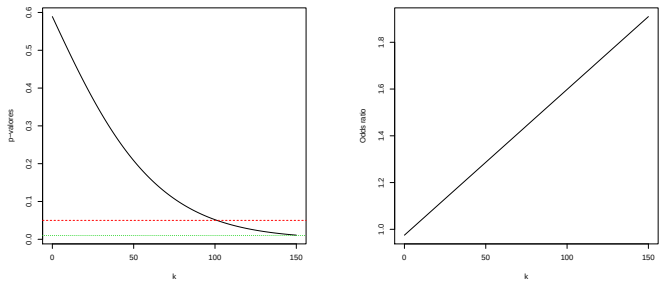
Tabla 13.3: S_0 (respectivamente S_1) indica el grupo de genes significativos (el grupo predefinido) y $G \setminus S_0$ ($G \setminus S_1$) su complementario. El universo de genes crece y suponemos que añadimos k genes.

	S_1	$G \setminus S_1$	
S_0	30	40	70
$G \setminus S_0$	120	156 + k	276 + k
	150	196 + k	346 + k

grupos de genes tenemos el número de genes en los dos y en uno de ellos pero no en el otro. Si miramos la tabla 13.3 significa que tenemos perfectamente definidas tres entradas de la tabla. Pero si suponemos que vamos incrementando nuestro universo de genes (sin incluir ningún gen adicional en S_0 o S_1) entonces la entrada n_{22} en la tabla 13.3 será cada vez mayor. En concreto hemos supuesto que $n_{22} = 156 + k$, es decir, el conteo original más los k genes que vamos incorporando al universo de genes. Vamos a tomar un valor de k creciente y representaremos el p-valor del test de Fisher unilateral. Vemos cómo conforme el valor de k crece el p-valor decrece (figura 13.1(a)) y el cociente de odds crece (figura 13.1(b)). En definitiva la interpretación tanto del p-valor como del cociente de odds depende del universo de genes que estamos considerando.

En la figura 13.1(a) (respectivamente en la figura 13.1(b)) tenemos el p-valor del test de Fisher unilateral correspondiente a la tabla 13.3 (respectivamente el cociente de odds) cuando el valor de k va 0 a 100. En trazo rojo discontinuo representamos la línea horizontal para un valor de ordenada igual a 0.05. La línea verde punteada corresponde con una ordenada de 0.01.

Vemos cómo *el incremento del universo de genes tiene como consecuencia que el p-valor del test de Fisher unilateral decrezca.*



(a) (b)

Figura 13.1: a) p-valores del test de Fisher unilateral. b) Cocientes de odds.

13.3 Utilizando Category y GOstats

En esta sección utilizamos los paquetes [37, GOstats] (ver también [38]) y [45, Category].

13.3.1 GSE1397

Leemos los datos normalizados.

```
pacman::p_load(Biobase)
data(gse1397,package = "tamidata")
eset = gse1397
y = pData(eset)[,"type"]
remove(gse1397) ## Ahorramos memoria
```

Determinamos grupo de genes significativos.

```
pacman::p_load(genefilter,multtest)
tt = rowttests(eset,y)
p0 = tt$p.value
p1 = mt.rawp2adjp(p0, "BH")
orden.original = order(p1$index)
p.BH = p1$adjp[orden.original,2]
significativos = which(p.BH < 0.05)
```

¿Qué anotación tienen?[77]

[77] Es una base de datos asociada a un chip.

```
annotation(eset)
```

```
[1] "hgu133a"
```

Cargamos el paquete de anotación correspondiente.

```
library(hgu133a.db)
```

¿Tenemos un solo identicador Gene Ontology para cada gen?

```
G1.entreizd = unlist(mget(featureNames(eset), hgu133aENTREZID))
anyDuplicated(G1.entreizd)
```

```
[1] 30
```

Hay duplicados. Nos quedamos, para cada gen, con el conjunto de sondas que nos da la máxima variabilidad, por ejemplo, el valor más grande del rango intercuartílico.

```
eset.iqr = apply(exprs(eset), 1, IQR)
uniqGenes = findLargest(featureNames(eset), eset.iqr, "hgu133a")
eset1 = eset[uniqGenes, ]
```

Y ahora construimos el universo de genes y comprobamos que no hay duplicidades.

```
G2.entrezid = unlist(mget(featureNames(eset1), hgu133aENTREZID))
anyDuplicated(G2.entrezid)
```

```
[1] 0
```

Determinamos la identificación en la misma base de datos de los genes declarados significativos.

```
seleccionados = unlist(mget(featureNames(eset[significativos,]),
    hgu133aENTREZID))
```

Vamos a aplicar un test de Fisher unilateral para los grupos definidos de acuerdo con Gene Ontology. Cargamos paquetes necesarios.

```
pacman::p_load(GO.db,Category,GOstats)
```

Y realizamos los tests.[78]

[78] En http://geneontology.org/ tenemos una aplicación en línea que nos realiza un análisis similar.

```
params = new("GOHyperGParams", geneIds = seleccionados,
             universeGeneIds = G2.entrezid,
             annotation = annotation(eset), ontology = "BP",
             pvalueCutoff = 0.001,conditional = FALSE,
             testDirection = "over")
overRepresented = hyperGTest(params)
```

Finalmente para visualizar los resultados guardamos los resultados en un fichero y lo vemos con el navegador.

```
htmlReport(overRepresented, file = "GSE1397overRepresented.html")
browseURL("GSE1397overRepresented.html")
```

También podemos ver un resumen.

```
head(summary(overRepresented))
```

Con el siguiente código obtenemos un grafo donde los vértices corresponden a los grupos definidos por la categorías Gene Ontology significativas. Las aristas nos muestran la estructura de la base de datos que es un grafo acíclico dirigido.

```
library(Rgraphviz)
plot(goDag(overRepresented))
```

```
library(Rgraphviz)
png(file=paste(dirTamiFigures,"GSE1397overRepresented.png",sep=""))
plot(goDag(overRepresented))
dev.off()
```

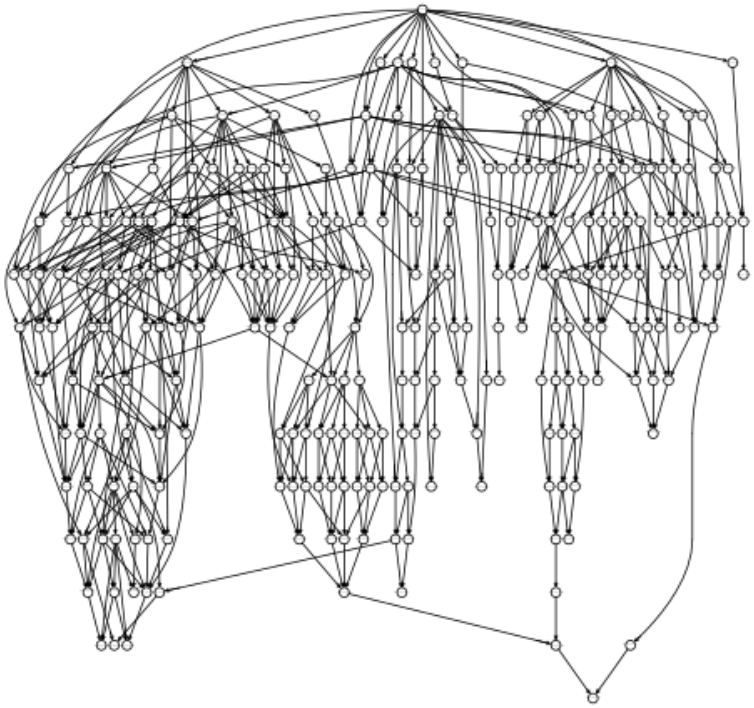


Figura 13.2: Grafo Gene Ontology con las grupos significativos.

13.3.2 GSE20986

Leemos los datos normalizados y lo renombramos.

```
data(gse20986, package = "tamidata")
```

En lo que sigue necesitamos la anotación de este `ExpressionSet`.

```
Biobase::annotation(gse20986)
```

```
[1] "hgu133plus2"
```

Cargamos el paquete con la base de datos correspondiente.

```
library("hgu133plus2.db")
```

Lo primero: **definir el universo de genes**. ¿Cómo? Una propuesta razonable[3] podría ser aplicar un filtrado no específico y que el universo de genes sean los restantes.

```
library(genefilter)
gse.filt = genefilter::nsFilter(gse20986, var.func = IQR, var.cutoff = 0.6,
                                require.GOBP = TRUE)$eset
```

¿Cuántos genes nos quedan?

```
dim(gse.filt)
```

```
Error in eval(expr, envir, enclos): objeto 'gse.filt' no encontrado
```

¿Tenemos un solo identicador Gene Ontology para cada gen?

```
G1.entreizd = unlist(mget(featureNames(gse.filt), hgu133plus2ENTREZID))
```

```
Error in h(simpleError(msg, call)): error in evaluating the argument '
    ↪ x' in selecting a method for function 'mget': error in
    ↪ evaluating the argument 'object' in selecting a method for
    ↪ function 'featureNames': objeto 'gse.filt' no encontrado
```

```
anyDuplicated(G1.entreizd)
```

```
[1] 30
```

Vemos que no hay duplicidades en los datos filtrados. Sin embargo, si consideramos como universo de genes todos los del chip entonces sí que nos encontramos con duplicidades.

```
G2.entrezid = unlist(mget(featureNames(gse20986), hgu133plus2ENTREZID))
anyDuplicated(G2.entrezid)
```

```
[1] 19
```

La idea sería quedarnos para cada gen con el conjunto de sondas que nos da la máxima variabilidad. Por ejemplo, con el grupo de sondas con el mayor rango intercuartílico.

```
gse.iqr = apply(exprs(gse20986), 1, IQR)
uniqGenes = findLargest(featureNames(gse20986), gse.iqr, "hgu133plus2")
gse.filt2 = gse20986[uniqGenes, ]
```

Y ahora construimos el universo de genes y comprobamos que no hay duplicidades.

[3] Y nada más que esto razonable y tan razonable como muchas otras.

```
G2.entrezid = unlist(mget(featureNames(gse.filt2), hgu133plus2ENTREZID))
anyDuplicated(G2.entrezid)
```

```
[1] 996
```

Aplicamos el procedimiento de expresión diferencial utilizado en el ejemplo **??**.

```
library(multtest)
gse.aov = rowFtests(gse.filt2, pData(gse20986)[,"tissue"])
p.originales = gse.aov[, 2]
p.BH = mt.rawp2adjp(p.originales, "BH")
pvalores = p.BH$adjp[p.BH$index, 2]
sig.1234 = which(pvalores < 0.001)
selected = unlist(mget(featureNames(gse.filt2[sig.1234,]),
    ↪ hgu133plus2ENTREZID))
```

Vamos a aplicar un test de Fisher unilateral para los grupos definidos de acuerdo con Gene Ontology. Cargamos paquetes necesarios.

```
pacman::p_load(GO.db,Category,GOstats)
```

Y realizamos los tests.

```
params = new("GOHyperGParams", geneIds = selected,
             universeGeneIds = G2.entrezid,
             annotation = annotation(gse.filt2), ontology = "BP",
             pvalueCutoff = 0.01,
             conditional = FALSE, testDirection = "over")
overRepresented = hyperGTest(params)
```

Finalmente para visualizar los resultados guardamos los resultados en un fichero y lo vemos con el navegador.

```
fl = tempfile()
htmlReport(overRepresented, file = fl)
browseURL(fl)
```

13.3.3 ALL

Esta sección se reproduce el análisis propuesto en http://www.bioconductor.org/help/course-materials/2009/SSCMay09/gsea/HyperG_Lecture.pdf. Utilizamos los datos [59, ALL]. En § 2.3.1 se indica cómo conseguir los datos `bcrneg` en donde hemos seleccionado algunas muestras. Los datos los tenemos en `bcrneg`. Veamos número de genes y muestras.

```
dim(bcrneg)
```

```
Features Samples
   12625 79
```

Realizamos un filtrado no específico quitando aquellos cuyo rango intercuartílico esté por debajo de la mediana (de los rangos intercuartílicos observados). Además se requiere que el gen tenga anotación en Gene Ontology. Esto lo pedimos con el argumento `require.GOBP = TRUE`.[4]

```
bcrneg.filt = nsFilter(bcrneg, var.cutoff = 0.5, require.GOBP = TRUE)$eset
```

Veamos qué nos queda.

[4] En concreto se pide su anotación en la ontología *Biological Process*.

Tabla 13.4: Tabla de contingencia para el término GO:0006468.

	InGO	
Selected	FALSE	TRUE
FALSE	3101	162
TRUE	658	30

```
dim(bcrneg.filt)
```

```
Features Samples
   4164 79
```

Vamos a generar una lista de genes significativos de un modo muy básico[5]. Calculamos el p-valor del test de la t y nos quedamos con los genes con un p-valor por debajo de 0.05.

```
fac0 = pData(bcrneg.filt)[, "mol.biol"]
fac0 = factor(fac0) ## Quitamos categorías vacías
rtt = rowttests(bcrneg.filt, fac0)
rttPrb = rtt$p.value
tThresh = rttPrb < 0.05
```

Le damos nombres a las componentes del vector utilizando los identificadores de Affymetrix GeneChip.

```
names(rttPrb) = featureNames(bcrneg.filt)
```

Guardamos estos identificadores en `ids`.

```
ids = featureNames(bcrneg.filt)
```

Guardamos también las correspondencias con Entrez.

```
map = hgu95av2ENTREZID
```

Definimos quién es nuestro universo. En nuestro caso todos los genes que intervenían en nuestro estudio y de los cuales teníamos su anotación (hay sondas de control que desaparecen con el filtrado que acabamos de hacer).

```
universe = unlist(mget(ids, map))
selected = unlist(mget(ids[tThresh], map))
```

Elegimos un grupo utilizando un término de Gene Ontology. Primero cargamos el paquete [18, GO.db].

```
library(GO.db)
```

Nos fijamos por ejemplo en el término siguiente

```
GOTERM[["GO:0006468"]]
```

```
GOID: GO:0006468
Term: protein phosphorylation
Ontology: BP
Definition: The process of introducing a
   phosphate group on to a protein.
Synonym: protein amino acid phosphorylation
```

La tabla 13.4 muestra los conteos observados.

Cargamos los paquetes [45, Category] y [37, GOstats].

[5] Y tampoco muy recomendable.

```
pacman::p_load(Category,GOstats)
```

```
params = new("GOHyperGParams", geneIds = selected,
             universeGeneIds = universe,
             annotation = annotation(bcrneg.filt),
             ontology = "BP", pvalueCutoff = 0.001,
             conditional = FALSE, testDirection = "over")
overRepresented = hyperGTest(params)
```

Veamos el resumen.

```
head(summary(overRepresented), n = 3)
```

Guardamos los resultados en un fichero HTML y lo vemos.

```
fl = tempfile()
htmlReport(overRepresented, file = fl)
browseURL(fl)
```

13.4 EnrichmentBrowser::sbea

Vamos a realizar un análisis de sobre representación utilizando `EnrichmentBrowser::sbea`.

```
pacman::p_load(EnrichmentBrowser,Biobase,SummarizedExperiment)
data(gse21942,package="tamidata")
```

Hemos transformar el `ExpressionSet` en un `SummarizedExperiment`.

```
se21942 = makeSummarizedExperimentFromExpressionSet(gse21942)
```

Podemos ver que los genes vienen identificados por los correspondientes al fabricante `AffyID` o `PROBEID`.

```
head(rowData(se21942),n=1)
```

```
DataFrame with 1 row and 7 columns
           PROBEID ENTREZID ENSEMBL
         <character> <character> <character>
1007_s_at 1007_s_at 780 ENSG00000204580
            SYMBOL GO EVIDENCE
         <character> <character> <character>
1007_s_at DDR1 GO:0001558 IEA
          ONTOLOGY
         <character>
1007_s_at BP
```

```
se21942=probe2gene(se21942)
```

¿Qué hemos hecho? Modificar los identificadores de los genes sustituyéndolos por los `ENTREZID`.

```
head(rowData(se21942))
```

```
DataFrame with 6 rows and 0 columns
```

Introducir una variable fenotípica `GROUP` con valores 0 y 1. Vamos a realizar un análisis de expresión diferencial utilizando el modelo Limma.

```
se21942 = deAna(expr = se21942) ## t-test moderados
head(rowData(se21942))
```

```
DataFrame with 6 rows and 4 columns
            FC limma.STAT PVAL ADJ.PVAL
      <numeric> <numeric> <numeric> <numeric>
780 0.18679961 2.8245963 0.00820538 0.0439322
5982 -0.20334770 -3.5661800 0.00119963 0.0104985
3310 -0.00376277 -0.0326465 0.97416583 0.9890259
7849 0.01222596 0.3640900 0.71826277 0.8519014
2978 0.05324267 1.3586019 0.18407863 0.3832102
7318 0.01396915 0.1890567 0.85128092 0.9258587
```

Utilizamos los grupos de genes Gene Ontology.

```
hsaGO = getGenesets("hsa",onto="BP") ## Biological processes
save(hsaGO,file=paste0(dirTamiData,"hsaGO.rda"))
```

Realizamos un análisis de sobre representación con el test de Fisher unilateral.

```
se21942.oraGO= sbea(method="ora", se=se21942, gs=hsaGO,
                    perm=0, alpha=0.05)
```

```
save(se21942.oraGO,file=paste0(dirTamiData,"se21942.oraGO.rda"))
```

```
load(paste0(dirTamiData,"se21942.oraGO.rda"))
```

Los resultados obtenidos son

```
gsRanking(se21942.oraGO)
```

```
DataFrame with 376 rows and 4 columns
              GENE.SET NR.GENES NR.SIG.GENES
            <character> <numeric> <numeric>
1 GO:0006120_mitochond.. 41 24
2 GO:0006974_cellular_.. 244 85
3 GO:0006886_intracell.. 228 78
4 GO:0006888_endoplasm.. 108 43
5 GO:0006396_RNA_proce.. 100 40
... ... ... ...
372 GO:2001241_positive_.. 10 5
373 GO:0008654_phospholi.. 23 9
374 GO:0006478_peptidyl-.. 2 2
375 GO:0008089_anterogra.. 30 11
376 GO:0000165_MAPK_casc.. 106 31
       PVAL
   <numeric>
1 4.45e-07
2 2.69e-06
3 1.44e-05
4 2.22e-05
5 3.73e-05
... ...
372 0.0481
373 0.0482
374 0.0485
375 0.0487
376 0.0497
```

13.5 ORA con clusterProfiler

Vamos a analizar los datos `tamidata::gse21942`. Preparamos los datos con el siguiente código (más detalles en 14.10).

```
data(gse21942,package="tamidata")
tt = genefilter::rowttests(gse21942,
                           pData(gse21942)$FactorValue..DISEASE.STATE.)
```

Vamos a considerar significativos aquellos genes cuyo p-valor ajustado por el método de Bonferroni sea menor o igual a 0.01.

```
sel = which(p.adjust(tt$p.value,method="bonferroni")<.01)
gene =fData(gse21942)$ENTREZID[sel]
gene = na.omit(gene)
gene = unique(gene)
```

El universo de genes que vamos a considerar será los contemplados en la plataforma.

```
universe = fData(gse21942)$ENTREZID
universe = na.omit(universe)
universe = unique(universe)
```

Hacemos un análisis de sobre representación con Gene Ontology.

```
pacman::p_load(clusterProfiler,org.Hs.eg.db)
ego = clusterProfiler::enrichGO(gene = gene,
                universe = universe,
                OrgDb = org.Hs.eg.db,
                ont = "CC",
                pAdjustMethod = "BH",
                pvalueCutoff = 0.01,
                qvalueCutoff = 0.05,
                readable = TRUE)
```

Para la base de datos KEGG.

```
kk = clusterProfiler::enrichKEGG(gene = gene,
                 organism = 'hsa',
                 pvalueCutoff = 0.05)
```

Para la base de datos KEGG Module.

```
mkk = enrichMKEGG(gene = gene,
                  organism = 'hsa',
                  pvalueCutoff = 1,
                  qvalueCutoff = 1)
```

13.6 Ejercicios

Ex. 31 — Utilizamos los datos `tamidata::gse20986`. En el problema 20 hemos determinado los genes significativos con un FDR de 0.05 para las comparaciones entre las muestras obtenidas en el iris, retina y coroides con las muestras huvec. Tenemos pues tres grupos de genes significativos que podemos denotar S_{iris}, S_{retina} y $S_{coroides}$.

1. Realizar, para cada uno de los tres grupos de genes seleccionados, un análisis de un sobre solapamiento, utilizando el test de Fisher unilateral, con los grupos definidos en Gene Ontology. En concreto hay que utilizar las tres bases de bases de Gene Ontology: BP (procesos biológicos), CC (componentes celulares) y MP (función molecular).

Ex. 32 — 1. Construir colecciones de grupos de genes utilizando [66] para el mus musculus. Para ello vamos a utilizar el paquete de anotación del organismo `org.Mm.eg.db`. Posiblemente ha de instalarse previamente.

2.Realiza un análisis de sobre representación con el paquete [41] utilizando la variable fenotípica `type`.

Ex. 33 — 1.Realizar el ejercicio 32 utilizando el paquete [99, clusterProfiler].

Capítulo 14

Análisis de conjuntos de genes

14.1 Introduction

[1]

El problema abordado en este capítulo corresponde a lo que se conoce como análisis de conjunto de genes.[79] Estudiamos si hay relación entre conjuntos de genes **previamente definidos** y fenotipo. Donde la expresión fenotipo tiene un sentido amplio como en todo el texto. Estos grupos vendrán definidos según distintos criterios. Por ejemplo, corresponder a una ruta metabólica, o localizados en un mismo cromosoma o bien definidos utilizando términos de Gene Ontology. De un modo genérico, un conjunto de genes que represente algo interpretable desde un punto de vista biológico. Ya no estamos interesados en un análisis *marginal* o *gen a gen* de los datos de expresión. En un análisis de expresión diferencial el resultado final es una lista ordenada de genes de modo que una mayor asociación con la covariable fenotípica produce una posición más alta en la lista. Un procedimiento de tests múltiples nos produce una clasificación en genes significativos y no significativos. En definitiva un valor de corte de la lista. En esta aproximación *no usamos ningún conocimiento previo sobre relaciones conocidas entre genes* que podrían ser esenciales a la hora de determinar la asociación no ya gen-fenotipo sino conjunto de genes-fenotipo.

[79] En la literatura se refieren a este problema como *gene set analysis*, *gene set enrichment analysis*, *set based enrichment analysis*.

14.2 Sobre la distribución de la matriz de expresión

Denotaremos la matriz de expresión (aleatoria) como

$$\boldsymbol{Y} = [Y_{ij}]_{i=1,\ldots,N;j=1,\ldots,n}$$

[1] Lo tratado en este capítulo tiene mucho que ver con la obra teatral de Lope de Vega, Fuenteovejuna. Se recomienda el video http://www.youtube.com/watch?v=-IcuFn57nAo. - *¿Quién mató al Comendador?*
- Fuenteovejuna, señor.
-¿Quién es Fuenteovejuna?
- Todo el pueblo, a una. Esta frase aparece repetida en la obra y es básica en este tema. Este tema va de esto, de la acción conjunta de un conjunto de genes. Unos pocos habitantes no pueden pero todos los habitantes de Fuenteovejuna sí que pueden.

donde Y_{ij} es la expresión aleatoria del i-ésimo gen en la j-ésima muestra. Las expresiones *observadas* serán

$$\boldsymbol{y} = [y_{ij}]_{i=1,\ldots,N;j=1,\ldots,n}$$

Notemos que, como es habitual, Y_{ij} es una variable aleatoria mientras que y_{ij} es un valor observado. Las columnas de la matriz de expresión Y son independientes. Son vectores aleatorios independientes[80] Por tanto podemos considerar que las distintas muestras son realizaciones de vectores independientes aunque no con la misma distribución ya que son observados bajo distintas condiciones experimentales. Si nos fijamos en las filas de la matriz de expresión, esto es, en los perfiles de expresión entonces tenemos realizaciones de vectores aleatorios dependientes y que además no tienen la misma distribución. Los genes no son independientes en su comportamiento hay relaciones entre ellos. Además tampoco tienen porqué comportarse de un modo (aleatorio) común.

[80] Corriendo un tupido velo por toda la parte de preprocesado de la información. Es claro que el preprocesado de la información introduce dependencias entre valores observados para distintas muestras pero las ignoramos. Con todo no se puede.

14.3 Conjunto(s) de genes

Tenemos distintos conjuntos de genes previamente definidos utilizando información previa: un grupo(s) definido por el grupo de investigación, grupos definidos utilizando Gene Ontology, ... Lo fundamental, estos grupos de genes no han de estar definidos en función de *nuestros* datos de expresión. Si denotamos por $G = \{1, \ldots, N\}$ el conjunto total de genes considerado (o universo de genes) entonces los conjuntos de genes serán $S_1, \ldots, S_K$. En particular, no es extraño considerar el caso $K = 1$, es decir, estar interesados en un conjunto de genes dado. Supondremos que el conjunto S_k tiene cardinal $|S_k|$. Los distintos conjuntos S_k **no** son una partición de G: no son necesariamente disjustos ($S_i \cap S_j \neq \emptyset$ para $i \neq j$) y su unión no tiene porqué ser todo el universo considerado de genes.

La cuestión básica podemos formularla como: *¿Hay asociación entre un conjunto de genes y el fenotipo?*. Es la misma pregunta que nos formulábamos para cada gen pero ahora referido a un conjunto dado de genes. Es una pregunta muy vaga. Se requiere una formulación más precisa. Caben distintas interpretaciones de la pregunta. En [86] formulan las siguientes dos hipótesis nulas que concretan de dos modos distintos la cuestión previa. Reproducimos las hipótesis nulas.[2]

Hipótesis Q1: Los genes de un conjunto muestran el mismo patrón de asociación con el genotipo comparado con el resto de genes.

Hipótesis Q2: El conjunto de genes no tiene ningún gen cuyo nivel de expresión está asociado con el fenotipo de interés.

No tenemos la misma hipótesis nula. La hipótesis nula **Q1** se centra en la comparación entre (la asociación entre) un conjunto dado de genes (con el fenotipo) y (la asociación entre) los otros (con el fenotipo).

[2]

Hypothesis Q1: The genes in a gene set show the same pattern of associations with the phenotype compared with the rest of the genes.

Hypothesis Q2: The gene set does not contain any genes whose expression levels are associated with the phenotype of interest.

En cambio, la hipótesis **Q2** se centra en el estudio de la expresión diferencial de los genes que pertenecen a un conjunto dado de genes.

Un planteamiento similar lo podemos encontrar en [50]. En concreto un test que se plantea si un conjunto de genes tiene una asociación con el fenotipo, la hipótesis **Q2**, recibe el nombre de *test autocontenido* (*self-contained test*) mientras que si nos ocupamos de la hipótesis nula **Q1** hablamos de un *test competitivo* (*competitive test*). De hecho el formula las hipótesis nulas del siguiente modo:

Hipótesis nula competitiva H_0^{comp}: Los genes en un grupo dado S están como mucho tan frecuentemente expresados de un modo diferencial como los genes en $S^c = G \setminus S$.

Hipótesis nula autocontenida H_0^{auto}: Ningún gen en S está diferencialmente expresado.

[3]

Muchos procedimientos estadísticos propuestos para contrastar estas hipótesis no formulan expresamente cuales son las hipótesis nulas que están contrastando. De hecho, la reflexión sobre el propio procedimiento de contraste propuesto es el que nos ha de indicar la hipótesis nula. No tenemos un modelo (estocástico) preciso y esto ocasiona esta indeterminación.

En lo que sigue veremos procedimientos que asumen una distribución conocida (o al menos asintóticamente conocida) para los estadísticos de contraste. Sin embargo, la opción más utilizada es evaluar la significación del estadístico asociado al contraste utilizando como distribución nula (o distribución bajo la hipótesis nula) una distribución de aleatorización o una distribución bootstrap. Qué distribución es adecuada dependerá de la hipótesis que estemos contrastando. De hecho, desde el punto de vista estadístico, este es el núcleo del problema. ¿Cómo estimamos la distribución nula?

14.4 Ejemplos

Es conveniente tener ejemplos artificiales para ilustrar y probar los procedimientos. En esta sección los comentamos.

14.4.1 Un ejemplo de Efron y Tibshirani

En [34] se proponen un par de ejemplos con datos simulados. Son los ejemplos 14.1 y 14.2.

Ejemplo 14.1. *Se consideran 1000 genes y 50 muestras. Se supone que las 25 primeras muestras son controles y las últimas 25 es el grupo de tratamiento. Definimos los grupos de genes como: las 20 primeras filas (de la matriz de expresión) corresponden al primer grupo, de la 21 a la 40 el segundo y así sucesivamente. Los niveles de expresión los generamos aleatoriamente independientemente y con la*

[3] Es interesante leer las hipótesis nula tal como las formulan:

Competitive null hypothesis H_0^{comp}: The genes in S are at most as often diferentially expressed as the genes in S^c.

Self-contained null hypothesis H_0^{self}: No genes in S are differentially expressed.

misma distribución. La distribución común es una normal estándar ($Y_{ij} \sim N(0,1)$). En el primer grupo añadimos un valor constante de 2.5 a los primeros 10 genes en las muestras correspondientes a tratamiento (últimas 25 columnas de la matriz de expresión). En consecuencia, lo que hacemos es que solamente el primer grupo tiene la mitad de los genes asociados con el fenotipo y la otra mitad sin ningún tipo de asociación. El resto de grupos no tiene ninguna asociación con fenotipo. Generamos estos datos.

```
N = 1000; n = 50
set.seed(280562) ## Para obtener los mismos valores generados
et1 = matrix(rnorm(N*n),nrow = N,ncol = n)
et1[1:10,26:50] = et1[1:10,26:50] + 2.5
```

En este ejemplo el vector que nos indica la clasificación de las muestras será

```
y.et = factor(rep(1:2,rep(25,2)),levels = 1:2,
  labels = c("Normal","Tratamiento"))
```

En lo que sigue utilizaremos como medida de asociación fenotipo-expresión el estadístico del t-test.

```
et1.tt = genefilter::rowttests(et1,y.et)$statistic
```

En figura 14.1 tenemos una estimación de la densidad. Podemos ver cómo se aprecia, por debajo de 8, el valor que corresponde al primer grupo.

```
pacman::p_load("ggplot2")
df = data.frame(et1.tt)
p = ggplot(df,aes(x=et1.tt))+geom_density()
ggsave(paste0(dirTamiFigures,"GeneSetAnalysis5b.png"),p)
```

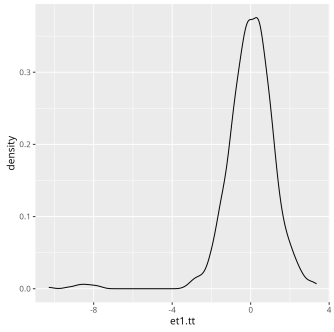


Figura 14.1: Densidad estimada de et1.tt.

Ejemplo 14.2. *Este ejemplo se define exactamente como el ejemplo 14.1 excepto lo que se hacía solamente para el primer grupo lo hacemos para todos: sumamos a los 10 primeros genes de cada grupo 2.5 unidades en las muestras correspondientes al tratamiento (últimas 25 columnas de la matriz de expresión). Generamos los datos.*

```
N = 1000; n = 50
set.seed(280562)
indices.temp = (0:49)*20 + 1
indices = NULL
for(i in indices.temp) indices = c(indices,i:(i+9))
et2 = matrix(rnorm(N*n),nrow = N,ncol = n)
et2[indices,26:50] = et1[1:10,26:50] + 2.5
```

Notemos que la covariable indicando el grupo es la misma que en ejemplo 14.1. Definimos los grupos para su uso posterior.

```
gen.name = function(i) paste0("g",((i-1)*20 +1):(i*20))
gsc.et = lapply(as.list(1:50),gen.name)
names(gsc.et) = paste0("set",as.character(1:50))
gsnames.et = paste0("set",as.character(1:50))
genenames.et = paste0("g",1:1000)
```

```
et2.tt = genefilter::rowttests(et2,y.et)$statistic
```

En figura 14.2 tenemos una estimación de la densidad.

```
df = data.frame(et2.tt)
p = ggplot(df,aes(x=et2.tt))+geom_density()
```

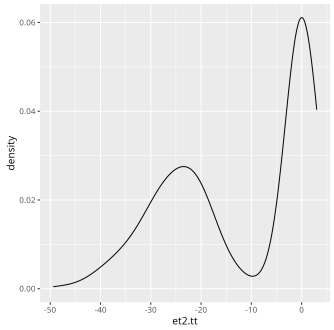


Figura 14.2: Densidad estimada

14.4.2 gse21942

Realizaremos un análisis de grupo de genes con los datos `tamidata ↪ ::gse21942`. Utilizaremos la colección de Gene Ontology utilizando el método visto en § 12.5.2.

```
pacman::p_load(GSEABase)
load(paste0(dirTamiData,"gscHs.rda"))
gscHs = geneIds(gscHs)
```

14.5 Cuantificando asociación gen-fenotipo

El primer paso es cuantificar la asociación entre los niveles de expresión de cada gen y el fenotipo. Esta cuantificación será un estadístico (una función de los datos que son las expresiones en la fila de la matriz de expresión) cuya definición dependerá del tipo de información fenotípica disponible. Por ejemplo, si tenemos dos condiciones entonces este estadístico será (habitualmente pero no siempre) el estadístico t que se utiliza en la comparación de medias de dos poblaciones normales. Pero no necesariamente, por ejemplo, si los tamaños muestras n_1 y n_2 son muy pequeños entonces muy probablemente un estadístico como la diferencia de medias es suficiente (y menos peligroso).

Si las muestras las tenemos clasificadas en más de dos grupos entonces podemos utilizar el valor del estadístico F del análisis de la varianza (**??**).

En **??** se estudiaba el método SAM (en el contexto de la expresión diferencial marginal). Este método propone, dependiendo de las descripción fenotípica con la que se trabaja, un estadístico d_i para cada gen. La definición de este estadístico es función del fenotipo. Son perfectamente utilizables como medidas de asociación gen-fenotipo (es lo que son).

En lo que sigue a la medida de asociación gen-fenotipo la denotaremos genéricamente por t_i (aunque no sea un t estadístico).

14.6 Enriqueciendo el conjunto de genes

El vector $\boldsymbol{t} = (t_1, \ldots, t_N)$ constituye, de hecho, una descripción de la asociación *marginal* gen-fenotipo. Pero tenemos unos grupos de genes en los que tenemos interés o que expresan simplemente conocimiento previo de posibles asociaciones entre los genes. Sea S uno de estos conjuntos. Enriquecer el estadístico t para el conjunto consiste en considerar un resumen de t sobre S. Y esto lo podemos hacer de muchas formas. Si denotamos el estadístico de enriquecimiento por $t(S)$ la opción más simple es

$$\boldsymbol{t}(S) = \sum_{i \in S} \frac{t_i}{n_S}, \tag{14.1}$$

es decir, el promedio de los t_i observados sobre el conjunto de genes de interés S. Podemos considerar muchas otras opciones y en las secciones que siguen las veremos.

14.7 Distribuciones condicionadas a los datos

Sea x la matrix de expresión observada. Y supongamos que el vector y (de dimensión n) nos indica la covariable de interés de la muestra (normalmente la pertenencia a un grupo). Para el universo de genes considerado tendremos el el vector $\boldsymbol{t} = (t_1, \ldots, t_N)$.

14.7.1 Distribución de permutación para un gen

Para un gen tenemos su perfil de expresión $u = (u_1, \ldots, u_n)$ y tenemos el vector y asociado a las muestras.

Sea π denota una permutación aleatoria de $(1, \ldots, n)$ de forma que los $\pi(i)$ será el índice que ocupa la posición i-ésima en la permutación π. En particular denotaremos por π_0 la permutación que nos devuelve el orden original: $\pi_0(j) = j$. Si el vector asociado a las muestras es y entonces tendremos el vector (permutado de y) $\boldsymbol{y}_\pi = (y_{\pi(1)}, \ldots, y_{\pi(n)})$. Obviamente $\boldsymbol{y} = y_{\pi_0}$.

La cantidad que describe la asociación entre u y $\boldsymbol{y}_\pi$ es t_π. Utilizando esta notación el valor observado de la asociación gen-fenotipo para u e y será t_{π_0}. Si consideramos B permutaciones aleatorias entonces tendremos los valores $t_{\pi_1}, \ldots, t_{\pi_B}$ para las B permutaciones aleatorias.

Si no hay asociación gen-fenotipo entonces el valor de t_{π_0} debe de ser *como* los valores $t_{\pi_1}, \ldots, t_{\pi_B}$. De hecho, cualquier ordenación de $t_{\pi_0}, t_{\pi_1}, \ldots, t_{\pi_B}$ tiene la misma probabilidad. Podemos considerar dos casos. En el primero una mayor asociación se expresa como un valor mayor de t (normalmente positivo). ¿Cuántos valores t_{π_b} (con $b = 1, \ldots, B$) son mayores que t_{π_0}? Si hay pocos indica que no hay equiprobabilidad y, por lo tanto, rechazamos esa hipótesis. Esto es un test de aleatorización aplicado a las muestras. El p-valor sería la proporción de t_{π_b}s mayores que t_{π_0}, es decir,

$$p_r = \frac{|\{b : t_{\pi_b} > t_{\pi_0}\}|}{B+1}. \tag{14.2}$$

En el caso en que una mayor asociación se exprese como un valor muy grande (positivo) o muy pequeño (negativo) de t entonces el p valor vendría dado por

$$p_r = \frac{|\{b : |t_{\pi_b}| > |t_{\pi_0}|\}|}{B+1}, \tag{14.3}$$

ya que tendríamos un test bilateral.

Ejemplo 14.3. *Consideremos el ejemplo 14.1, en concreto, los valores del primer gen. La covariable y nos indica la pertenencia a control o tratamiento. Como medida de asociación consideramos el estadístico t (con varianzas desiguales). Notemos que una asociación grande supone un valor o muy grande (positivo) o muy pequeño (negativo).*

```
u = et1[1,]
t0 = t.test(u ~y.et)$statistic
t0 = abs(t0)
```

Generamos $B = 100$ permutaciones aleatorias de y utilizando la función `sample`.

```
B = 100
tb = rep(0,B)
for(i in 1:B) tb[i] = t.test(u ~sample(y.et))$statistic
```

Determinamos los valores absolutos de los estadísticos.

```
tb = abs(tb)
```

El valor observado de t_{π_0} es 8.77. Y el p-valor será la proporción de los que su valor absoluto es mayor que el valor absoluto de t_{π_0}), es decir,

```
sum(tb > t0) / B+1
```

```
[1] 1
```

Tenemos un p-valor nulo. No parece que todas las ordenaciones de los valores $t_{\pi_0}, t_{\pi_1}, \ldots, t_{\pi_B}$ sean equiprobables.

La distribución nula que hemos considerado en esta sección corresponde con la hipótesis nula autocontenida que veíamos previamente. No asumimos ninguna distribución asintótica para el estadístico enriquecido y generamos una distribución nula en la que intercambiamos las etiquetas de las muestras. Si nuestro interés es contrastar la hipótesis Q2 o hipótesis autocontenida esta sería la distribución nula natural.[81]

[81] Podemos hablar de aleatorización por columnas entendiendo que las distintas muestras corresponden a las distintas columnas en la matriz de expresión.

14.7.2 Distribución de aleatorización

Denotemos por S_0 el conjunto de genes original en el que tenemos interés. Ahora vamos a elegir al azar un grupo S de genes (filas) del mismo cardinal que S_0. Lo aleatorio ahora no es el vector y sino el conjunto S de genes. Si S es aleatorio entonces también lo es $\boldsymbol{t}(S)$ (ahora las muestras tienen su ordenación original). A la distribución de probabilidad de $\boldsymbol{t}(S)$ se le llama **distribución de aleatorización** del estadístico de enriquecimiento. Si tomamos B selecciones aleatorias de n_{S_0} genes tendremos los conjuntos S_b con $b = 1, \ldots, B$ y los valores del estadístico de enriquecimiento observados son $\boldsymbol{t}(S_0)$ (para el grupo original) y $\boldsymbol{t}(S_b)$ con $b = 1, \ldots, B$ para los seleccionados al azar. El p-valor se define análogamente como

$$p = \frac{|\{b : t(S_b) > t(S_0)\}|}{B+1}. \tag{14.4}$$

si solamente rechazamos para valores grandes (positivos). En el caso bilateral tendremos

$$p = \frac{|\{b : |t(S_b)| > |t(S_0)|\}|}{B+1}. \tag{14.5}$$

La distribución nula que hemos considerado en esta sección corresponde con la hipótesis nula competitiva o hipótesis Q1. No asumimos ninguna distribución asintótica para el estadístico enriquecido y generamos una distribución nula en la que seleccionamos al azar grupos del mismo tamaño. Realmente lo que estamos haciendo es generar permutaciones aleatorias de las filas en la matriz de expresión. Si nuestro interés es contrastar la hipótesis Q1 o hipótesis competitiva esta sería la distribución nula natural.[82]

[82] Podemos hablar de aleatorización por filas entendiendo que los distintos genes (o genéricamente características que observamos) corresponden a las distintas filas en la matriz de expresión. Tam-

En lo que sigue repasamos distintos paquetes R/Bioconductor que implementan distintas medidas de enriquecimiento y distribuciones nulas en donde permutamos aleatoriamente las columnas (muestras) o las filas (características).

14.8 Usando Limma

14.8.1 genSetTest

En el paquete [81, limma] tenemos la función `limma::geneSetTest ↪ ()`. En este caso la medida de enriquecimiento que se utiliza es la media muestral de los estadísticos calculados para cada gen. Simplemente se permuta por filas. Contrastamos pues la hipótesis competitiva.

Ejemplo 14.4 (genSetTest y ejemplo 14.1)**.** *Consideremos los datos del ejemplo 14.1 y el t-estadístico para cada gen. El grupo de interés es el primero (primeras 20 filas). Supongamos que tomamos como alternativa si tienden a tomar valores mayores en el primer grupo.*

```
pacman::p_load(limma)
geneSetTest(1:20,et1.tt,alternative ="up")
```

```
[1] 0.9984722
```

El p-valor nos indica que no rechazamos la hipótesis nula. ¿Y si contrastamos la alternativa de valores mayores en el segundo grupo?

```
geneSetTest(1:20,et1.tt,alternative ="down")
```

```
[1] 0.00153168
```

Vamos a realizar el contraste para cada uno de los 50 grupos considerados.

```
pvalores = NULL
for(i in 0:49){
  indices = (i * 20 + 1): (i * 20 + 10)
  p.temp = geneSetTest(indices,et1.tt,alternative ="down")
  pvalores = c(pvalores,p.temp)
}
```

Comprobamos que el p-valor del primer grupo es claramente menor que el p-valor para los restantes grupos.

14.8.2 wilcoxGST

Siguiendo con el paquete [81, limma] una opción simple (no necesariamente muy potente) y que no utiliza la distribución de aleatorización consiste en tomar los valores del estadístico en el grupo de interés y en el resto de genes y compararlos utilizando un test de Wilcoxon. Por lo tanto estamos en el caso en que **conocemos** la distribución nula. No aleatorizamos ni por filas ni por columnas. Es un test exacto. El test de Wilcoxon tampoco asume ninguna hipótesis distribucional sobre los estadísticos por gen que utilizamos lo que es una ventaja. Sin embargo, asume independencia entre estos estadísticos que **no** se verifica ya que las expresiones de los genes son interdependientes.

Ejemplo 14.5 (Datos de ejemplo 14.1)**.** *Utilizamos la función limma::wilcoxGST.*

```
wilcoxGST(1:20,et1.tt,alternative ="down")
```

```
[1] 0.00153168
```

```
wilcoxGST(1:20,et1.tt,alternative ="up")
```

```
[1] 0.9984722
```

```
wilcoxGST(1:20,et1.tt,alternative ="mixed")
```

```
[1] 2.837697e-08
```

Tomemos otro grupo y repitamos el análisis.

```
wilcoxGST(21:30,et1.tt,alternative ="down")
```

```
[1] 0.09546351
```

```
wilcoxGST(21:30,et1.tt,alternative ="up")
```

```
[1] 0.904723
```

```
wilcoxGST(21:30,et1.tt,alternative ="mixed")
```

```
[1] 0.8181643
```

14.8.3 CAMERA

El método fue propuesto en [95].[83] Es un procedimiento para contrastar la hipótesis competitiva. Es un procedimiento que tiene en cuenta la correlación entre las expresiones de los distintos genes en cada muestra. La mayor parte de los procemientos propuestos para el contraste de la hipótesis competitiva suelen asumir (falsamente) la independencia de la expresión observada para distintos genes y su validez depende una hipótesis que sabemos no es cierta. Se ha visto que no tener en cuenta esta dependencia entre genes nos lleva a un incremento de la tasa de error FDR.

[83] El nombre es un acrónimo **C**orrelation **A**djusted **ME**an **RA**nk.

Se asume que la expresión está cuantificada en escala logarítmica (base 2 como es habitual en este contexto).

$$E[Y_{ij}] = \mu_{ij} = \sum_{k=1}^{p} \alpha_{ik} y_{jk} \tag{14.6}$$

siendo y_{jk} la k-ésima covariable (o variable fenotípica) de la j-ésima muestra. Se supone que las expresiones aleatorias (su perfil aleatorio de expresión) de un mismo gen tienen una varianza común σ_i^2.[84] Se asume que las expresiones para distintas muestras son independientes pero no entre distintos genes. En particular, denotamos el coeficiente de correlación de Pearson entre las variables aleatorias $Y_{i_1 j}$ y $Y_{i_2 j}$ como $cor(Y_{i_1 j}, Y_{i_2 j}) = \rho_{i_1, i_2}$. Obviamente no asumimos que estos coeficientes sean nulos.[85]

[84] Asumible si hemos aplicado previamente algún procedimiento de normalización.

[85] No asumimos incorrelación. Recordemos que independencia implica incorrelación pero no viceversa.

Un contraste, para el gen i-ésimo viene dado por

$$\beta_i = \sum_{k=1}^{p} c_j \alpha_{ik}.$$

Se está interesado en contrastar la hipótesis nula de que el contraste es nulo. Tenemos interés en los contrastes de hipótesis: $H_0 : \beta_i = 0$ frente a la alternativa de que $H_0 : \beta_i \neq 0$. Denotemos el estadístico del contraste como Z_i.[86] ¿Qué estadísticos Z_i vamos a considerar?

[86] No necesariamente con distribución normal.

14.9 GSA

[87] En el paquete [33, GSA] se implementa el método propuesto en [34].

[87] Se utiliza la distribución de permutación a la que aplican una reestandarización. La medida de enriquecimiento que proponen por defecto [4] es la siguiente: partimos de los valores t_i que miden asociación gen-fenotipo. Definimos $t^+ = \max\{t, 0\}$ y $t^- = -\min\{t, 0\}$. Consideramos un conjunto de genes S. Definimos $\bar{t}_S^+ = \sum_{i \in S} \frac{t_i^+}{n_S}$ y $\bar{t}_S^- = \sum_{i \in S} \frac{t_i^-}{n_S}$. Finalmente la medida de enriquecimiento (que llamaremos el estadístico `maxmean`) es

$$\boldsymbol{t}(S) = \max\{\bar{t}_S^+, \bar{t}_S^-\}. \tag{14.7}$$

Estamos tomando la media de las partes positivas t_i^+, la media de las partes negativas t_i^- y nos quedamos con el máximo de ambos valores.

Ejemplo 14.6. *[34, página 119] La medida de enriquecimento que acabamos de definir es robusta frente al caso en que tengamos una medida extrema. El ejemplo que proponen los autores es el siguiente: supongamos que tenemos 100 genes, 99 de los valores t_i's son -0.5 y el valor restante es 10. Tenemos que $\bar{t}_S^+ = 10/100 = 0.1$ y $\bar{t}_S^- = -99(-0.5)/100 = 0.495$. Vemos que los valores negativos dominan pues son la mayor parte de los datos. Si se tomara la media de las partes positivas t^+ y la media de las partes negativas t^- tendríamos los valores 10 y -0.5 (es decir, solamente consideramos cuando el valor no es nulo) y la medida de enriquecimiento vendría dominada por valores extremos.*

Como medidas de asociación gen-prototipo t_i utilizan las mismas del paquete [87, samr] que aparecen en la sección **??**. Cuando analizan los conjuntos de genes hablan de *grupos de genes positivos* y *grupos de genes negativos*. Un grupo de genes se dice negativo si corresponden con genes que en la clase 2 tiene expresiones menores cuando tenemos dos grupos (1 y 2). Si tenemos una covariable y numérica entonces los negativos corresponden al caso en que expresiones menores se asocian a valores mayores de y. Los grupos positivos se definen de modo contrario a los positivos. Cargamos el paquete.

```
pacman::p_load(GSA)
```

Ejemplo 14.7 (Ejemplo 14.1 con GSA)**.** *Empezamos analizando el ejemplo 14.1.*

```
grupo = c(rep(1,25),rep(2,25)) #El grupo tiene que numerarse 1,2
et1.gsa = GSA(et1,grupo, genenames=genenames.et, genesets=gsc.et,
  resp.type="Two class unpaired", nperms=100)
```

[4] Aunque lleva el promedio de los t_i's y el promedio de $|t_i|$ como otras opciones.

Podemos ver los estadísticos enriquecidos para los grupos

```
head(et1.gsa$GSA.scores)
```

```
[1] 3.99534187 -0.01231818 0.11759218
[4] -0.27081919 0.05557714 -0.07898232
```

El valor observado para el grupo 1 es 4 mientras que una descriptiva de los demás es la siguiente

```
summary(et1.gsa$GSA.scores[-1])
```

```
   Min. 1st Qu. Median Mean 3rd Qu.
-0.44625 -0.12429 -0.04239 -0.05241 0.05558
   Max.
 0.21225
```

Los p-valores obtenidos para lo que ellos llaman genes negativos que corresponden con genes que en la clase 2 tiene expresiones menores. Podemos obtenerlos con

```
head(et1.gsa$pvalues.lo)
```

```
[1] 1.00 0.41 0.55 0.07 0.62 0.32
```

Nuestro grupo 1 no destaca. Si considerados los p-valores de grupos positivos (en grupo 2 expresiones mayores) tenemos

```
head(et1.gsa$pvalues.hi)
```

```
[1] 0.00 0.59 0.45 0.93 0.38 0.68
```

que para el grupo 1 vale 0 y un resumen de los demás es

```
summary(et1.gsa$pvalues.hi[-1])
```

```
  Min. 1st Qu. Median Mean 3rd Qu. Max.
 0.1500 0.4500 0.5800 0.5776 0.7300 0.9700
```

Los valores originales t_i los tenemos con

```
head(et1.gsa$gene.scores)
```

```
[1] 3.485459 3.901185 3.698664 4.143949 3.555439
[6] 3.656246
```

En la figura 14.3 podemos ver un diagrama de cajas que compara el primer grupo con los demás.

```
fac0 = factor(c(rep(1,20),rep(2,980)),levels = 1:2,labels=c("Grupo 1","Resto"))
boxplot(et1.gsa$gene.scores ~fac0)
```

En el primer grupo tenemos la mitad de los genes diferenciados y la otra mitad no. En el resto de los genes no hay diferenciación. De ahí la gran variabilidad del primer grupo y que se solape por la parte inferior con el resto de los genes. El análisis de grupo que hemos realizado lo diferencia sin problemas.

En la figura 14.4 tenemos los grupos de genes que se considerarían significativos (diferenciando grupos up y down) y el valor de FDR para que los declaremos significativos en los datos `et1` *(ejemplo 14.1).*

```
GSA.plot(et1.gsa)
```

Vemos cómo solamente admitimos un grupo con un valor bajo de FDR.

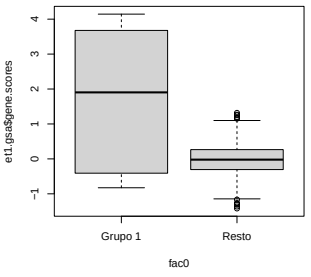


Figura 14.3: Valores t_i para el primer grupo (izquierda) y los demás. El primer grupo tiene diez genes claramente diferenciados mientras que los demás no lo están. De ahí la gran variabilidad que muestra el grupo.

Ejemplo 14.8 (Ejemplo 14.2 con GSA)**.** *En el ejemplo 14.2 todos los grupos considerados tienen el mismo comportamiento. Aunque todos tienen expresión diferencial, no hay un comportamiento diferenciado del grupo respecto de los otros grupos. Los datos son* `et2` *mientras que los grupos los tenemos en* `gsc.et` *con nombres en* `gsnames.et`*. Finalmente* `genenames.tt` *nos da los nombres de los genes.*

```
grupo = c(rep(1,25),rep(2,25))
et2.gsa = GSA(et2,grupo, genenames=genenames.et, genesets=gsc.et,
    resp.type="Two class unpaired", nperms=100)
```

La figura 14.5 muestra los grupos significativos en función de la FDR para los datos `et2` *(ejemplo 14.2).*

```
png(paste0(dirTamiFigures,"et2GSAplot.png"))
GSA.plot(et2.gsa)
dev.off()
```

```
pdf
  2
```

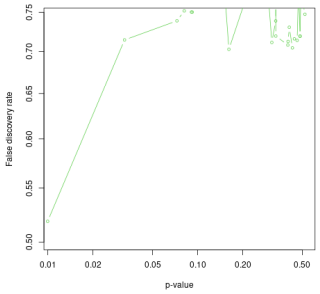


Figura 14.5: Grupos significativos con datos et2 en función de FDR.

Es claro que ningún grupo se diferencia de los demás.

Ejemplo 14.9 (Datos GSE1397 y GSA)**.** *Leemos los datos.*

```
data(gse1397,package="tamidata")
data(gse1397.gsc,package="tamidata")
gsc = gse1397.gsc
gruposGrandes = which(sapply(geneIds(gsc),length) > 50)
gsc = gsc[gruposGrandes]
gse = gse1397
```

Realizamos el análisis. Previamente convertimos la variable `tipo` *que es un factor a un vector numérico con valores 1 y 2 (es como lo pide [33, GSA].*

```
tipo.num = as.numeric(pData(gse)[,"type"])
```

Utilizamos como nombre de los genes los AffyId.

```
gse1397.gsa = GSA(exprs(gse),tipo.num, genenames=featureNames(gse),
  genesets=geneIds(gsc),resp.type="Two class unpaired", nperms=1000)
```

En la figura 14.6 podemos ver la representación de la tasa de falsos positivos como función del p-valor.

```
png(paste0(dirTamiFigures,"gse1397GSAplot.png"))
GSA.plot(gse1397.gsa)
dev.off()
```

```
pdf
  2
```

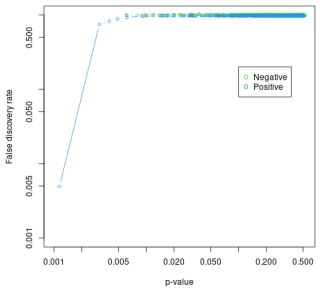


Figura 14.6: Grupos significativos con datos gse1397.

Fijamos una tasa de error de 0.05. Veamos qué grupos son los que o bien con una asociación negativa o bien con asociación positiva son los que presentan una mayor diferenciación entre los dos grupos considerados (con y sin síndrome de Down). Primero determinar los índices de los grupos en nuestra colección.

```
(ind.lo = which(gse1397.gsa$pvalues.lo <.05))
```

```
 [1]   7   8  38  56 103 105 119 138 180
[10] 220 257 262 281 351 356 418 426 438
[19] 454 459 478 494 543 566 569 575 608
[28] 610 613 655 684 857 910 972 1011 1012
[37] 1146 1161
```

```
(ind.hi = which(gse1397.gsa$pvalues.hi <.05))
```

```
 [1] 2 44 45 67 69 174 177 239 263
[10] 302 337 338 376 403 431 479 532 539
[19] 582 589 619 635 636 680 697 712 714
[28] 715 784 786 796 893 928 936 958 1055
[37] 1080 1098 1102 1123 1149 1151 1162
```

Podemos ver sus (primeros) identificadores en Gene Ontology. Para unos

```
head(names(gsc[ind.lo]))
```

```
  GO:0000002 GO:0000003 GO:0000012
"GO:0000022" "GO:0000023" "GO:0000183"
  GO:0000017 GO:0000018 GO:0000019
"GO:0000301" "GO:0000492" "GO:0000494"
```

y análogamente

```
head(names(gsc[ind.hi]))
```

```
  GO:0000002 GO:0000003 GO:0000012
"GO:0000003" "GO:0000245" "GO:0000255"
  GO:0000017 GO:0000018 GO:0000019
"GO:0000379" "GO:0000381" "GO:0001544"
```

Y ahora viene el trabajo del especialista para ver hasta qué punto lo que sale tiene sentido o no.

14.10 GSEA: Gene set enrichment analysis

La referencia básica es [83] que presenta una versión modificada del procedimiento originalmente propuesto en [65]. Nos sirve para contrastar la hipótesis competitiva y utiliza una modificación del test de Kolmogorov-Smirnov para dos muestras.

Método

Empezamos ordenando el universo de genes de acuerdo al grado de asociación gen-fenotipo utilizando algún estadístico Tendremos la lista de genes ordenada, L.

Entradas

1. La matriz de expresión X con N filas y n columnas.
2. Procedimiento de ordenación con objeto de producir la lista ordenada de genes, L.
3. Un valor p
4. Un conjunto de genes S.

Cálculo del enriquecimiento

1. Calculamos medida de asociación gen-fenotipo, t_i.
2. Ordenamos el universo de genes de acuerdo a las medidas de asociación del paso 1. Denotamos los índices ordenados con $r_1 \ldots, r_N$, es decir,

$$t_{r_1} \geq \ldots \geq t_{r_N}.$$

3. Calculamos para cada i con $i = 1, \ldots, N$

$$h_S(i) = \frac{1}{\sum_{r_i \in S} |t_i|^p} \sum_{j \leq i; r_j \in S} |t_j|^p, \tag{14.8}$$

Calculamos también

$$m_S(i) = \sum_{j \leq i; r_j \notin S} \frac{1}{N - n_S}, \tag{14.9}$$

El valor de e_S es la máxima desviación de cero de $h_S(i) - m_S(i)$, es decir,

$$e_S = \max_{1 \leq i \leq N} |h_S(i) - m_S(i)|. \tag{14.10}$$

Si elegimos S de un modo aleatorio (del universo de genes) entonces e_S tendrá un valor pequeño en relación a lo que se observa cuando S no es aleatorio bien concentrándose en la parte superior de la lista o en la inferior o siguiendo algún patrón no aleatorio. Si $p = 0$ entonces e_S es el estadístico del test de Kolmogorov-Smirnov.

Estimación del p-valor 1. Consideramos una asignación aleatoria del fenotipo a las muestras (una permutación aleatoria de las muestras manteniendo fijo el fenotipo), reordenamos los genes y calculamos el valor $E(S)$.

2. Se repite el paso anterior un gran número de veces.

3. Si e_0 es el valor de e_S sobre los datos originales y $e_1, \ldots, e_B$ son los valores de e_S para las asignaciones aleatorias de fenotipo a muestras entonces el p-valor viene dado por

$$p = \begin{cases} 2\frac{|\{e_i : e_i \geq e_0\}|}{B+1}, \text{ for } e_0 > 0, \\ 2\frac{|\{e_i : e_i \leq e_0\}|}{B+1} \text{ for } e_0 < 0. \end{cases}$$

Contrastes múltiples Suponemos que consideramos ahora una colección de conjuntos de genes de interés: $S_1, \ldots S_K$.

1. Calculamos e_{S_k} para $k = 1, \ldots, K$.

2. Para cada S_k y 1000 permutaciones fijas π_i con $i = 1, \ldots, 1000$ (las mismas para todos los conjuntos de genes) del fenotipo, reordenamos los genes y calculamos el enriquecimiento e_{S_k, π_i}.

3. Ajustamos por el tamaño variable de los conjuntos de genes. Para ello consideramos los valores

$$\bar{e}_+ = \sum_{k,i: e_{S_k,\pi_i} > 0} \frac{e_{S_k,\pi_i}}{|\{k, i : e_{S_k,\pi_i} > 0\}|},$$

y

$$\bar{e}_- = \sum_{k,i: e_{S_k,\pi_i} < 0} \frac{e_{S_k,\pi_i}}{|\{k, i : e_{S_k,\pi_i} < 0\}|}.$$

Consideramos los valores

$$\tilde{e}_0 = \begin{cases} \frac{e_0}{\bar{e}_+}, \text{ si } e_0 > 0, \\ \frac{e_0}{\bar{e}_-}, \text{ si } e_0 < 0. \end{cases}$$

y

$$\tilde{e}_{S_k,\pi_i} = \begin{cases} \frac{e_{S_k,\pi_i}}{\bar{e}_+}, & \text{si } e_{S_k,\pi_i} > 0, \\ \frac{e_{S_k,\pi_i}}{\bar{e}_-}, & \text{si } e_{S_k,\pi_i} < 0. \end{cases}$$

4. Consideremos un valor normalizado (según método de punto anterior) $\tilde{e}$. Se *estima* la tasa de falsos positivos para ese valor como: si $\tilde{e} > 0$

$$q(\tilde{e}) = \frac{|\{k,i : e_{S_k,\pi_i} \geq \tilde{e}\}|/|\{k,i : e_{S_k,\pi_i} \geq 0\}|}{|\{k : e_{S_k} \geq \tilde{e}\}|/|\{k : e_{S_k} \geq 0\}|}$$

Si $\tilde{e} < 0$ entonces

$$q(\tilde{e}) = \frac{|\{k,i : e_{S_k,\pi_i} \leq \tilde{e}\}|/|\{k,i : e_{S_k,\pi_i} \leq 0\}|}{|\{k : e_{S_k} \leq \tilde{e}\}|/|\{k : e_{S_k} \leq 0\}|}$$

Ejemplo 14.10. *Vamos a analizar los datos* `tamidata::gse21942` *con [99, clusterProfiler]. Vamos a construir un vector que almacene los estadísticos por gen y tenga como nombres de sus elementos sus identificadores* `Entrez`.

```
data(gse21942,package="tamidata")
tt = genefilter::rowttests(gse21942,
                           pData(gse21942)$FactorValue..DISEASE.STATE.)
tt3 = tt$statistic
names(tt3) = fData(gse21942)$ENTREZID
tt3 = na.omit(tt3)
```

Eliminamos duplicidades eligiendo una sonda por gen. Nos quedamos con la primera aparición.

```
tt3 = tt3[match(unique(names(tt3)),names(tt3))]
```

Hay que ordenar en orden decreciente según el estadístico que hemos calculado por gen.

```
tt3 = sort(tt3,decreasing=TRUE)
```

Guardamos los nombre de los genes.

```
gene = names(tt3)
```

La función `clusterProfiler::groupGO()` *nos va construir la colección de grupos de genes Gene Ontology. Lo podemos hacer siempre que tengamos un paquete* `OrgDb` *§ 18.3.*

```
pacman::p_load(clusterProfiler,org.Hs.eg.db)
ggo = clusterProfiler::groupGO(gene = gene,
              OrgDb = org.Hs.eg.db,
              ont = "CC",
              level = 3,
              readable = TRUE)
```

Podemos ver la información de los dos primeros grupos.

```
head(ggo,n=2)
```

```
               ID Description Count
GO:0000133 GO:0000133 polarisome 0
GO:0000408 GO:0000408 EKC/KEOPS complex 5
          GeneRatio
GO:0000133 0/21337
GO:0000408 5/21337
                                geneID
GO:0000133
GO:0000408 OSGEP/LAGE3/TP53RK/GON7/TPRKB
```

El método GSEA visto en esta sección lo podemos aplicar con la función `clusterProfiler::gseGO()`.

```
ego3 = clusterProfiler::gseGO(geneList = tt3,
              OrgDb = org.Hs.eg.db,
              ont = "CC",
              minGSSize = 100,
              maxGSSize = 500,
              pvalueCutoff = 0.05,
              verbose = FALSE)
```

```
Warning in fgseaMultilevel(pathways = pathways, stats = stats, minSize
    ↪ = minSize, : For some pathways, in reality P-values are less
    ↪ than 1e-10. You can set the `eps` argument to zero for better
    ↪ estimation.
```

También podemos hacer un análisis GSEA utilizando las rutas KEGG.

```
k3 = clusterProfiler::gseKEGG(geneList = tt3,
                              organism = 'hsa',
                              minGSSize = 120,
                              pvalueCutoff = 0.05,
                              verbose = FALSE)
```

```
Warning in fgseaMultilevel(pathways = pathways, stats = stats, minSize
    ↪ = minSize, : For some pathways, in reality P-values are less
    ↪ than 1e-10. You can set the `eps` argument to zero for better
    ↪ estimation.
```

```
head(k3,n=1)
```

```
            ID Description
hsa00190 hsa00190 Oxidative phosphorylation
        setSize enrichmentScore NES pvalue
hsa00190 127 0.6601883 2.689856 1e-10
           p.adjust qvalue rank
hsa00190 7.071429e-10 2.706767e-10 2540
                        leading_edge
hsa00190 tags=46%, list=12%, signal=41%




hsa00190 518/1351/4708/374291/4720/4722/4728/29796/126328/1345/9114/
    ↪ 27089/1349/7384/4697/523/526/4723/1337/1329/4714/4718/4694/4695
    ↪ /522/537/4726/51079/7388/4513/533/4716/51606/10312/4696/8992/
    ↪ 4712/4717/528/4713/527/9296/4710/7386/506/514/4707/155066/51382
    ↪ /4702/55967/4706/4537/4725/515/27068/9551/9377/1347
```

También podemos repetir el análisis para los módulos KEGG (otra colección dentro de KEGG).

```
mkk2 = gseMKEGG(geneList = tt3,
                organism = 'hsa',
                pvalueCutoff = 1)
head(mkk2,n=1)
```

```
         ID Description setSize
M00160 M00160 V-type ATPase, eukaryotes 24
      enrichmentScore NES pvalue
M00160 0.7659055 2.314657 3.016729e-07
         p.adjust qvalue rank
M00160 1.387695e-05 9.526513e-06 1632
                    leading_edge
```

```
M00160 tags=54%, list=8%, signal=50%
                                                    core_enrichment
M00160 9114/523/526/537/533/51606/10312/8992/528/527/9296/155066/51382
```

Podemos visualizar los conjuntos de genes que hemos visto que tienen un comportamiento diferenciado desde el punto de vista competitivo.[5]

```
clusterProfiler::browseKEGG(k3,'hsa00190')
```

[5] La interpretación de estas rutas se salen completamente de los limitados conocimientos biológicos del que escribe.

Parte VI

Investigación reproducible

Capítulo 15

Investigacion reproducible

Tratamos sobre **investigación reproducible**.[88] relacionado con la programación comentada[89] aunque no son sinónimos.

Estamos acostumbrados a que los investigadores[90] en sus publicaciones comenten los resultados obtenidos en sus investigaciones. Han obtenido unos datos, los han analizado y, finalmente, los han interpretado y discutido. *H*emos de creer que han producido los datos correctamente, que han aplicado un tratamiento adecuado (que han hecho lo que dicen que han hecho es un mínimo) y que, finalmente, la interpretación de los resultados de las técnicas utilizadas es correcta. Los demás hemos de creer en ellos. Que los distintos paso se han hecho bien. Al menos, lo mejor que han podido. Y que además el nivel de corrección en todas las etapas es suficiente. Si no tenemos los datos, el código y la interpretación: ¿cómo sabemos que el trabajo es correcto? Es una cuestión de fe. ¿Dónde están los datos? Exactamente, no aproximadamente: ¿qué análisis de los datos han realizado? En otras palabras, deme el código para que yo (y cualquier otra persona) pueda reproducir *exactamente* el tratamiento estadístico que se ha realizado. La interpretación es lo único que tenemos: es la publicación científica que nos ha dado la *n*oticia de la existencia de la investigación.

Por investigación reproducible entendemos procedimientos que permitan *reproducir* la investigación completa. En su totalidad. Y si esto es posible no se debiera de publicar. Algún ejemplo de lo que puede ocurrir si la transparencia no es total es https://www.theguardian.com/world/2020/jun/03/covid-19-surgisphere-who-world-health-organization-hydroxychloroquine?.

Personalmente añadiría que los distintos elementos han de ser totalmente libres. Los datos han de estar a disposición de la comunidad[91] y el software que se utiliza para realizar los análisis han de ser libres también. Quizás esta opción personal es radical pero creo en ella firmemente. [92]

Este manual es un ejemplo de lo que Yihui Xie [96] llama un **documento dinámico**. Es un documento en donde se combinan las explicaciones metodológicas junto con el código que las implementa. [93]

En este tipo de documentos se distinguen dos partes. La primera en el documento dónde se indica el título, autores e información sobre el formato del documento. Suele hacerse en formato YAML o bien en

[88] Reproducible research

[89] Literate programming

[90] En ciencias experimentales.

[91] Una vez se han publicado los trabajos.

[92] Todo lo que he usado en este manual es libre. No se ha utilizado sofware propietario ni datos que no estén a disposición de todos en algún repositorio público.

[93] Sobre la obtención de los datos hay que consultar las referencias bibliográficas.

en LaTeX.

[94] Obviamente no es castellano pero es cuestión de tipo que la RAE lo admita.

La segunda parte está compuesta por bloques (chunks) [94] de dos tipos: chunks de texto y chunks de código. Los chunks de texto se pueden escribir en Markdown § 15.1 o en LaTeX. Los chunks de código pueden usar distintos lenguajes de programación.

1. El task view en el repositorio de R http://cran.r-project.org/-web/views/ReproducibleResearch.html.

2. La página de Harrell http://biostat.mc.vanderbilt.edu/twiki/bin/view/Main/StatReport

3. Si se elige la opción de investigación reproducible utilizando LaTeXentonces una (muy) buena referencia [69].

4. Las referencias [47, 46].

En este documento nos centramos en R y por ello la opción para investigación reproducible es combinar R bien con Markdown bien con LaTeX. Sin embargo, en otros lenguajes hay otras opciones similares y tan buenas como esta. Es recomendable leer https://blog.ousef ul.info/2017/11/15/programming-meh-lets-teach-how-to-write-computational-essays-instead/.

15.1 Markdown

Es un lenguaje de marcas ligero (minimal sería más correcto). Pretende ser una forma rápida de escribir HTML. Según su autor, John Gruber, "HTML is a publishing format; Markdown is a writing format."

La dirección indicada contiene una exposición de este lenguaje de marcas. Al ser tan limitado en su sintaxis han aparecido diferentes variaciones que lo extienden (entre otras cosas para tablas) de las cuales la más interesante y usada es Pandoc Markdown. Pandoc Markdown.

15.2 Pandoc

Es un programa que convierte textos escritos en distintos lenguajes de marcas. Es absolutamente impresionante la cantidad de posibilidades que ofrece. En https://pandoc.org/ tenemos una detallada exposición de sus posibilidades. Está en la base [97, knitr], [6, rmarkdown] y quarto Quarto.

15.3 knitr

El paquete [97, knitr] ha sido desarrollado por Yihui Xie y supuso un punto de inflexión en investigación reproducible. Nos permite trabajar con Markdown y LaTeXen chunk de texto y con R como lenguaje de programación. Es una herramienta básica en RMarkdown y Quarto. Consultar https://yihui.org/knitr/. En [96] tenemos una detallada y amena exposición.

15.4 RMarkdown

El paquete [6, rmarkdown] incorpora una implementación del lenguaje de marcas Markdown y utilizado conjuntamente con [97, knitr] permite generar unos muy buenos informes. La página http://rmarkdown.rstudio.com/ es el mejor lugar para aprender a manejarlo.

15.5 Quarto

Es una evolución de § 15.4. Permite trabajar con distintos lenguajes de programación en los chunks de código. En particular, aquí nos interesa R y Bash aunque también incluye Python entre otros.

La sintaxis es muy similar a RMarkdown y en lo esencial las etiquetas que especifican el comportamiento de los chunks de código se incluyen dentro del chunk precedidas de `#\|`.

15.6 Entornos de desarrollo

Una vez elijamos una opción de trabajo para investigación reproducible necesitamos un entorno de desarrollo (IDE).

15.6.1 RStudio

RStudio es la mejor opción y la que tiene un manejo más sencillo cuando trabajamos con R y Markdown. También permite trabajar con R y LaTeXaunque no es tan buena la implementación.

15.6.2 emacs

En mi opinión la mejor opción de trabajo es esta y mi opción personal. Sin embargo, no es la más simple de usar. Hay un aprendizaje previo requerido. Es un editor de texto que puede ser configurado para trabajar con muchos lenguajes y para muchas utilidades. Su dirección principal de consulta es https://www.gnu.org/software/emacs/.

ESS Para el trabajo con R tenemos el modo **Emacs Speaks Statistics** (https://ess.r-project.org/).

RMarkdown Para trabajar con RMarkdown tenemos el modo polymode que podemos instalar en el subdirectorio `~/.emacs.d/` ejecutando

```
git clone https://github.com/vitoshka/polymode.git
```

Y luego incluimos en `.emacs` los caminos correspondientes así como los modos necesarios. Hay que añadir las siguientes líneas (sustituyendo el directorio personal por el correspondiente).

```
(add-to-list 'load-path "/home/gag/.emacs.d/polymode
    ↪ /")
(add-to-list 'load-path "/home/gag/.emacs.d/polymode
    ↪ /modes/")
(require 'poly-R)
(require 'poly-markdown)
```

Quarto Para trabajar con Quarto (§ 15.5) hay un `quarto-mode` de emacs en https://github.com/quarto-dev/quarto-emacs. En Debian/Ubuntu tenemos el paquete quarto que nos permite utilizarlo en la shell (por ejemplo, con la bash).

Capítulo 16

Generando un informe

En datos ómicos trabajamos con muchas variables y pocas muestras. Cuando analizamos su posible expresión diferencial marginal o por grupos, o bien cuando hacemos un análisis de asociación. En fin, cualquier análisis con centenares o miles de variables suele acabar con informes largos que contienen tablas interminables. No es un buen material para el experto que ha de interpretar para obtener unas conclusiones. Estas tablas están dando información sobre entidades biológicas (genes, exones, posiciones genómicas, grupos de genes, etc.) que no podemos conocer por su enorme cantidad. Y para cada una de estas entidades tenemos información (resuménes estadísticos, p-valores, etc.). ¿Cómo generar un informe útil? El informe que hemos de proporcionar al experto ha de ser **útil** (y no solo bonito con muchos colores). Posiblemente la mejor opción es generar archivos en HTML con enlaces a bases de datos donde nos proporcionan información sobre las entidades a que referimos nuestro análisis (genes, grupos de genes, SNPs, proteínas o lo que sea que estemos analizando). Por ejemplo, podemos querer enlazar los genes con Gene Ontology. Esto permitirá al experto interpretar los resultados con su navegador consultando una y otra vez rápidamente con las bases de datos. En la práctica estadística clásica se genera un informe (en formatos orientados a texto impreso como puede ser pdf) y el investigador interpreta los resultados y redacta un informe en un formato similar. En este contexto el problema es mayor. Básicamente se trabaja con grandes cantidades de variables y por ello los resultados han de ser validados y nuevas hipótesis generadas para posterior investigación. Un análisis estadístico de datos ómicos nunca es conclusivo. Ha de validarse con técnicas no ómicas pero con mayor precisión.

En este tema nos planteamos el siguiente problema: cómo generar un fichero centrado en las entidades biológicas de interés y que contenga enlaces a sus bases de datos en línea correspondientes y resúmenes estadísticos (p-valores originales, ajustados, q-valores, etc.) y gráficos (con moderación lo de los dibujos).

En § 16.1 vemos un análisis sencillo y cómo asociar distintos identificadores utilizando el paquete [42, annotate]. Construimos un `data.frame` con la información relevante del análisis. En § 16.2.1, § 16.2.2 y § 16.2.3 generamos ficheros HTML a partir del `data.frame` que hemos construido previamente utilizando los paquetes [55, ReportingTools], [98, DT] y [101, kableExtra] respectivamente.

16.1 Generando la información

Utilizamos como ejemplo los datos `tamidata::GSE20986` ??.

```
pacman::p_load(Biobase)
data(gse20986,package="tamidata")
```

Comparamos las muestras extraídas de células endoteliales vasculares de la retina y de células endoteliales de la vena umbilical (abreviadamente células de la retina y huvec). Con el siguiente código (§ 8 y § 9) aplicamos un t-test, a los p-valores obtenidos le aplicamos el método de Benjamini-Hochberg utilizando un FDR de 0.05. Los genes significativos son los que utilizamos en el resto de tema para ilustrar la generación de un informe a partir de ellos.

```
eset = gse20986[, c(2, 3, 5, 10:12)]
tissue = factor(rep(1:2, each = 3), levels = 1:2,labels = c("retina", "huvec"))
tt = genefilter::rowttests(eset, tissue)
padj = p.adjust(tt[,"p.value"],method="BH")
sig = which(padj < 0.05)
```

En el vector `sig` tenemos los índices de las filas que ocupan, en la matriz de expresión, los genes declarados significativos por el procedimiento indicado. Podemos ver las primeras filas.

```
head(sig)
```

```
[1] 163 262 316 317 336 359
```

Tenemos 553. Guardamos los p-valores y los p-valores ajustados.

```
pvalor = tt[sig,"p.value"]
pajustado = padj[sig]
```

El primer problema es asociar a los genes (en general, la entidad biológica con la que trabajamos) seleccionados uno o varios identificadores. ¿Qué anotación tienen nuestros datos? Necesitamos el paquete [42, annotate].

```
pacman::p_load(annotate)
```

La anotación la obtenemos con `annotate::annotation()`.

```
annotation(eset)
```

```
[1] "hgu133plus2"
```

Cargamos el paquete correspondiente.

```
pacman::p_load(hgu133plus2.db)
```

Obtenemos los identificadores de nuestros genes.

```
ID = featureNames(eset)[sig]
```

Los primeros serían

```
head(ID)
```

```
[1] "1552487_a_at" "1552626_a_at" "1552701_a_at"
[4] "1552703_s_at" "1552730_at" "1552760_at"
```

Podemos determinar los nombres abreviados de nuestros genes (mostramos los dos primeros en todo lo que sigue).

```
lookUp(ID, "hgu133plus2.db", "SYMBOL")[1:2]
```

```
$`1552487_a_at`
[1] "BNC1"

$`1552626_a_at`
[1] "TMEM163"
```

```
getSYMBOL(ID,"hgu133plus2.db")[1:2]
```

```
1552487_a_at 1552626_a_at
     "BNC1" "TMEM163"
```

Sus nombres con

```
lookUp(ID, "hgu133plus2.db", "GENENAME")[1:2]
```

```
$`1552487_a_at`
[1] "basonuclin 1"

$`1552626_a_at`
[1] "transmembrane protein 163"
```

Supongamos que queremos obtener información sobre estos genes en la base de datos Ensembl ¿Cuáles son sus identificadores en esta base de datos?

```
lookUp(ID, "hgu133plus2.db", "ENSEMBL")[1:2]
```

```
$`1552487_a_at`
[1] "ENSG00000169594"

$`1552626_a_at`
[1] "ENSG00000152128"
```

¿O en Gene Ontology?

```
lookUp(ID, "hgu133plus2.db", "GO")
```

O bien sus identificadores Entrez.

```
lookUp(ID, "hgu133plus2.db", "ENTREZID")[1:2]
```

Toda la información que podemos obtener se puede consultar en la ayuda del paquete [20, hgu133plus2.db] o bien con

```
ls("package:hgu133plus2.db")
```

Vamos a generar un **`data.frame`** que vamos a guardar en formato html utilizando distintos paquetes. Obtenemos distintos identificadores con alguna modificación de tipo y valores. En particular es interesante ver cómo generamos la dirección web para los identificadores `ENTREZID`.

```
ID = featureNames(eset)[sig]
Name = as.character(lookUp(ID,"hgu133plus2.db", "GENENAME"))
entrezid = as.character(lookUp(ID, "hgu133plus2.db", "ENTREZID"))
ID[ID == "NA"] = NA
Name[Name == "NA"] = NA
entrezid = ifelse(entrezid == "NA",NA,
  paste0("<a href='http://www.ncbi.nlm.nih.gov/gene/?term=",
         entrezid,"'>",entrezid,"</a>"))
```

La generación de URL a partir de del identificador está implementado en `tami::entrezid2url()`. De un modo análogo lo podemos hacer para otros identificadores. En las funciones `tami::ensembl2url()`, `tami::go2url()` y `tami::kegg2url()` vemos cómo hacerlo para los identificadores Ensembl, Gene Ontology y KEGG.

Posiblemente queramos añadir a estos descriptores los p-valores originales así como los ajustados. Generamos un **`data.frame`** con los distintos identificadores.

```
df = data.frame(ID = ID,Name = Name,entrezid = entrezid,
    pvalor = pvalor,pajustado = pajustado,stringsAsFactors=F)
```

En el **`data.frame`** que acabamos de generar podemos ver que aparecen sondas que no corresponden a ningún gen. Podemos eliminarlas con `stats::na.omit()` y generar el informe sin estas sondas.

```
df = na.omit(df)
```

16.2 Generando un informe en html

16.2.1 ReportingTools

En esta sección vemos cómo generar un informe con [55, ReportingTools].

```
pacman::p_load(ReportingTools)
```

Utilizando `ReportingTools::HTMLReport()` fijamos el nombre del fichero en que guardamos la información así como el directorio en donde queremos guardarlo.

```
foutput = "gse20986_DE"
htmlRep1 = HTMLReport(shortName = foutput,title = foutput,
                      reportDirectory = "./reports/")
```

Guardamos la información y cerramos el fichero con las funciones `ReportingTools::publish()` y `ReportingTools::finish()`.

```
publish(df,htmlRep1)
finish(htmlRep1)
```

```
[1] "./reports//gse20986_DE.html"
```

16.2.2 DT

El paquete [98] permite la generación de tablas en formato html de un modo muy simple y con muchas posibilidades. Si consideramos el **`data.frame`** que hemos generado en el punto anterior

```
ff = DT::datatable(df,escape=FALSE)
```

Podemos guardar el informe generado en el fichero `reports/gse20986_DE_DT.html`.

```
DT::saveWidget(ff, "reports/gse20986_DE_DT.html")
```

16.2.3 Utilizando kableExtra

Vamos a generar el mismo informe utilizando [100].

```
pacman::p_load(kableExtra)
df %>% kable(escape=FALSE) %>%
    kable_styling() %>%
    save_kable("reports/gse20986_DE_kE.html")
```

```
Error in save_kable_latex(x, file, latex_header_includes, keep_tex,
    ↪ density): We hit an error when trying to use magick to read the
    ↪  generated PDF file. You may check your magick installation and
    ↪  try to use magick::image_read to read the PDF file manually.
    ↪ It's also possible that you didn't have ghostscript installed.
```

16.3 Generación de enlaces

En § 16.1 hemos necesitado generar en enlace a bases de datos que nos den información sobre la entidad que analizamos (genes en el ejemplo). Esto es habitual y con frecuencia tendremos que construirnos una función que genere el enlace a otra base de datos. En el paquete [8] se han añadido algunos ejemplos que mostramos aquí.

```
tami::entrezid2url
```

```
function (id)
ifelse(id == "NA", NA, paste("<a href='http://www.ncbi.nlm.nih.gov/gene
    ↪ /?term=",
   id, "'>", id, "</a>", sep = ""))
<bytecode: 0x56441170c6b0>
<environment: namespace:tami>
```

```
tami::ensembl2url
```

```
function (id, site = "http://www.ensembl.org")
paste("<a href='", site, "/id/", id, "'>", id, "</a>", sep = "")
<bytecode: 0x5643b9e86e98>
<environment: namespace:tami>
```

```
tami::go2url
```

```
function (id)
ifelse(id == "NA", NA, paste("<a href='http://amigo.geneontology.org/
    ↪ amigo/term/",
   id, "'>", id, "</a>", sep = ""))
<bytecode: 0x5643b33ec568>
<environment: namespace:tami>
```

```
tami::kegg2url
```

```
function (id)
ifelse(id == "NA", NA, paste("<a href='http://www.genome.jp/dbget-bin/
    ↪ www_bget?",
   id, "'>", id, "</a>", sep = ""))
<bytecode: 0x564454459f08>
<environment: namespace:tami>
```

```
tami::WormBase2url
```

```
function (id)
ifelse(id == "NA", NA, paste("<a href='http://www.wormbase.org/species/
    ↪ c_elegans/gene/",
   id, "'>", id, "</a>", sep = ""))
<bytecode: 0x5644542cb930>
<environment: namespace:tami>
```

16.4 Ejercicios

Ex. 34 — Utilizando el `ExpressionSet` `tamidata::gse1397` se pide:

1. ¿Cuál es el modelo de chip utilizado?
2. Seleccionar el gen en la fila 678. ¿Cuál es su identificador Affymetrix o AffyID? Determinar los identificadores en las siguientes bases de datos: Ensembl, Gene Ontology y Entrez.
3. Elegir al azar 100 genes con la función `stats::sample()`. Determinar sus AffyIDs y los correspondientes en Ensembl, Gene Ontology y Entrez.
   (a) Generar un fichero HTML tal que en las cuatro primeras columnas tengamos los identificadores.
   (b) Repetir el punto anterior pero de modo que, asociado a los identificadores en Ensembl, Gene Ontology y Entrez nos aparezca el enlace para el acceso a la base de datos correspondiente.

Ex. 35 — En el ejemplo descrito en § 16.2.1 incorporar los enlaces a Ensembl.

Parte VII

R/Bioconductor

Capítulo 17

Bioconductor

En este curso tan (o más) importante que el propio R es el proyecto Bioconductor. Básicamente es una colección de paquetes de R para Bioinformática.

Para instalar paquetes de Bioconductor necesitamos el paquete [67, BiocManager]. Se instala con (la versión hay que actualizarla)

```
if (!require("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install(version = "3.16") ##Revisar cuál es la versión
    ↪ actualizada
```

Una vez instalado la parte básica de Bioconductor podemos instalar paquetes adicionales como [73, multtest] con

```
BiocManager::install("multtest")
```

Podemos elegir el repositorio desde el cual instalamos los paquetes de Bioconductor con

```
chooseBioCmirror()
```

También podemos elegir interactivamente el repositorio con

```
setRepositories()
```

Capítulo 18

Anotación

¿De qué variables hablamos? En este manual trabajamos con medidas de abundancia entendida como expresión de un gen o presencia de una mayor cantidad de proteína. En este contexto cuando hablamos de variables utilizaremos con frecuencia el nombre de características. Estas son nuestras variables. Pero, repetimos, ¿de qué variables hablamos? En ocasiones nos estaremos refiriendo a expresión de un gen cuantificado via microarrays o RNA-Seq o alguna otra técnica. En otras ocasiones hablaremos de exones o bien, cuando sea pertinente, de isoformas del gen obtenidas con empalmes alternativos.[95] Por tanto es de gran importancia[96] poder manejar para el organismo que nos ocupe bases de datos que nos permitan conocer cómo denominar a las características: distintos identificadores de genes, exones, isoformas, proteínas. Además cómo pasar de unos identificadores a otros. Cuáles son las correspondencias entre ellos que con frecuencia no son correspondencias 1-1. Para cada organismo podemos encontrar bases de datos disponibles online en la red. Algunas de ellas se ocupan de más de un organismo. Hay bases de datos específicas de cierto tipo de genes (como de microRNAs). Es habitual en la práctica del investigador acudir y usar este tipo de bases de datos con el gran hallazgo informático de copiar y pegar. Es un trabajo tedioso y, posiblemente, innnesario. ¿Todo lo que hacemos en este tema se puede hacer de este modo? Diría que no. En este tema nos ocupamos de cómo acceder a esta información pero desde R/Bioconductor.[97]

[95] Alternative splicing.

[96] Estamos al principio del texto.

[97] El navegador para leer periódicos.

Es muy conveniente consultar http://www.bioconductor.org/help/workflows/annotation/annotation/. Los paquetes de anotación de Bioconductor los tenemos en https://www.bioconductor.org/packages/release/data/annotation/.

Tipos de paquetes de anotación. Tenemos diferentes tipos de paquetes de anotación:

ChipDb Se refieren a una plataforma concreta. Por ejemplo, un chip de microarray. § 18.2

OrgDb Centrados en el organismo.

TxDb Paquetes relativos a transcriptomas de un organismo.

BSgenome Paquetes con genomas.

[98]

[98] En lo que sigue seguimos el flujo de trabajo annotation de Bioconductor.

18.1 AnnotationDbi

Las bases de datos de tipo `ChipDb`, `OrgDb` y `TxDb` heredan todos los métodos de la clase `AnnotationDb` que está definida en [72, AnnotationDbi]. Por ello los métodos aplicables a `AnnotationDb` son aplicables a las demás: `columns`, `keytypes`, `keys` y `select`. Con estos métodos podremos extraer información de las bases de las datos (objetos de clase `ChipDb`, `OrgDb` y `TxDb`) correspondientes. Previo al uso de las distintas bases de datos de anotación es pues conveniente conocer los objetos `AnnotationDb`.

```
pacman::p_load("AnnotationDbi")
```

Para ilustrar los métodos vamos a considerar un paquete de tipo `ChipDb`, en concreto [19, hgu133a.db].

```
pacman::p_load("hgu133a.db")
```

La información contenida la podemos ver con

```
ls("package:hgu133a.db")
```

```
 [1] "hgu133a" "hgu133a_dbconn"
 [3] "hgu133a_dbfile" "hgu133a_dbInfo"
 [5] "hgu133a_dbschema" "hgu133a.db"
 [7] "hgu133aACCNUM" "hgu133aALIAS2PROBE"
 [9] "hgu133aCHR" "hgu133aCHRLENGTHS"
[11] "hgu133aCHRLOC" "hgu133aCHRLOCEND"
[13] "hgu133aENSEMBL" "hgu133aENSEMBL2PROBE"
[15] "hgu133aENTREZID" "hgu133aENZYME"
[17] "hgu133aENZYME2PROBE" "hgu133aGENENAME"
[19] "hgu133aGO" "hgu133aGO2ALLPROBES"
[21] "hgu133aGO2PROBE" "hgu133aMAP"
[23] "hgu133aMAPCOUNTS" "hgu133aOMIM"
[25] "hgu133aORGANISM" "hgu133aORGPKG"
[27] "hgu133aPATH" "hgu133aPATH2PROBE"
[29] "hgu133aPFAM" "hgu133aPMID"
[31] "hgu133aPMID2PROBE" "hgu133aPROSITE"
[33] "hgu133aREFSEQ" "hgu133aSYMBOL"
[35] "hgu133aUNIPROT"
```

Con el nombre del paquete también tenemos información.

```
hgu133a.db
```

```
ChipDb object:
| DBSCHEMAVERSION: 2.1
| Db type: ChipDb
| Supporting package: AnnotationDbi
| DBSCHEMA: HUMANCHIP_DB
| ORGANISM: Homo sapiens
| SPECIES: Human
| MANUFACTURER: Affymetrix
| CHIPNAME: Affymetrix HG-U133A Array
| MANUFACTURERURL: http://www.affymetrix.com
| EGSOURCEDATE: 2021-Apr14
| EGSOURCENAME: Entrez Gene
| EGSOURCEURL: ftp://ftp.ncbi.nlm.nih.gov/gene/DATA
| CENTRALID: ENTREZID
| TAXID: 9606
| GOSOURCENAME: Gene Ontology
| GOSOURCEURL: http://current.geneontology.org/ontology/go-basic.obo
| GOSOURCEDATE: 2021-02-01
| GOEGSOURCEDATE: 2021-Apr14
| GOEGSOURCENAME: Entrez Gene
```

```
| GOEGSOURCEURL: ftp://ftp.ncbi.nlm.nih.gov/gene/DATA
| KEGGSOURCENAME: KEGG GENOME
| KEGGSOURCEURL: ftp://ftp.genome.jp/pub/kegg/genomes
| KEGGSOURCEDATE: 2011-Mar15
| GPSOURCENAME: UCSC Genome Bioinformatics (Homo sapiens)
| GPSOURCEURL:
| GPSOURCEDATE: 2021-Feb16
| ENSOURCEDATE: 2021-Feb16
| ENSOURCENAME: Ensembl
| ENSOURCEURL: ftp://ftp.ensembl.org/pub/current_fasta
| UPSOURCENAME: Uniprot
| UPSOURCEURL: http://www.UniProt.org/
| UPSOURCEDATE: Mon Apr 26 21:53:12 2021
```

En los paquetes de anotación tenemos `columns`. Algunas de estas columnas pueden ser `keys`. Podemos realizar consultas en la base de datos utilizando una `key` y pedir que nos devuelva una o más de una `columns`. ¿Qué información podemos recuperar utilizando `select`?

```
columns(hgu133a.db)
```

```
 [1] "ACCNUM" "ALIAS" "ENSEMBL"
 [4] "ENSEMBLPROT" "ENSEMBLTRANS" "ENTREZID"
 [7] "ENZYME" "EVIDENCE" "EVIDENCEALL"
[10] "GENENAME" "GENETYPE" "GO"
[13] "GOALL" "IPI" "MAP"
[16] "OMIM" "ONTOLOGY" "ONTOLOGYALL"
[19] "PATH" "PFAM" "PMID"
[22] "PROBEID" "PROSITE" "REFSEQ"
[25] "SYMBOL" "UCSCKG" "UNIPROT"
```

Pero: ¿qué información es? Lo obtenemos con[99]

```
help("ENTREZID")
```

[99] Podemos poner cualquiera de los nombres anteriores y nos saldrá la misma ayuda.

No todas las variables que hemos obtenido con `columns` son utilizables para realizar consultas. Aquellas utilizables para las consultas las podemos conocer con `keytypes`. A estas variables las llamamos llaves (`keys`).

```
keytypes(hgu133a.db)
```

```
 [1] "ACCNUM" "ALIAS" "ENSEMBL"
 [4] "ENSEMBLPROT" "ENSEMBLTRANS" "ENTREZID"
 [7] "ENZYME" "EVIDENCE" "EVIDENCEALL"
[10] "GENENAME" "GENETYPE" "GO"
[13] "GOALL" "IPI" "MAP"
[16] "OMIM" "ONTOLOGY" "ONTOLOGYALL"
[19] "PATH" "PFAM" "PMID"
[22] "PROBEID" "PROSITE" "REFSEQ"
[25] "SYMBOL" "UCSCKG" "UNIPROT"
```

¿Cómo conseguir todas los valores de una llave determinada?

```
head(keys(hgu133a.db,keytype="ENTREZID"))
```

```
[1] "10" "100" "1000"
[4] "10000" "100008586" "10001"
```

```
head(keys(hgu133a.db,keytype="ENSEMBL"))
```

```
[1] "ENSG00000121410" "ENSG00000175899"
[3] "ENSG00000291190" "ENSG00000171428"
[5] "ENSG00000156006" "ENSG00000196136"
```

Supongamos que tenemos algunos identificadores Affy (AffyID) y pretendemos conocer sus identificadores `ENTREZID`. Empezamos eligiendo cinco identificadores Affy al azar.[100]

[100] De ahí el uso de la función base::sample.

```
(ids = sample(keys(hgu133a.db,keytype="PROBEID"),5))
```

```
[1] "205812_s_at" "220444_at" "222110_at"
[4] "214052_x_at" "205021_s_at"
```

Vamos a obtener sus identificadores Entrez, los de ENSEMBL y su SYMBOL.

```
df = AnnotationDbi::select(hgu133a.db,keys=ids,
                           columns=c("ENTREZID","ENSEMBL","SYMBOL"),
                           keytype="PROBEID")
```

```
data(gse20986,package="tamidata")
ids = featureNames(gse20986)
library(hgu133plus2.db)
df = AnnotationDbi::select(hgu133plus2.db,keys=ids,
                           columns=c("ENTREZID","ENSEMBL"),
                           keytype="PROBEID")
a = match(featureNames(gse20986),df[,"PROBEID"])
df1 = df[a,]
df1[1,]
```

```
   PROBEID ENTREZID ENSEMBL
1 1007_s_at 780 ENSG00000204580
```

```
dim(df1)
```

```
[1] 54675 3
```

```
dim(gse20986)
```

```
Features Samples
   54675 12
```

```
fData(gse20986) = df1
library("limma")
pData(gse20986)
```

```
                tissue
GSM524662.CEL.gz iris
GSM524663.CEL.gz retina
GSM524664.CEL.gz retina
GSM524665.CEL.gz iris
GSM524666.CEL.gz retina
GSM524667.CEL.gz iris
GSM524668.CEL.gz choroides
GSM524669.CEL.gz choroides
GSM524670.CEL.gz choroides
GSM524671.CEL.gz huvec
GSM524672.CEL.gz huvec
GSM524673.CEL.gz huvec
```

```
design = model.matrix(~ 0 + pData(gse20986)[,"tissue"])
colnames(design) = levels(pData(gse20986)[,"tissue"])
design
```

```
  iris retina choroides huvec
1 1 0 0 0
2 0 1 0 0
3 0 1 0 0
4 1 0 0 0
5 0 1 0 0
6 1 0 0 0
7 0 0 1 0
8 0 0 1 0
9 0 0 1 0
10 0 0 0 1
11 0 0 0 1
12 0 0 0 1
attr(,"assign")
[1] 1 1 1 1
attr(,"contrasts")
attr(,"contrasts")$`pData(gse20986)[, "tissue"]`
[1] "contr.treatment"
```

```
(contrast.matrix = makeContrasts(dif12 = iris − retina,
                                 dif13 = iris − choroides,
                                 dif14 = iris − huvec
                                ,levels = design))
```

```
         Contrasts
Levels dif12 dif13 dif14
  iris 1 1 1
  retina -1 0 0
  choroides 0 -1 0
  huvec 0 0 -1
```

```
fit = lmFit(gse20986,design)
fit2 = contrasts.fit(fit,contrast.matrix)
fit3 = eBayes(fit2)
topTable(fit3,coef=1,number=2)
```

```
                                PROBEID
AFFX-HUMRGE/M10098_3_at AFFX-HUMRGE/M10098_3_at
219273_at 219273_at
                    ENTREZID ENSEMBL
AFFX-HUMRGE/M10098_3_at <NA> <NA>
219273_at 8812 ENSG00000090061
                      logFC AveExpr
AFFX-HUMRGE/M10098_3_at 4.716983 10.696277
219273_at 2.696519 4.241303
                         t P.Value
AFFX-HUMRGE/M10098_3_at 19.69965 2.588982e-08
219273_at 17.96796 5.507795e-08
                       adj.P.Val B
AFFX-HUMRGE/M10098_3_at 0.001047691 9.974955
219273_at 0.001047691 9.202545
```

18.2 ChipDb

Si trabajamos con microarrays (§ 2) nuestras características serán las sondas.[101] Estas sondas tendrán un identificador que el fabricante del chip le ha asignado. Esto no es informativo para nosotros. Hemos de poder hacer corresponder este identificador con el gen al que corresponde. Estas bases de datos son de tipo `ChipDb`. Uno de los chips más populares de Affymetrix, Affymetrix Human Genome U133. El paquete Bioconductor con la anotación de este chip es [19, hgu133a.db]. Lo cargamos.

[101] Asociadas a genes.

```
pacman::p_load(hgu133a.db)
```

[102] Probes.

¿Qué sondas[102] tienen correspondencia en Entrez?

```
mappedProbes = mappedkeys(hgu133aENTREZID)
```

Lo guardamos en forma de lista.

```
mappedProbesList = as.list(hgu133aENTREZID[mappedProbes])
```

Por ejemplo, la primera posición de la lista nos da el identificador de Affymetrix[103] y su identificador Entrez[104].

[103] AffyID.

[104] ENTREZID.

```
mappedProbesList[1]
```

```
$`1007_s_at`
[1] "780"
```

O el correspondiente a la posición 4567.

```
mappedProbesList[4567]
```

```
$`205155_s_at`
[1] "6712"
```

Si hacemos

```
ls("package:hgu133a.db")
```

```
 [1] "hgu133a" "hgu133a_dbconn"
 [3] "hgu133a_dbfile" "hgu133a_dbInfo"
 [5] "hgu133a_dbschema" "hgu133a.db"
 [7] "hgu133aACCNUM" "hgu133aALIAS2PROBE"
 [9] "hgu133aCHR" "hgu133aCHRLENGTHS"
[11] "hgu133aCHRLOC" "hgu133aCHRLOCEND"
[13] "hgu133aENSEMBL" "hgu133aENSEMBL2PROBE"
[15] "hgu133aENTREZID" "hgu133aENZYME"
[17] "hgu133aENZYME2PROBE" "hgu133aGENENAME"
[19] "hgu133aGO" "hgu133aGO2ALLPROBES"
[21] "hgu133aGO2PROBE" "hgu133aMAP"
[23] "hgu133aMAPCOUNTS" "hgu133aOMIM"
[25] "hgu133aORGANISM" "hgu133aORGPKG"
[27] "hgu133aPATH" "hgu133aPATH2PROBE"
[29] "hgu133aPFAM" "hgu133aPMID"
[31] "hgu133aPMID2PROBE" "hgu133aPROSITE"
[33] "hgu133aREFSEQ" "hgu133aSYMBOL"
[35] "hgu133aUNIPROT"
```

podemos ver todas las correspondencias que nos ofrece el paquete. Consideremos unos identificadores Affymetrix.

```
ids = c("39730_at", "1635_at", "1674_at", "40504_at", "40202_at")
```

Se supone que el chip es `hgu95av2`. Cargamos el paquete de anotación correspondiente al chip utilizado [21, hgu95av2.db].

```
pacman::p_load("hgu95av2.db")
```

Y lo mismo que antes. Ahora tenemos además la correspondencia entre las sondas y los genes.

```
columns(hgu95av2.db)
```

```
 [1] "ACCNUM" "ALIAS" "ENSEMBL"
 [4] "ENSEMBLPROT" "ENSEMBLTRANS" "ENTREZID"
 [7] "ENZYME" "EVIDENCE" "EVIDENCEALL"
[10] "GENENAME" "GENETYPE" "GO"
[13] "GOALL" "IPI" "MAP"
[16] "OMIM" "ONTOLOGY" "ONTOLOGYALL"
[19] "PATH" "PFAM" "PMID"
[22] "PROBEID" "PROSITE" "REFSEQ"
[25] "SYMBOL" "UCSCKG" "UNIPROT"
```

```
keytypes(hgu95av2.db)
```

```
 [1] "ACCNUM" "ALIAS" "ENSEMBL"
 [4] "ENSEMBLPROT" "ENSEMBLTRANS" "ENTREZID"
 [7] "ENZYME" "EVIDENCE" "EVIDENCEALL"
[10] "GENENAME" "GENETYPE" "GO"
[13] "GOALL" "IPI" "MAP"
[16] "OMIM" "ONTOLOGY" "ONTOLOGYALL"
[19] "PATH" "PFAM" "PMID"
[22] "PROBEID" "PROSITE" "REFSEQ"
[25] "SYMBOL" "UCSCKG" "UNIPROT"
```

Los identificadores que teníamos eran los AffyID, esto es, los identificadores de las sondas utilizadas o PROBEID. Podemos plantearnos su correspondencia con el símbolo o nombre del gen y las proteínas asociadas en la base de datos PFAM.

```
columns = c("PFAM","SYMBOL")
AnnotationDbi::select(hgu95av2.db, keys=ids, columns, keytype="PROBEID")
```

18.3 OrgDb

Este tipo de paquete se refiere a un organismo dado y está centrado en el gen. Veamos como ejemplo el relativo a ser humano.

```
pacman::p_load(org.Hs.eg.db)
```

¿Qué tipo de cosas o qué claves podemos manejar?

```
columns(org.Hs.eg.db)
```

```
 [1] "ACCNUM" "ALIAS" "ENSEMBL"
 [4] "ENSEMBLPROT" "ENSEMBLTRANS" "ENTREZID"
 [7] "ENZYME" "EVIDENCE" "EVIDENCEALL"
[10] "GENENAME" "GENETYPE" "GO"
[13] "GOALL" "IPI" "MAP"
[16] "OMIM" "ONTOLOGY" "ONTOLOGYALL"
[19] "PATH" "PFAM" "PMID"
[22] "PROSITE" "REFSEQ" "SYMBOL"
[25] "UCSCKG" "UNIPROT"
```

O bien con

```
keytypes(org.Hs.eg.db)
```

```
 [1] "ACCNUM" "ALIAS" "ENSEMBL"
 [4] "ENSEMBLPROT" "ENSEMBLTRANS" "ENTREZID"
 [7] "ENZYME" "EVIDENCE" "EVIDENCEALL"
[10] "GENENAME" "GENETYPE" "GO"
[13] "GOALL" "IPI" "MAP"
[16] "OMIM" "ONTOLOGY" "ONTOLOGYALL"
[19] "PATH" "PFAM" "PMID"
[22] "PROSITE" "REFSEQ" "SYMBOL"
[25] "UCSCKG" "UNIPROT"
```

Figura 18.1: Un anuncio.

Por ejemplo, si queremos los primeros identificadores de genes utilizando los identificadores Entrez o ENTREZID tenemos

```
head(keys(org.Hs.eg.db, keytype="ENTREZID"))
```

```
[1] "1" "2" "3" "9" "10" "11"
```

Si consideramos sus identificadores en la base de datos Ensembl entonces podemos usar

```
head(keys(org.Hs.eg.db, keytype="ENSEMBL"))
```

```
[1] "ENSG00000121410" "ENSG00000175899"
[3] "ENSG00000291190" "ENSG00000171428"
[5] "ENSG00000156006" "ENSG00000196136"
```

Y en Gene Ontology.

```
head(keys(org.Hs.eg.db, keytype="GO"))
```

```
[1] "GO:0003674" "GO:0005576" "GO:0005615"
[4] "GO:0005886" "GO:0008150" "GO:0031093"
```

Supongamos que elegimos un sistema de identificación, por ejemplo, ENTREZID. En concreto, los cinco primeros genes.

```
(ids = keys(org.Hs.eg.db, keytype="ENTREZID")[1:5])
```

```
[1] "1" "2" "3" "9" "10"
```

A partir de estos identicadores podemos obtener el resto.

```
AnnotationDbi::select(org.Hs.eg.db,keys=ids,column="SYMBOL",
                      keytype='ENTREZID')
```

```
  ENTREZID SYMBOL
1 1 A1BG
2 2 A2M
3 3 A2MP1
4 9 NAT1
5 10 NAT2
```

Supongamos que nos fijamos en el gen con código Ensembl `ENSG00000000003` ↪ .

```
(id = "ENSG00000171428")
```

```
[1] "ENSG00000171428"
```

Buscamos su correspondencia en Gene Ontology.

```
(res = AnnotationDbi::select(org.Hs.eg.db, keys=id, column="GO",
                             keytype="ENSEMBL"))
```

```
         ENSEMBL GO EVIDENCE ONTOLOGY
1 ENSG00000171428 GO:0004060 TAS MF
2 ENSG00000171428 GO:0005829 TAS CC
3 ENSG00000171428 GO:0006805 TAS BP
```

Como vemos no tenemos una correspondencia 1-1. Al mismo gen le corresponden distintos términos Gene Ontology. Si solamente tenemos interés en ellos podemos hacer

```
res[,"GO"]
```

```
[1] "GO:0004060" "GO:0005829" "GO:0006805"
```

Puesto que tenemos identificadores Gene Ontology podemos utilizar el paquete [18, GO.db] para obtener los términos Gene Ontology correspondientes.

```
pacman::p_load(GO.db)
```

Y los términos Gene Ontology serían

```
AnnotationDbi::select(GO.db, keys=res[,"GO"], columns="TERM",
                      keytype="GOID")
```

18.4 TxDb

Los paquetes `TxDb` están centrados en el genoma. Vamos a trabajar con la drosophila melanogaster. https://en.wikipedia.org/wiki/Drosophila_melanogaster. Cargamos la base de datos.

```
library("GenomicFeatures")
library("TxDb.Dmelanogaster.UCSC.dm3.ensGene")
```

Al cargar este paquete lo que hemos hecho es leer un objeto que se llama como el propio paquete y de clase `TxDb`.[105]

[105] Todos los paquetes de Bioconductor que empiezan con TxDb son de este tipo.

```
class(TxDb.Dmelanogaster.UCSC.dm3.ensGene)
```

```
[1] "TxDb"
attr(,"package")
[1] "GenomicFeatures"
```

Hacemos una copia con un nombre más breve.

```
txdb = TxDb.Dmelanogaster.UCSC.dm3.ensGene
```

¿De qué clase es este objeto?

```
class(txdb)
```

```
[1] "TxDb"
attr(,"package")
[1] "GenomicFeatures"
```

Es pues un objeto de clase `TxDb`. Tenemos un resumen sobre los datos contenidos en `txdb` con

```
txdb
```

```
TxDb object:
# Db type: TxDb
# Supporting package: GenomicFeatures
# Data source: UCSC
# Genome: dm3
# Organism: Drosophila melanogaster
# Taxonomy ID: 7227
# UCSC Table: ensGene
# Resource URL: http://genome.ucsc.edu/
# Type of Gene ID: Ensembl gene ID
# Full dataset: yes
# miRBase build ID: NA
# transcript_nrow: 29173
# exon_nrow: 76920
# cds_nrow: 62135
```

```
# Db created by: GenomicFeatures package from Bioconductor
# Creation time: 2015-10-07 18:15:53 +0000 (Wed, 07 Oct 2015)
# GenomicFeatures version at creation time: 1.21.30
# RSQLite version at creation time: 1.0.0
# DBSCHEMAVERSION: 1.1
```

Podemos ver la información de la que disponemos con

```
columns(txdb)
```

```
 [1] "CDSCHROM" "CDSEND" "CDSID"
 [4] "CDSNAME" "CDSSTART" "CDSSTRAND"
 [7] "EXONCHROM" "EXONEND" "EXONID"
[10] "EXONNAME" "EXONRANK" "EXONSTART"
[13] "EXONSTRAND" "GENEID" "TXCHROM"
[16] "TXEND" "TXID" "TXNAME"
[19] "TXSTART" "TXSTRAND" "TXTYPE"
```

y con

```
keytypes(txdb)
```

```
[1] "CDSID" "CDSNAME" "EXONID" "EXONNAME"
[5] "GENEID" "TXID" "TXNAME"
```

Para el manejo de este tipo de bases de datos es útil [16, GenomicFeatures].

```
pacman::p_load(GenomicFeatures)
```

Por ejemplo: ¿Qué cromosomas tenemos?[106]

```
seqlevels(txdb)
```

```
[1] "chr2L"
```

Cuando se carga la base de datos todos los cromosomas están activos y lo que hagamos nos dará información sobre todos ellos. Supongamos que no queremos esto. Queremos, por ejemplo, trabajar solamente con `chr2L`. Lo conseguimos con

```
seqlevels(txdb) = "chr2L"
```

Supongamos que queremos conocer los `GENEID` de los primeros genes.

```
(keysGENEID = head(keys(txdb, keytype="GENEID"),n=3))
```

```
[1] "FBgn0000003" "FBgn0000008" "FBgn0000014"
```

```
columns = c("TXNAME", "TXSTART","TXEND","TXSTRAND")
AnnotationDbi::select(txdb, keysGENEID, columns, keytype="GENEID")
```

```
      GENEID TXNAME TXSTRAND TXSTART
1 FBgn0000003 FBtr0081624 + 2648220
2 FBgn0000008 FBtr0100521 + 18024494
3 FBgn0000008 FBtr0071763 + 18024496
4 FBgn0000008 FBtr0071764 + 18024938
5 FBgn0000014 FBtr0306337 - 12632936
6 FBgn0000014 FBtr0083388 - 12633349
7 FBgn0000014 FBtr0083387 - 12633349
8 FBgn0000014 FBtr0300485 - 12633349
    TXEND
1 2648518
2 18060339
3 18060346
```

[106] En http://ucscbrowser.genenetwork.org/cgi-bin/hgGateway?hgsid=732&clade=insect&org=0&db=0 podemos encontrar una explicación detallada.

```
4 18060346
5 12655767
6 12653845
7 12655300
8 12655474
```

Nos devuelve el identificador del gen (`GENEID`), el nombre del transcrito, la hebra o cadena y el punto de inicio y de finalización. Notemos que el primer gen solo tiene un transcrito mientras que el segundo gen tiene tres transcritos.

Los objetos `TxDb` nos permiten obtener las anotaciones como `GRanges` ↪ . Empezamos con los transcritos.

```
(txdb.tr = transcripts(txdb))
```

```
GRanges object with 5384 ranges and 2 metadata columns:
       seqnames ranges strand |
          <Rle> <IRanges> <Rle> |
    [1] chr2L 7529-9484 + |
    [2] chr2L 7529-9484 + |
    [3] chr2L 7529-9484 + |
    [4] chr2L 21952-24237 + |
    [5] chr2L 66584-71390 + |
    ... ... ... ... .
 [5380] chr2L 22892306-22918560 - |
 [5381] chr2L 22892306-22918647 - |
 [5382] chr2L 22959606-22960915 - |
 [5383] chr2L 22959606-22961179 - |
 [5384] chr2L 22959606-22961179 - |
          tx_id tx_name
       <integer> <character>
    [1] 1 FBtr0300689
    [2] 2 FBtr0300690
    [3] 3 FBtr0330654
    [4] 4 FBtr0309810
    [5] 5 FBtr0306539
    ... ... ...
 [5380] 5380 FBtr0331166
 [5381] 5381 FBtr0111127
 [5382] 5382 FBtr0111241
 [5383] 5383 FBtr0111239
 [5384] 5384 FBtr0111240
 -------
 seqinfo: 1 sequence from dm3 genome
```

Como estamos trabajando con el cromosoma `chr2L` nos devuelve la información en este nuevo (y mejor) formato. Los que ocupan las posiciones 1, 2, 3 y 1000 serían

```
txdb.tr[c(1:3,1000)]
```

```
GRanges object with 4 ranges and 2 metadata columns:
     seqnames ranges strand | tx_id
        <Rle> <IRanges> <Rle> | <integer>
 [1] chr2L 7529-9484 + | 1
 [2] chr2L 7529-9484 + | 2
 [3] chr2L 7529-9484 + | 3
 [4] chr2L 8011405-8026898 + | 1000
        tx_name
     <character>
 [1] FBtr0300689
 [2] FBtr0300690
 [3] FBtr0330654
 [4] FBtr0079533
 -------
 seqinfo: 1 sequence from dm3 genome
```

También podemos obtener información sobre los exones en modo de un objeto `GRanges`.

```
(txdb.ex = exons(txdb))
```

```
GRanges object with 13850 ranges and 1 metadata column:
        seqnames ranges strand |
           <Rle> <IRanges> <Rle> |
    [1] chr2L 7529-8116 + |
    [2] chr2L 8193-8589 + |
    [3] chr2L 8193-9484 + |
    [4] chr2L 8229-9484 + |
    [5] chr2L 8668-9484 + |
    ... ... ... ... .
[13846] chr2L 22959606-22959815 - |
[13847] chr2L 22959877-22960833 - |
[13848] chr2L 22959877-22960876 - |
[13849] chr2L 22959877-22960915 - |
[13850] chr2L 22960932-22961179 - |
         exon_id
        <integer>
    [1] 1
    [2] 2
    [3] 3
    [4] 4
    [5] 5
    ... ...
[13846] 13846
[13847] 13847
[13848] 13848
[13849] 13849
[13850] 13850
-------
seqinfo: 1 sequence from dm3 genome
```

Como antes el exon que ocupa la posición 123 sería

```
txdb.ex[123]
```

```
GRanges object with 1 range and 1 metadata column:
    seqnames ranges strand | exon_id
       <Rle> <IRanges> <Rle> | <integer>
[1] chr2L 277930-278323 + | 123
-------
seqinfo: 1 sequence from dm3 genome
```

Podemos incluir metadatos adicionales como puede ser el identificador del gen.

```
transcripts(txdb, columns = c("tx_id","tx_name","gene_id"))
```

```
GRanges object with 5384 ranges and 3 metadata columns:
       seqnames ranges strand |
          <Rle> <IRanges> <Rle> |
    [1] chr2L 7529-9484 + |
    [2] chr2L 7529-9484 + |
    [3] chr2L 7529-9484 + |
    [4] chr2L 21952-24237 + |
    [5] chr2L 66584-71390 + |
    ... ... ... ... .
[5380] chr2L 22892306-22918560 - |
[5381] chr2L 22892306-22918647 - |
[5382] chr2L 22959606-22960915 - |
[5383] chr2L 22959606-22961179 - |
[5384] chr2L 22959606-22961179 - |
         tx_id tx_name gene_id
       <integer> <character> <CharacterList>
```

```
   [1] 1 FBtr0300689 FBgn0031208
   [2] 2 FBtr0300690 FBgn0031208
   [3] 3 FBtr0330654 FBgn0031208
   [4] 4 FBtr0309810 FBgn0263584
   [5] 5 FBtr0306539 FBgn0067779
   ... ... ... ...
[5380] 5380 FBtr0331166 FBgn0250907
[5381] 5381 FBtr0111127 FBgn0250907
[5382] 5382 FBtr0111241 FBgn0086683
[5383] 5383 FBtr0111239 FBgn0086683
[5384] 5384 FBtr0111240 FBgn0086683
-------
seqinfo: 1 sequence from dm3 genome
```

Obtenemos las regiones CDS con

```
(txdb.cds = cds(txdb))
```

```
GRanges object with 11003 ranges and 1 metadata column:
        seqnames ranges strand |
           <Rle> <IRanges> <Rle> |
    [1] chr2L 7680-8116 + |
    [2] chr2L 8193-8589 + |
    [3] chr2L 8193-8610 + |
    [4] chr2L 8229-8610 + |
    [5] chr2L 8668-9276 + |
    ... ... ... ... .
[10999] chr2L 22959877-22960833 - |
[11000] chr2L 22959877-22960873 - |
[11001] chr2L 22959877-22960876 - |
[11002] chr2L 22960932-22960995 - |
[11003] chr2L 22960932-22961048 - |
          cds_id
       <integer>
    [1] 1
    [2] 2
    [3] 3
    [4] 4
    [5] 5
    ... ...
[10999] 10999
[11000] 11000
[11001] 11001
[11002] 11002
[11003] 11003
-------
seqinfo: 1 sequence from dm3 genome
```

En los tres casos hemos obtenido un objeto `GRanges`. A este tipo de objetos podemos aplicar otros métodos que nos dan información adicional. Por ejemplo, el número de elementos que lo componen

```
length(txdb.cds)
```

```
[1] 11003
```

Podemos ver la hebra en la que están.

```
strand(txdb.cds)
```

A partir de un objeto de clase `TxDb` podemos obtener un `GRangesList` ↪ en la cual separamos por alguna característica. Por ejemplo, los objetos `GRanges` para los distintos genes.

```
transcriptsBy(txdb, by="gene")
```

Podemos agrupar los exones por gen.

```
exonsBy(txdb, by="gene")
```

O bien podemos tener los CDS agrupados por transcrito.

```
cdsBy(txdb, by="tx")
```

O los intrones agrupados por transcrito.

```
intronsByTranscript(txdb)
```

También podemos tener las regiones UTR 5' y 3' agrupadas por transcrito.

```
fiveUTRsByTranscript(txdb)
```

```
threeUTRsByTranscript(txdb)
```

18.5 BSgenome

Estos paquetes contienen datos de secuencias para organismos secuenciados.

BSgenome.Dmelanogaster.UCSC.dm3

```
pacman::p_load(BSgenome.Dmelanogaster.UCSC.dm3,GenomicRanges)
tx2seqs = extractTranscriptSeqs(BSgenome.Dmelanogaster.UCSC.dm3,
    ↪ TxDb.Dmelanogaster.UCSC.dm3.ensGene)
```

La secuencia correspondiente al primer gen sería

```
tx2seqs[[1]]
```

Si queremos conocer la secuencia entre las posiciones 1000 y 1020 entonces

```
tx2seqs[[1]][1000:1020]
```

También podemos traducir estas secuencias a proteínas con

```
suppressWarnings(translate(tx2seqs[[1]]))
```

Para todos los transcritos lo hacemos con

```
suppressWarnings(translate(tx2seqs))
```

No todo lo que se transcribe se traduce. Para obtener obtener las que realmente se traducen se puede hacer

```
cds2seqs = extractTranscriptSeqs(BSgenome.Dmelanogaster.UCSC.dm3,
                                 cdsBy(txdb, by="tx"))
translate(cds2seqs)
```

BSgenome.Hsapiens.UCSC.hg19

Estos paquetes contienen datos de secuencias para organismos secuenciados.

```
pacman::p_load(BSgenome.Hsapiens.UCSC.hg19)
```

```
Hsapiens
```

```
| BSgenome object for Human
| - organism: Homo sapiens
| - provider: UCSC
| - genome: hg19
| - release date: June 2013
| - 298 sequence(s):
| chr1 chr2
| chr3 chr4
| chr5 chr6
| chr7 chr8
| chr9 chr10
| ... ...
| chr19_gl949752_alt chr19_gl949753_alt
| chr20_gl383577_alt chr21_gl383578_alt
| chr21_gl383579_alt chr21_gl383580_alt
| chr21_gl383581_alt chr22_gl383582_alt
| chr22_gl383583_alt chr22_kb663609_alt
|
| Tips: call 'seqnames()' on the object to get all
| the sequence names, call 'seqinfo()' to get the
| full sequence info, use the '$' or '[[' operator
| to access a given sequence, see '?BSgenome' for
| more information.
```

Nos fijamos en las secuencias.

```
seqNms = seqnames(Hsapiens)
head(seqNms)
```

```
[1] "chr1" "chr2" "chr3" "chr4" "chr5" "chr6"
```

Nos fijamos en las dos primeras.

```
getSeq(Hsapiens, seqNms[1:2])
```

Podemos, utilizando un objeto GRanges, obtener la secuencia de bases en los cromosomas que queramos, en la hebra que queramos, entre las posiciones que queramos.

```
rngs ←GRanges(seqnames = c('chr1', 'chr4'), strand=c('+','−'),
                    ranges = IRanges(start=c(100000,300000),
                        end=c(100023,300037)))
rngs
res ←getSeq(Hsapiens, rngs)
res
```

18.6 OrganismDb

Un paquete de tipo OrganismDb nos permite combinar información (para el organismo con que estemos trabajando) de GO.db con el correspondiente TxDb y OrgDb.[107]

[107] Buscar OrganismDb en Bioconductor.

```
pacman::p_load(Homo.sapiens)
```

Tomamos las dos primeras.

```
keys = head(keys(Homo.sapiens, keytype="ENTREZID"), n=2)
columns = c("SYMBOL","TXNAME")
AnnotationDbi::select(Homo.sapiens, keys, columns, keytype="ENTREZID")
```

```
  ENTREZID SYMBOL TXNAME
1 1 A1BG uc002qsd.4
2 1 A1BG uc002qsf.2
3 2 A2M uc001qvk.1
4 2 A2M uc009zgk.1
```

También podemos obtener `GRanges`.

```
transcripts(Homo.sapiens, columns=c("TXNAME","SYMBOL"))
```

```
GRanges object with 82960 ranges and 2 metadata columns:
             seqnames ranges strand |
                <Rle> <IRanges> <Rle> |
    [1] chr1 11874-14409 + |
    [2] chr1 11874-14409 + |
    [3] chr1 11874-14409 + |
    [4] chr1 69091-70008 + |
    [5] chr1 321084-321115 + |
    ... ... ... ... .
 [82956] chrUn_gl000237 1-2686 - |
 [82957] chrUn_gl000241 20433-36875 - |
 [82958] chrUn_gl000243 11501-11530 + |
 [82959] chrUn_gl000243 13608-13637 + |
 [82960] chrUn_gl000247 5787-5816 - |
             TXNAME SYMBOL
       <CharacterList> <CharacterList>
    [1] uc001aaa.3 DDX11L1
    [2] uc010nxq.1 DDX11L1
    [3] uc010nxr.1 DDX11L1
    [4] uc001aal.1 OR4F5
    [5] uc001aaq.2 <NA>
    ... ... ...
 [82956] uc011mgu.1 <NA>
 [82957] uc011mgv.2 <NA>
 [82958] uc011mgw.1 <NA>
 [82959] uc022brq.1 <NA>
 [82960] uc022brr.1 <NA>
 -------
 seqinfo: 93 sequences (1 circular) from hg19 genome
```

18.7 biomaRt

El paquete [32, biomaRt] es un interfaz para poder acceder a una serie de bases de datos que implementan BioMart. [108]

[108] http://www.biomart.org.

```
pacman::p_load(biomaRt)
```

Podemos ver la lista de `mart`'s que tenemos (mostramos los primeros).

```
listMarts(host="https://www.ensembl.org")
```

```
Error in `collect()`:
! Failed to collect lazy table.
Caused by error in `db_collect()`:
! Arguments in `...` must be used.
x Problematic argument:
* ..1 = Inf
i Did you misspell an argument name?
```

Elegimos utilizar `ensembl`.

```
(ensembl = useMart("ENSEMBL_MART_ENSEMBL",host="https://
    ↪ www.ensembl.org"))
```

```
Object of class 'Mart':
  Using the ENSEMBL_MART_ENSEMBL BioMart database
  No dataset selected.
```

Ahora hemos de elegir el conjunto de datos a utilizar. Podemos ver los disponibles con

```
head(listDatasets(ensembl))
```

```
                    dataset
1 abrachyrhynchus_gene_ensembl
2 acalliptera_gene_ensembl
3 acarolinensis_gene_ensembl
4 acchrysaetos_gene_ensembl
5 acitrinellus_gene_ensembl
6 amelanoleuca_gene_ensembl
                         description
1 Pink-footed goose genes (ASM259213v1)
2 Eastern happy genes (fAstCal1.2)
3 Green anole genes (AnoCar2.0v2)
4 Golden eagle genes (bAquChr1.2)
5 Midas cichlid genes (Midas_v5)
6 Giant panda genes (ASM200744v2)
     version
1 ASM259213v1
2 fAstCal1.2
3 AnoCar2.0v2
4 bAquChr1.2
5 Midas_v5
6 ASM200744v2
```

Elegimos la correspondiente a ser humano.

```
(ensembl = useMart("ENSEMBL_MART_ENSEMBL",dataset="
    ↪ hsapiens_gene_ensembl",
    host="https://www.ensembl.org"))
```

Podemos ver los atributos con

```
head(listAttributes(ensembl))
```

De hecho, son

```
nrow(listAttributes(ensembl))
```

Podemos comprobar que hay muchos que identifican el gen con las sondas de Affymetrix, en concreto, con los AffyID. Supongamos que nos fijamos en los siguientes AffyID.

```
affyids=c("202763_at","209310_s_at","207500_at")
```

¿A qué genes corresponden?

```
getBM(attributes=c('affy_hg_u133_plus_2', 'entrezgene'),
                    filters = 'affy_hg_u133_plus_2',
                    values = affyids, mart = ensembl)
```

Podemos obtener todos los identificadores.

```
head(getBM(attributes='affy_hg_u133_plus_2', mart = ensembl))
```

18.8 KEGGREST

Con [84, KEGGREST] podemos acceder a la base de datos KEGG. En concreto permite el acceso a KEGG REST API.

18.9 Tareas habituales con anotaciones

¿Cuáles son las tareas que tenemos que realizar habitualmente? En esta sección vemos posibles soluciones que las resuelvan.

18.9.1 Cambiar identificadores de un ExpressionSet

[109] ??.

Consideremos el ExpressionSet tamidata::gse1397.[109] Las características están codificadas utilizando los identificadores Affymetrix. Pretendemos incorporar la información que nos permita conocer la correspondencia entre estos identificadores (PROBEID) y los identificadores Entrez y Ensembl. Esta información la incorporamos como `fData` del ExpressionSet.

Necesitamos los paquetes.

```
pacman::p_load("Biobase","AnnotationDbi","BiocGenerics")
```

Leemos los datos.

```
data(gse1397,package="tamidata")
```

¿Qué paquete de anotación necesitamos?

```
Biobase::annotation(gse1397)
```

```
[1] "hgu133a"
```

Lo cargamos. En https://www.bioconductor.org/packages/3.3/data/annotation/ tenemos el listado de los paquetes de anotación de los que dispone Bioconductor.

```
pacman::p_load("hgu133a.db")
```

Como hemos visto en § 18.1 con `AnnotationDbi::columns` podemos ver la información que tenemos y con AnnotationDbi::keytypes las llaves con las que podemos realizar consultas.

```
columns(hgu133a.db)
```

```
 [1] "ACCNUM" "ALIAS" "ENSEMBL"
 [4] "ENSEMBLPROT" "ENSEMBLTRANS" "ENTREZID"
 [7] "ENZYME" "EVIDENCE" "EVIDENCEALL"
[10] "GENENAME" "GENETYPE" "GO"
[13] "GOALL" "IPI" "MAP"
[16] "OMIM" "ONTOLOGY" "ONTOLOGYALL"
[19] "PATH" "PFAM" "PMID"
[22] "PROBEID" "PROSITE" "REFSEQ"
[25] "SYMBOL" "UCSCKG" "UNIPROT"
```

```
keytypes(hgu133a.db)
```

```
 [1] "ACCNUM" "ALIAS" "ENSEMBL"
 [4] "ENSEMBLPROT" "ENSEMBLTRANS" "ENTREZID"
 [7] "ENZYME" "EVIDENCE" "EVIDENCEALL"
[10] "GENENAME" "GENETYPE" "GO"
[13] "GOALL" "IPI" "MAP"
[16] "OMIM" "ONTOLOGY" "ONTOLOGYALL"
[19] "PATH" "PFAM" "PMID"
[22] "PROBEID" "PROSITE" "REFSEQ"
[25] "SYMBOL" "UCSCKG" "UNIPROT"
```

Vemos que, en este caso coinciden `columns` y `keys`. Los valores los tenemos con

```
head(keys(hgu133a.db,keytype="PROBEID"))
```

```
[1] "1007_s_at" "1053_at" "117_at"
[4] "121_at" "1255_g_at" "1294_at"
```

que corresponde con los identificadores Affy o identificadores de las sondas. Podemos ver los primeros que tenemos en nuestros datos.

```
head(featureNames(gse1397))
```

```
[1] "1007_s_at" "1053_at" "117_at"
[4] "121_at" "1255_g_at" "1294_at"
```

¿Coinciden todos?

```
table(featureNames(gse1397) == keys(hgu133a.db,keytype="PROBEID"))
```

```
FALSE TRUE
   53 22230
```

Vemos que no todos coinciden. ¿Quienes son?

```
(control = which(featureNames(gse1397) != keys(hgu133a.db,keytype="
    ↪ PROBEID")))
```

```
 [1] 22231 22232 22233 22234 22235 22236 22237
 [8] 22238 22239 22240 22241 22242 22243 22244
[15] 22245 22246 22247 22248 22249 22250 22251
[22] 22252 22253 22254 22255 22256 22257 22258
[29] 22259 22260 22261 22262 22263 22264 22265
[36] 22266 22267 22268 22269 22270 22271 22272
[43] 22273 22274 22275 22276 22277 22278 22279
[50] 22280 22281 22282 22283
```

¿Que identificadores tienen estas sondas?

```
head(featureNames(gse1397)[control])
```

```
[1] "AFFX-hum_alu_at"
[2] "AFFX-HUMGAPDH/M33197_3_at"
[3] "AFFX-HUMGAPDH/M33197_5_at"
[4] "AFFX-HUMGAPDH/M33197_M_at"
[5] "AFFX-HUMISGF3A/M97935_3_at"
[6] "AFFX-HUMISGF3A/M97935_5_at"
```

Son sondas de control que no corresponden a ningún gen. Hemos dicho antes que queremos modificar los identificadores utilizados en el ExpressionSet. Y queremos hacerlo pasando a Entrez y Ensembl. Buscamos las correspondencias.

```
probeid2entrez =
    AnnotationDbi::select(hgu133a.db,keys=featureNames(gse1397),
                          columns="ENTREZID",keytype="PROBEID")
```

Nos ha devuelvo un **`data.frame`**.

```
class(probeid2entrez)
```

```
[1] "data.frame"
```

Es importante notar que no hay una correspondencia 1-1 entre los distintos identificadores.

```
head(probeid2entrez)
```

```
   PROBEID ENTREZID
1 1007_s_at 780
2 1053_at 5982
3 117_at 3310
4 121_at 7849
5 1255_g_at 2978
6 1294_at 7318
```

La primera sonda tiene dos identificadores Entrez distintos. Una opción para resolver esta multiplicidad es utilizar BiocGenerics::match.

```
indices = match(featureNames(gse1397),probeid2entrez$PROBEID)
```

¿Cómo se ha resuelto?

```
head(featureNames(gse1397))
```

```
[1] "1007_s_at" "1053_at" "117_at"
[4] "121_at" "1255_g_at" "1294_at"
```

```
head(probeid2entrez$PROBEID,n=7)
```

```
[1] "1007_s_at" "1053_at" "117_at"
[4] "121_at" "1255_g_at" "1294_at"
[7] "1316_at"
```

```
head(indices)
```

```
[1] 1 2 3 4 5 6
```

Elige la primera correspondencia e ignora las demas como hemos visto.

```
eset = gse1397
fData(eset) = probeid2entrez[indices,]
```

Podemos comprobar, con Base::all.equal, si son iguales los nombres del ExpressionSet con la columna de identificadores Affymetrix.

```
all.equal(fData(eset)$PROBEID,featureNames(eset))
```

```
[1] TRUE
```

18.10 Ejercicios

* **Ex. 36** — Determinar los códigos Entrez, Gene Ontology y Ensembl para los genes BRCA1 y BRCA2 implicados en el cáncer de mama.

* **Ex. 37** — Se pide encontrar el gen BRCA1 en el ratón (Mus musculus). Utilizad el paquete de anotación [23, org.Mm.eg.db].

Apéndice A

Matrices

En este tema incluidos definiciones y conceptos necesarios en el resto del manual sobre matrices. Una buena referencia para todo el tema es [79, Appendix A].

A.1 Determinantes

Ver [13, Capítulo 10]. Sea π una permutación de $(1, \ldots, n)$. El signo de la permutación π, $\sigma(\pi)$, se define a partir del número de intercambios que hemos de realizar en $(\pi(1), \ldots, \pi(n))$ para restablecer el orden natural. Si este número de intercambios es par entonces $\sigma(\pi) = 1$ y si es impar entonces $\sigma(\pi) = -1$.

Definición A.1 (Determinante)**.** *Si $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ entonces definimos el determinante de $\boldsymbol{A}$ denotado como $|\boldsymbol{A}|$ o $det(\boldsymbol{A})$ como*

$$det(\boldsymbol{A}) = \sum_{\pi} \sigma(\pi) a_{1\pi(1)} \cdots a_{n\pi(n)},$$

siendo $(\pi(1), \ldots, \pi(n))$ la permutación de $(1, \ldots, n)$. El sumatorio se extiende sobre todas las posibles permutaciones.

Equivalentemente

$$det(\boldsymbol{A}) = \sum_{\pi} (-1)^{\phi(\pi)} a_{1\pi(1)} \cdots a_{n\pi(n)}, \tag{A.1}$$

donde $\phi(\pi)$ denota el mínimo número de intercambios que debemos aplicar a la permutación para restablecer el orden natural.

A.2 Matriz ortogonal

Definición A.2 (Matriz ortogonal)**.** *Una matriz ortogonal es una matriz cuadrada cuyas filas y columnas son vectores unitarios ortogonales.*

En consecuencia si $\boldsymbol{A} \in \mathbb{R}^{n \times n}$ es ortogonal si

$$\boldsymbol{A}^T \boldsymbol{A} = \boldsymbol{A} \boldsymbol{A}^T = \boldsymbol{I}_n.$$

Si $\boldsymbol{A}$ es ortogonal necesariamente no es singular (tiene inversa) y

$$\boldsymbol{A}^{-1} = \boldsymbol{A}^T.$$

El determinante de una matriz ortogonal ha de ser $+1$ o -1 y conserva el producto escalar. Notemos que si i $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es ortogonal entonces $\|\boldsymbol{Ax}\| = (\boldsymbol{Ax})^T(\boldsymbol{Ax}) = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{A}^T\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{x} = \|\boldsymbol{x}\|$, es decir, mantenemos la longitud del vector. Tenemos una rotación o el simétrico respecto del origen.

A.3 Valores y vectores propios

Consultar para una introducción más completa [13, Capítulo 11].

Definición A.3. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ entonces $\boldsymbol{x} \in \mathbb{R}^n$ es un vector propio de $\boldsymbol{A}$ con valor propio λ si $\boldsymbol{Ax} = \lambda\boldsymbol{x}$.*

λ es un valor propio de $\boldsymbol{A}\mathbb{R}^{n\times n}$ si y solamente si el sistema $(\boldsymbol{A} - \lambda\boldsymbol{I})\boldsymbol{x} = \boldsymbol{0}$ tiene una solución no trivial no nula $\boldsymbol{x} \neq \boldsymbol{0}$. De otro modo, cualquier λ para el cual $\boldsymbol{A} - \lambda\boldsymbol{I}$ es singular es un valor propio de $\boldsymbol{A}$. λ es un valor propio de $\boldsymbol{A}$ si y solo si $null(\boldsymbol{A} - \lambda\boldsymbol{I})$ tiene un elemento no nulo.

A.4 Traza y valores propios

Denotamos por tr la traza de la matriz, esto es, la suma de los elementos de la diagonal principal.

Proposición A.1. *Suponiendo matrices conformables se verifica*

1. $tr(\boldsymbol{A} + \boldsymbol{B}) = tr(\boldsymbol{A}) + tr(\boldsymbol{B})$.
2. $tr(\boldsymbol{AC}) = tr(\boldsymbol{CA})$.

La prueba de la proposición es inmediata.
El polinomio característico es $det(\lambda\boldsymbol{I} - \boldsymbol{A})$.

Proposición A.2. *El polinomio característico de $\boldsymbol{A}$ es el mismo que el de su traspuesta.*

Prueba. El determinante de una matriz es igual al determinante de su traspuesta, por tanto, $det(\lambda\boldsymbol{I} - \boldsymbol{A}) = det(\lambda\boldsymbol{I} - \boldsymbol{A}^T)$. □

Proposición A.3. *Si $A \in \mathbb{R}^{n\times n}$ con valores propios $\lambda_1, \ldots, \lambda_n$ entonces*

$$tr(\boldsymbol{A}) = \sum_{i=1}^{n} \lambda_i.$$

y

$$det(\boldsymbol{A}) = \prod_{i=1}^{n} \lambda_i.$$

Prueba.

$$det(\lambda\boldsymbol{I}-\boldsymbol{A}) = \prod_{i=1}^{n}(\lambda-\lambda_i) = \lambda^n - \lambda^{n-1}(\lambda_1+\ldots+\lambda_n)+\ldots+(-1)^n(\lambda_1\cdots\lambda_n).$$

Expresamos el polinomio característico como

$$det(\lambda\boldsymbol{I} - \boldsymbol{A}) = \lambda^n + c_{n-1}\lambda^{n-1} + \ldots + c_1\lambda + c_0.$$

Comparando la constante en ambas expresiones se sigue que

$$c_0 = (-1)^n(\lambda_1 \cdots \lambda_n).$$

Si tomamos $\lambda = 0$ entonces tendremos

$$c_0 = det(0\boldsymbol{I} - \boldsymbol{A}) = det(-\boldsymbol{A}) = (-1)^n det(\boldsymbol{A})$$

Por tanto:

$$(-1)^n(\lambda_1 \cdots \lambda_n) = c_0 = (-1)^n det(\boldsymbol{A}).$$

En resumen, $det(\boldsymbol{A}) = \lambda_1 \cdots \lambda_n$.

El término que acompaña a λ^{n-1} en la factorización inicial es $-(\lambda_1 + \ldots + \lambda_n)$. Se sigue que

$$c_{n-1} = -(\lambda_1 + \ldots + \lambda_n).$$

Si vemos la expresión del determinante es claro que en los distintos sumandos solamente puede aparecer el término λ^{n-1} en el sumando $\prod_{i=1}^n(\lambda - a_{ii})$. En particular, es el coefiente que en este productorio acompaña a λ^{n-1}. Esto es, $c_{n-1} = -\sum_{i=1}^n a_{ii} = -tr(\boldsymbol{A})$. Se sigue

$$\lambda_1 + \ldots + \lambda_n = -c_{n-1} = tr(\boldsymbol{A}).$$

□

Proposición A.4 (Teorema del eje principal)**.** *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ simétrica entonces existe una matriz ortogonal $\boldsymbol{T} = [\boldsymbol{t}_1|\ldots|\boldsymbol{t}_n]$ tal que*

$$\boldsymbol{T}^T\boldsymbol{A}\boldsymbol{T} = \Lambda,$$

*con $\Lambda = diag(\lambda_1, \ldots, \lambda_n)$ siendo λ_i los valores propios de $\boldsymbol{A}$ y $\boldsymbol{A}\boldsymbol{t}_i = \lambda_i\boldsymbol{t}_i$. Los vectores propios $\boldsymbol{t}_i$ forman una base ortonormal de $\mathbb{R}^n$. La descomposición $\boldsymbol{A} = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T$ recibe el nombre de **descomposición espectral** de la matriz $\boldsymbol{A}$.*

Proposición A.5. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es simétrica: $tr(\boldsymbol{A}^k) = \sum_{i=1}^n \lambda_i^k$.*

Prueba. Los valores propios de $\boldsymbol{A}^k$ son λ_i^k con vector propio asociado $\boldsymbol{t}_i$. □

Proposición A.6. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es simétrica y tiene inversa entonces los valores propios de $\boldsymbol{A}^{-1}$ son λ_i^{-1} y $tr(\boldsymbol{A}^{-1}) = \sum_{i=1}^n \lambda_i^{-1}$.*

Prueba. $\boldsymbol{A}\boldsymbol{t}_i = \lambda_i\boldsymbol{t}_i$ de donde $\boldsymbol{A}^{-1}\boldsymbol{A}\boldsymbol{t}_i = \lambda_i\boldsymbol{A}^{-1}\boldsymbol{t}_i$. Finalmente $\boldsymbol{A}^{-1}\boldsymbol{t}_i = \frac{1}{\lambda_i}\boldsymbol{t}_i$. □

Proposición A.7. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es simétrica. Los valores propios de $\boldsymbol{I}_n + c\boldsymbol{A}$ son $1 + c\lambda_i$ con $i = 1, \ldots, n$.*

Prueba. $(\boldsymbol{I}_n + c\boldsymbol{A})\boldsymbol{t}_i = \boldsymbol{t}_i + c\lambda_i\boldsymbol{t}_i = (1 + c\lambda_i)\boldsymbol{t}_i$. □

A.5 Espacio columna, espacio nulo y rango de una matriz

Tenemos $A \in \mathbb{R}^{n\times p}$.

Definición A.4 (Espacio columna de A)**.**

$$C(A) = \{\boldsymbol{y} \in \mathbb{R}^n : \boldsymbol{y} = \boldsymbol{A}\boldsymbol{x}, \boldsymbol{x} \in \mathbb{R}^p\}.$$

Definición A.5 (Espacio nulo de A)**.**

$$null(A) = \{\boldsymbol{x} \in \mathbb{R}^p : \boldsymbol{A}\boldsymbol{x} = \boldsymbol{0}\}.$$

Si $A = [\boldsymbol{a}_1|\ldots|\boldsymbol{a}_p]$ donde $\boldsymbol{a}_i$ es la i-ésima columna entonces

$$C(A) = span\{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_p\}.$$

Definición A.6 (Rango de A)**.** *El rango de una matriz es*

$$rank(A) = dim(C(A)).$$

Decimos que A no es de rango completo si $rank(A) < \min\{n, p\}$. El rango de una matriz es el máximo número de filas o columnas linealmente independientes.

Si denotamos por espacio columna y espacio fila el espacio vectorial generado por los vectores columna y por los vectores filas entonces se tiene el siguiente resultado que justifica la afirmación anterior.

Proposición A.8. *El número de filas linealmente independientes coincide con el número de columnas linealmente independientes.*[110]

[110] https://en.wikipedia.org/wiki/Rank_(linear_algebra).

Prueba. Sea $\boldsymbol{A} \in \mathbb{R}^{n\times p}$. Supongamos que el número de filas linealmente independientes es r. El espacio generado por las filas o **espacio fila** tiene dimensión r y $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_r\}$ es una base. Veamos que $\boldsymbol{A}\boldsymbol{x}_1, \boldsymbol{A}\boldsymbol{x}_2, \ldots, \boldsymbol{A}\boldsymbol{x}_r$ son linealmente independientes. Supongamos coeficientes $c_1, c_2, \ldots, c_r$ tales que

$$\boldsymbol{0} = c_1\boldsymbol{A}\boldsymbol{x}_1 + c_2\boldsymbol{A}\boldsymbol{x}_2 + \ldots + c_r\boldsymbol{A}\boldsymbol{x}_r = \boldsymbol{A}(c_1\boldsymbol{x}_1 + c_2\boldsymbol{x}_2 + \cdots + c_r\boldsymbol{x}_r) = \boldsymbol{A}\boldsymbol{v}$$

con $\boldsymbol{v} = c_1\boldsymbol{x}_1 + c_2\boldsymbol{x}_2 + \cdots + c_r\boldsymbol{x}_r$. Es claro que $\boldsymbol{v}$ está en el espacio fila. Pero $\boldsymbol{A}\boldsymbol{v} = \boldsymbol{0}$ por lo que $\boldsymbol{v}$ es ortogonal a cada fila de $\boldsymbol{A}$. Es decir, $\boldsymbol{v}$ está en el espacio ortogonal al espacio fila. En consecuencia $\boldsymbol{v}$ es ortogonal a sí mismo $\boldsymbol{v} \perp \boldsymbol{v}$ lo que supone que $\boldsymbol{v}$ es el vector nulo. Se sigue: $c_1\boldsymbol{x}_1 + c_2\boldsymbol{x}_2 + \cdots + c_r\boldsymbol{x}_r = \boldsymbol{0}$ y, por lo tanto, $c_1 = c_2 = \ldots = c_r = 0$. En resumen, $\boldsymbol{A}\boldsymbol{x}_1, \boldsymbol{A}\boldsymbol{x}_2, \ldots, \boldsymbol{A}\boldsymbol{x}_r$ son linealmente independientes. Pero $\boldsymbol{A}\boldsymbol{x}_i$ están en el espacio columna (engendrado por los vectores columna). En resumen tenemos al menos r vectores linealmente independientes en el espacio columna. La dimensión del espacio ha de ser mayor o igual a r. El rango por filas (o dimension del espacio filas) es menor o igual al rango por columnas (o dimensión del espacio columna). Aplicamos este resultado a la matriz traspuesta y el resultado se sigue. □

Proposición A.9. *Si $\boldsymbol{A}$ y $\boldsymbol{B}$ son matrices conformes entonces*

$$rank(\boldsymbol{A}\boldsymbol{B}) \leq \min\{rank(\boldsymbol{A}), rank(\boldsymbol{B})\}.$$

Prueba. Las filas de $\boldsymbol{AB}$ son combinaciones lineales de las filas de $\boldsymbol{B}$. El número de filas linealmente independientes de $\boldsymbol{AB}$ ha de ser menor o igual al número de filas linealmente independientes de $\boldsymbol{B}$, es decir, $rank(\boldsymbol{AB}) \leq rank(\boldsymbol{B})$. Las columnas de $\boldsymbol{AB}$ son combinaciones lineales de las columnas de $\boldsymbol{A}$. Por tanto, $rank(\boldsymbol{AB}) \leq rank(\boldsymbol{A})$. □

Proposición A.10. *Si $\boldsymbol{A}$ es una matriz arbitraria. $\boldsymbol{P}$ y $\boldsymbol{Q}$ son matrices conformes no singulares entonces $rank(\boldsymbol{PAQ}) = rank(\boldsymbol{A})$.*

Prueba. $rank(\boldsymbol{A}) \leq rank(\boldsymbol{AQ}) \leq rank(\boldsymbol{AQQ}^{-1}) = rank(\boldsymbol{A})$ de donde $rank(\boldsymbol{A}) = rank(\boldsymbol{AQ})$. Análogamente llegamos al resultado. □

Proposición A.11. *$\boldsymbol{A} \in \mathbb{R}^{n\times p}$ tal que $r = rank(\boldsymbol{A})$ y s es la dimensión de $null(\boldsymbol{A})$. Entonces: $r + s = p$.*

Prueba. Sea $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s$ una base de $null(\boldsymbol{A})$. Completamos esta base hasta tener una base de $\mathbb{R}^p$: $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_s, \boldsymbol{y}_1, \ldots, \boldsymbol{y}_t\}$. Un vector del espacio columna de la matriz $\boldsymbol{A}$ verifica

$$\boldsymbol{Ax} = \boldsymbol{A}\left(\sum_{i=1}^{s} a_i\boldsymbol{x}_i + \sum_{j=1}^{t} b_j\boldsymbol{y}_j\right) = \boldsymbol{A}\sum_{j=1}^{t} b_j\boldsymbol{y}_j = \sum_{j=1}^{t} b_j\boldsymbol{Ay}_j = \sum_{j=1}^{t} b_j\boldsymbol{\gamma}_j. \quad \text{(A.2)}$$

Supongamos que $\sum_{j=1}^{t} c_j\boldsymbol{\gamma}_j = \boldsymbol{0}$. Se tiene

$$\boldsymbol{A}\left(\sum_{j=1}^{t} c_j\boldsymbol{y}_j\right) = \sum_{j=1}^{t} c_j\boldsymbol{\gamma}_j = \boldsymbol{0}.$$

Por tanto se tiene $\sum_{j=1}^{t} c_j\boldsymbol{y}_j \in null(\boldsymbol{A})$. Pero $\sum_{j=1}^{t} c_j\boldsymbol{y}_j \in null(\boldsymbol{A})^{\perp}$ de donde $c_1 = \ldots = c_t = 0$. Los vectores $\boldsymbol{\gamma}_j = \boldsymbol{Ay}_j$ son linealmente independientes. Cada vector $\boldsymbol{Ax}$ del espacio columna de $\boldsymbol{A}$ puede expresarse como combinación lineal de los $\boldsymbol{\gamma}_j$. En resumen los $\boldsymbol{\gamma}_j$ forman una base del espacio columna. Se sigue que $t = r$. Puesto que $s+t = p$ se sigue el resultado. □

Proposición A.12. $rank(\boldsymbol{A}) = rank(\boldsymbol{A}^T) = rank(\boldsymbol{AA}^T) = rank(\boldsymbol{A}^T\boldsymbol{A})$.

Prueba. $\boldsymbol{Ax} = \boldsymbol{0} \rightarrow \boldsymbol{A}^T\boldsymbol{Ax} = \boldsymbol{0}$. Pero $\boldsymbol{A}^T\boldsymbol{Ax} = \boldsymbol{0} \rightarrow \boldsymbol{x}^T\boldsymbol{A}^T\boldsymbol{Ax}$ de donde $\boldsymbol{Ax} = \boldsymbol{0}$. Tenemos que los espacios nulos de las matrices $\boldsymbol{A}$ y $\boldsymbol{A}^T\boldsymbol{A}$ son el mismo. Pero $\boldsymbol{A}$ y $\boldsymbol{A}^T\boldsymbol{A}$ tienen el mismo número de columnas. Utilizando la proposición A.11 tenemos que $rank(\boldsymbol{A}) = rank(\boldsymbol{A}^T\boldsymbol{A})$. De un modo análogo, $rank(\boldsymbol{A}^T) = rank(\boldsymbol{A}^T\boldsymbol{A})$ y tenemos el resultado. □

Proposición A.13. $C(\boldsymbol{A}^T\boldsymbol{A}) = C(\boldsymbol{A}^T)$.

Prueba. $\boldsymbol{A}^T\boldsymbol{Ax} = \boldsymbol{A}^T\boldsymbol{y}$ con $\boldsymbol{y} = \boldsymbol{Ax}$. Tenemos pues que $C(\boldsymbol{A}^T\boldsymbol{A}) \subset C(\boldsymbol{A}^T)$. Pero por la proposición A.12 tienen la misma dimensión por lo que han de ser iguales. □

Proposición A.14. *Si $\boldsymbol{A}$ es simétrica entonces $rank(\boldsymbol{A})$ es el número de valores propios no nulos.*

Prueba. Por la descomposición espectral tenemos una matriz ortogonal tal que $\boldsymbol{T}^T\boldsymbol{A}\boldsymbol{T} = \Lambda$ con Λ diagonal. Utilizando A.10 tenemos que $rank(\boldsymbol{A}) = rank(\boldsymbol{T}^T\boldsymbol{A}\boldsymbol{T}) = rank(\Lambda)$ que obviamente tiene como dimensión del espacio columna el número de valores propios no nulos. □

Proposición A.15. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ simétrica entonces existen n vectores propios ortonormales y el espacio columna $C(\boldsymbol{A})$ es el espacio engendrado por los vectores propios correspondientes a los valores propios no nulos.*

Prueba. Realizamos la descomposición espectral $\boldsymbol{T}^T\boldsymbol{A}\boldsymbol{T} = \Lambda$. Se sigue $\boldsymbol{A}\boldsymbol{T} = \boldsymbol{T}\Lambda$. Si $\boldsymbol{T} = [\boldsymbol{t}_1|\dots|\boldsymbol{t}_n]$ entonces los $\boldsymbol{t}_i$ son ortonormales ya que $\boldsymbol{T}$ es ortogonal. Además $\boldsymbol{A}\boldsymbol{t}_i = \lambda_i\boldsymbol{t}_i$. Supongamos que los valores propios λ_i son nulos de $r+1$ a n. Consideramos $\boldsymbol{x} = \sum_{i=1}^n a_i\boldsymbol{t}_i$. Tenemos

$$\boldsymbol{A}\boldsymbol{x} = \boldsymbol{A}\sum_{i=1}^n a_i\boldsymbol{t}_i = \sum_{i=1}^n a_i\boldsymbol{A}\boldsymbol{t}_i = \sum_{i=1}^r a_i\lambda_i\boldsymbol{t}_i = \sum_{i=1}^r b_i\boldsymbol{t}_i.$$

El espacio columna, $C(\boldsymbol{A})$, está generado por los vectores $\boldsymbol{t}_1,\dots,\boldsymbol{t}_r$. □

A.6 Matrices semidefinidas positivas

Definición A.7. *Una matriz simétrica $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ se dice semidefinida positiva si $x^T\boldsymbol{A}x \geq 0$ para cualquier x.*

Proposición A.16. *Los valores propios de una matriz semidefinida positiva son no negativos.*

Prueba. Por ser simétrica consideramos su descomposición espectral y tenemos una matriz ortogonal $\boldsymbol{T}$ tal que $\boldsymbol{T}^T\boldsymbol{A}\boldsymbol{T} = \Lambda$ siendo $\Lambda = diag(\lambda_1,\dots,\lambda_n)$. Tenemos que si $\boldsymbol{x} = \boldsymbol{T}\boldsymbol{y}$ entonces $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}^T\boldsymbol{T}^T\boldsymbol{A}\boldsymbol{T}\boldsymbol{y} = \sum_{i=1}^n \lambda_i y_i^2 \geq 0$. Si consideramos el vector $\boldsymbol{y}$ tal que $y_j = \delta_{ij}$ entonces $\lambda_i = \boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} \geq 0$. □

Proposición A.17. *Si $\boldsymbol{A}$ es semidefinida positiva (o definida no negativa) entonces $tr(A) \geq 0$.*

Prueba. Por A.3, $tr(\boldsymbol{A}) = \sum_{i=1}^n \lambda_i$ y estos λ_i son no negativos por A.16. □

Proposición A.18. *$\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es semidefinida positiva de rango r si y solo si existe $\boldsymbol{B} \in \mathbb{R}^{n\times n}$ de rango r tal que $\boldsymbol{A} = \boldsymbol{B}\boldsymbol{B}^T$.*

Prueba. $\boldsymbol{A}$ es simétrica por lo que su rango coincide con los valores propios no nulos (proposición A.14). Estos valores propios son no negativos (proposición A.16). La matriz diagonal de valores propios será $\boldsymbol{\Lambda} = diag(\lambda_1,\dots,\lambda_r,0,\dots,0)$ y la descomposición espectral de $\boldsymbol{A}$ nos da $\boldsymbol{T}^T\boldsymbol{A}\boldsymbol{T} = \boldsymbol{\Lambda}$. Si $\boldsymbol{\Lambda}^{1/2} = diag(\lambda_1^{1/2},\dots,\lambda_r^{1/2},0,\dots,0)$ entonces $\boldsymbol{A} = \boldsymbol{T}\boldsymbol{\Lambda}^{1/2}\boldsymbol{\Lambda}^{1/2}\boldsymbol{T}^T = \boldsymbol{B}\boldsymbol{B}^T$ con $\boldsymbol{B} = \boldsymbol{T}\boldsymbol{\Lambda}^{1/2}$. Pero $rank(\boldsymbol{B}) = rank(\boldsymbol{\Lambda}^{1/2}) = r$ por ser $\boldsymbol{T}$ no singular (proposicion A.10).

Si $\boldsymbol{A} = \boldsymbol{B}\boldsymbol{B}^T$ entonces $rank(\boldsymbol{A}) = rank(\boldsymbol{B}) = r$ (proposición A.12). Pero $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{x}^T\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{x} = \boldsymbol{y}^T\boldsymbol{y} \geq 0$ con $\boldsymbol{y} = \boldsymbol{B}^T\boldsymbol{x}$. □

Proposición A.19. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es semidefinida positiva de rango r entonces existe $\boldsymbol{B} \in \mathbb{R}^{n\times r}$ de rango r tal que $\boldsymbol{B}^T\boldsymbol{A}\boldsymbol{B} = \boldsymbol{I}_r$.*

Prueba. La descomposición espectral nos da

$$\boldsymbol{T}^T\boldsymbol{A}\boldsymbol{T} = \begin{bmatrix} \boldsymbol{\Lambda}_r & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix}.$$

Si $\boldsymbol{T}_1$ denota las r primeras columnas de $\boldsymbol{T}$ entonces tomamos $\boldsymbol{B} = \boldsymbol{T}_1\boldsymbol{\Lambda}_r^{1/2}$. □

Proposición A.20. *Si $\boldsymbol{A}$ es semidefinida positiva entonces $\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X} = \boldsymbol{0}$ implica que $\boldsymbol{A}\boldsymbol{X} = \boldsymbol{0}$.*

Prueba. Por ser semidefinida positiva existe $\boldsymbol{B}$ (prop A.18) tal que $\boldsymbol{A} = \boldsymbol{B}\boldsymbol{B}^T$. Tenemos $\boldsymbol{0} = \boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X} = \boldsymbol{X}^T\boldsymbol{B}\boldsymbol{B}^T\boldsymbol{X} = \boldsymbol{C}^T\boldsymbol{C}$ con $\boldsymbol{C} = \boldsymbol{B}^T\boldsymbol{X}$. Si $\boldsymbol{B} = [\boldsymbol{b}_1|\dots|\boldsymbol{b}_n]$ entonces $\boldsymbol{b}_i^T\boldsymbol{b}_i = 0$ y $\boldsymbol{A}\boldsymbol{X} = \boldsymbol{B}\boldsymbol{B}^T\boldsymbol{X} = \boldsymbol{0}$. □

A.7 Matrices definidas positivas

Definición A.8. *Una matriz simétrica $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ se dice definida positiva si $x^T\boldsymbol{A}x > 0$ para cualquier $x \neq \boldsymbol{0}$.*

Proposición A.21. *Los valores propios de una matriz definida positiva son todos positivos. Y por lo tanto, es no singular.*

Prueba. Prueba análoga a la de proposición A.16. Si todos los valores propios son positivos entonces por ser simétrica su rango coincide con el número de valores propios no nulos (proposición A.14). □

Proposición A.22. *Una matriz $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es definida positiva sí y solo si existe una matriz no singular $\boldsymbol{R}$ tal que $\boldsymbol{A} = \boldsymbol{R}\boldsymbol{R}^T$.*

Prueba. Es consecuencia de la proposición A.18 y de la proposición A.21. □

Proposición A.23. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es definida positiva entonces $\boldsymbol{A}^{-1}$ también es definida positiva.*

Prueba. Por la proposición A.22, $\boldsymbol{A}^{-1} = (\boldsymbol{R}\boldsymbol{R}^T)^{-1} = \boldsymbol{R}^{-T}\boldsymbol{R}^{-1} = \boldsymbol{S}\boldsymbol{S}^T$. Y por la proposición A.22 se sigue el resultado. □

Proposición A.24. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es definida positiva entonces entonces $rank(\boldsymbol{C}\boldsymbol{A}\boldsymbol{C}^T) = rank(\boldsymbol{C})$.*

Prueba. Por proposición A.22, $\boldsymbol{A} = \boldsymbol{R}\boldsymbol{R}^T$, y $rank(\boldsymbol{C}\boldsymbol{A}\boldsymbol{C}^T) = rank(\boldsymbol{C}\boldsymbol{R}\boldsymbol{R}^T\boldsymbol{C}^T)$. Por proposición A.12, $rank(\boldsymbol{C}\boldsymbol{R}\boldsymbol{R}^T\boldsymbol{C}^T) = rank(\boldsymbol{C}\boldsymbol{R})$. Pero $\boldsymbol{R}$ es no singular y por proposición A.10 tenemos que $rank(\boldsymbol{C}\boldsymbol{R}) = rank(\boldsymbol{C})$. □

Proposición A.25. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es definida positiva y $\boldsymbol{C} \in \mathbb{R}^{p\times n}$ de rango p entonces $\boldsymbol{C}\boldsymbol{A}\boldsymbol{C}^T$ es definida positiva.*

Prueba. Si $\boldsymbol{x}^T\boldsymbol{C}\boldsymbol{A}\boldsymbol{C}^T\boldsymbol{x} = \boldsymbol{y}^T\boldsymbol{A}\boldsymbol{y} \geq 0$. La igualdad a cero es equivalente a que $\boldsymbol{y} = \boldsymbol{0}$ que equivale con $\boldsymbol{C}^T\boldsymbol{x} = \boldsymbol{0}$. Pero las columnas de $\boldsymbol{C}^T$ son linealmente independientes pues tiene rango p. En consecuencia $\boldsymbol{C}^T\boldsymbol{x} = \boldsymbol{0}$ equivale con $\boldsymbol{x} = \boldsymbol{0}$. En resumen, $\boldsymbol{x}^T\boldsymbol{C}\boldsymbol{A}\boldsymbol{C}^T\boldsymbol{x} > 0$ para cualquier $\boldsymbol{x}$ tal que $\boldsymbol{x} \neq \boldsymbol{0}$. □

Proposición A.26. *Si $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ es una matriz de rango p entonces $\boldsymbol{X}^T\boldsymbol{X}$ es definida positiva.*

Prueba. Si $\boldsymbol{x}^T\boldsymbol{X}^T\boldsymbol{X}\boldsymbol{x} = \boldsymbol{y}^T\boldsymbol{y} \geq 0$. Pero la igualdad a cero equivale con $\boldsymbol{X}\boldsymbol{x} = \boldsymbol{0}$ que equivale con $\boldsymbol{x} = \boldsymbol{0}$ ya que asumimos independientes las columnas de $\boldsymbol{X}$. □

Proposición A.27. *$\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es definida positiva y solo si todos los determinantes de los menores principales (incluyendo toda la matriz) son positivos.*

Prueba. Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es definida positiva entonces, aplicando la proposición A.21, se sigue

$$det(\boldsymbol{A}) = det(\boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T) = det(\boldsymbol{\Lambda}) = \prod_{i=1}^{n} \lambda_i > 0.$$

Tomamos un menor principal eliminando las últimas $n - r + 1$ filas y columnas. Este menor será

$$\boldsymbol{A}_r = \begin{bmatrix} a_{11} & \cdots & a_{1r} \\ \vdots & \vdots & \vdots \\ a_{r1} & \cdots & a_{rr} \end{bmatrix}$$

y $\boldsymbol{x}_r^T = (x_1, \ldots, x_r)$. Tenemos que

$$\boldsymbol{x}_r^T\boldsymbol{A}_r\boldsymbol{x}_r = (\boldsymbol{x}_r^T, \boldsymbol{0}^T)\boldsymbol{A}\begin{pmatrix} \boldsymbol{x}_r \\ \boldsymbol{0} \end{pmatrix} > 0$$

para $\boldsymbol{x}_r \neq \boldsymbol{0}$. Se sigue que $\boldsymbol{A}_r$ es definida positiva.

Veamos el recíproco. Suponemos que todos los menores principales tienen determinantes positivos y queremos ver que la matriz $\boldsymbol{A}$ es definida positiva. Tomamos

$$\boldsymbol{A} = \begin{bmatrix} \boldsymbol{A}_{n-1} & \boldsymbol{c} \\ \boldsymbol{c}^T & a_{nn} \end{bmatrix}, \quad \boldsymbol{R} = \begin{bmatrix} \boldsymbol{I}_{n-1} & \boldsymbol{\alpha} \\ \boldsymbol{0}^T & -1 \end{bmatrix},$$

siendo $\boldsymbol{\alpha} = \boldsymbol{A}_{n-1}^{-1}\boldsymbol{c}$. Entonces

$$\boldsymbol{R}^T\boldsymbol{A}\boldsymbol{R} = \begin{bmatrix} \boldsymbol{A}_{n-1} & \boldsymbol{0} \\ \boldsymbol{0}^T & k \end{bmatrix}$$

siendo

$$k = det(\boldsymbol{R}^T\boldsymbol{A}\boldsymbol{R})/det(\boldsymbol{A}_{n-1}) = det(\boldsymbol{R})^2 det(\boldsymbol{A})/det(\boldsymbol{A}_{n-1}) > 0,$$

donde $\boldsymbol{R}$ es no singular. Se prueba por inducción. El resultado es cierto para $n = 1$ y lo asumimos para matrices hasta de orden $n - 1$. Tomamos $y = \boldsymbol{R}^{-1}\boldsymbol{x}$ con $\boldsymbol{x} \neq \boldsymbol{0}$, $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = \boldsymbol{y}^T\boldsymbol{R}^T\boldsymbol{A}\boldsymbol{R}\boldsymbol{y} = \boldsymbol{y}_{n-1}^T\boldsymbol{A}_{n-1}\boldsymbol{y}_{n-1} + ky_m^2 > 0$ ya que $\boldsymbol{A}_{n-1}$ es definida positiva por hipótesis de inducción con $\boldsymbol{y} \neq \boldsymbol{0}$. Y el resultado se sigue. □

Proposición A.28. *Los elementos de la diagonal principal de una matriz definida positiva son todos positivos.*

Prueba. Tomamos $x_j = \delta_{ij}$ para $j = 1, \ldots, n$ y $\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} = a_{ii} > 0$. □

Proposición A.29. *Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es definida positiva y $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es una matriz simétrica entonces $\boldsymbol{A} - t\boldsymbol{B}$ es definida positiva para $|t|$ suficientemente pequeño.*

Prueba. Por la proposición A.27 el determinante del i-th menor principal de la matriz $\boldsymbol{A} - t\boldsymbol{B}$ es positivo cuando $t = 0$. Pero este determinante es una función continua por lo que será positiva cuando $|t| < \delta_i$ para δ_i suficientemente pequeño. Tomamos $\delta = \min\{\delta_1, \ldots, \delta_n\}$. Tenemos que los determinantes de los menores principales de $\boldsymbol{A} - t\boldsymbol{B}$ son positivos cuando $|t| < \delta$. Utilizando la proposición A.27 se sigue el resultado. □

Proposición A.30. *[Raiz cuadrada de matrices definidas positivas] Si $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ es definida positiva entonces existe una matriz definida positiva que llamamos su raiz cuadrada y denotamos $\boldsymbol{A}^{1/2}$ tal que $(\boldsymbol{A}^{1/2})^2 = \boldsymbol{A}$.*

Prueba. Consideramos la descomposición espectral de $\boldsymbol{A}$, $\boldsymbol{A} = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T$. Por la proposición A.21 todos valores propios son positivos. Por tanto los elementos de la diagonal de $\boldsymbol{\Lambda}$ son positivos. Definimos $\boldsymbol{A}^{1/2} = \boldsymbol{T}\boldsymbol{\Lambda}^{1/2}\boldsymbol{T}^T$ donde si $\boldsymbol{\Lambda} = diag(\lambda_1, \ldots, \lambda_n)$ entonces $\boldsymbol{\Lambda}^{1/2} = diag(\lambda_1^{1/2}, \ldots, \lambda_n^{1/2})$. Se tiene que $\boldsymbol{A}^{1/2}\boldsymbol{A}^{1/2} = \boldsymbol{T}\boldsymbol{\Lambda}^{1/2}\boldsymbol{T}^T\boldsymbol{T}\boldsymbol{\Lambda}^{1/2}\boldsymbol{T}^T = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T$ ya que $\boldsymbol{T}$ es ortogonal y $\boldsymbol{T}^T\boldsymbol{T} = \boldsymbol{I}$. □

A.8 Derivación

Veamos algunas expresiones de fácil prueba sobre gradientes de productos de matrices por vectores.

Proposición A.31. 1. *Si $\boldsymbol{a}, \boldsymbol{\beta} \in \mathbb{R}^p$*

$$\frac{\partial \boldsymbol{a}^T\boldsymbol{\beta}}{\partial \boldsymbol{\beta}} = \boldsymbol{a}. \tag{A.3}$$

2. *Si A un matriz $p \times p$ y $\boldsymbol{\beta} \in \mathbb{R}^p$ entonces*

$$\frac{\partial \boldsymbol{\beta}^T\boldsymbol{A}\boldsymbol{\beta}}{\partial \boldsymbol{\beta}} = (A + A^T)\boldsymbol{\beta}. \tag{A.4}$$

3. *Adicionalmente, en las condiciones de 2, Si $A \in \mathbb{R}^{p\times p}$ es simétrica entonces*

$$\frac{\partial \boldsymbol{\beta}^T\boldsymbol{A}\boldsymbol{\beta}}{\partial \boldsymbol{\beta}} = 2\boldsymbol{A}\boldsymbol{\beta}. \tag{A.5}$$

A.9 Matrices de proyección

Definición A.9 (Matriz de proyección)**.** *Una matriz cuadrada $\boldsymbol{P} \in \mathbb{R}^{p\times p}$ es una matriz de proyección sobre un subespacio vectorial S si*

1. *Para todo $\boldsymbol{y} \in S$, $\boldsymbol{P}\boldsymbol{y} = \boldsymbol{y}$.*
2. *Para todo $\boldsymbol{y} \in S^{\perp}$, $\boldsymbol{P}\boldsymbol{y} = \boldsymbol{0}$.*

De acuerdo con esta definición tenemos que si $\boldsymbol{y} \in S$, $\boldsymbol{y} = P\boldsymbol{y}$, por tanto, $\boldsymbol{y}$ es una combinación lineal de las columnas de la matriz $\boldsymbol{P}$. Si denotamos el espacio vectorial generado por las columnas de $\boldsymbol{P}$ como $C(\boldsymbol{P})$ tenemos que $S = C(\boldsymbol{P})$.

Proposición A.32. *Dado S un subespacio vectorial de $\mathbb{R}^p$, entonces cada vector $\boldsymbol{x}$ tiene una descomposición única como $\boldsymbol{x} = \boldsymbol{u} + \boldsymbol{v}$ con $\boldsymbol{u} \in S; \boldsymbol{v} \in S^\perp$.*

Prueba. Si hay dos $\boldsymbol{x} = \boldsymbol{u}_1 + \boldsymbol{v}_1 = \boldsymbol{u}_2 + \boldsymbol{v}_2$ entonces $(\boldsymbol{u}_1 - \boldsymbol{u}_2) + (\boldsymbol{v}_1 - \boldsymbol{v}_2) = \boldsymbol{0}$. Pero $\boldsymbol{u}_1 - \boldsymbol{u}_2 \in S$, $\boldsymbol{v}_1 - \boldsymbol{v}_2 \in S^\perp$ y $\boldsymbol{u}_1 - \boldsymbol{u}_2 = -(\boldsymbol{v}_1 - \boldsymbol{v}_2) \in S \cap S^\perp$. Se sigue que $\boldsymbol{u}_1 = \boldsymbol{u}_2$ y $\boldsymbol{v}_1 = \boldsymbol{v}_2$. □

Proposición A.33. *La matriz de proyección es única.*

Prueba. Si suponemos dos matrices distintas $\boldsymbol{P}_1$ y $\boldsymbol{P}_2$ entonces, puesto que la proyección es única para cada punto que proyectamos $\boldsymbol{y}$ tendremos $(\boldsymbol{P}_1 - \boldsymbol{P}_2)\boldsymbol{y} = \boldsymbol{u} - \boldsymbol{u} = \boldsymbol{0}$. En resumen $\boldsymbol{P}_1 = \boldsymbol{P}_2$. □

Proposición A.34. *La matriz de proyección $\boldsymbol{P}_S$ puede expresarse como $\boldsymbol{P}_S = \boldsymbol{T}\boldsymbol{T}^T$ siendo las columnas de $\boldsymbol{T}$ una base ortonormal.*

Prueba. Consideremos $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r$ una base ortonormal de S que suponemos de dimensión r. Completamos esta base hasta obtener una base ortonormal de $\mathbb{R}^p$: $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_r, \ldots, \boldsymbol{x}_p$. Se sigue:

$$\boldsymbol{y} = \sum_{i=1}^{p} c_i \boldsymbol{x}_i = \sum_{i=1}^{r} c_i \boldsymbol{x}_i + \sum_{i=r+1}^{p} c_i \boldsymbol{x}_i = \boldsymbol{u} + \boldsymbol{v}. \tag{A.6}$$

Obviamente: $\boldsymbol{u} \in S$, $\boldsymbol{v} \in S^\perp$. Además: $\boldsymbol{x}_i^T \boldsymbol{x}_j = \delta_{ij}$(con $\delta_{ij} = 1$ si $i = j$ y cero en otro caso) y $\boldsymbol{x}_i^T \boldsymbol{y} = c_i$. Se tiene que

$$\boldsymbol{u} = [\boldsymbol{x}_1 | \ldots | \boldsymbol{x}_r] \begin{bmatrix} \boldsymbol{x}_1^T y \\ \vdots \\ \boldsymbol{x}_r^T y \end{bmatrix} = \boldsymbol{T}\boldsymbol{T}^T \boldsymbol{y}. \tag{A.7}$$

[111] $[\boldsymbol{x}_1 | \ldots | \boldsymbol{x}_k]$ es la matriz con columnas $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k$.

[111] □

Proposición A.35. *Si $\boldsymbol{P}$ es la matriz de proyección sobre S entonces $\boldsymbol{I} - \boldsymbol{P}$ es la matriz de proyección sobre $S^\perp$.*

Prueba. Para $\boldsymbol{y} \in \mathbb{R}^p$ si consideramos $\boldsymbol{y} = \boldsymbol{y}_1 + \boldsymbol{y}_2$ con $\boldsymbol{y}_1 \in S$ y $\boldsymbol{y}_2 \in S^\perp$ entonces $\boldsymbol{P}\boldsymbol{y} = \boldsymbol{y}_1$ y $(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{y} = \boldsymbol{y} - \boldsymbol{y}_1 = \boldsymbol{y}_2$. Tenemos pues que

$$\boldsymbol{y} = \boldsymbol{P}\boldsymbol{y} + (\boldsymbol{I} - \boldsymbol{P})\boldsymbol{y},$$

nos da la única descomposición ortogonal de $\boldsymbol{y}$ y por lo tanto $\boldsymbol{I} - \boldsymbol{P}$ es la matriz de proyección sobre $S^\perp$. Además, $\boldsymbol{P}(\boldsymbol{I} - \boldsymbol{P})\boldsymbol{y} = \boldsymbol{0}$. □

Proposición A.36. *$\boldsymbol{P}$ es una matriz de proyección si y solo si es simétrica e idempotente.*

Prueba. Si $\boldsymbol{P}$ es simétrica e idempotente veamos que es una matriz de proyección. Si $\boldsymbol{v} \in C(\boldsymbol{P})$ entonces $\boldsymbol{v} = \boldsymbol{P}\boldsymbol{b}$ para algún $\boldsymbol{b}$. Pero $\boldsymbol{P}\boldsymbol{v} = \boldsymbol{P}(\boldsymbol{P}\boldsymbol{b}) = \boldsymbol{P}^2\boldsymbol{b} = \boldsymbol{P}\boldsymbol{b} = \boldsymbol{v}$. Si tomamos $\boldsymbol{v} \in C(\boldsymbol{P})^\perp$ entonces $\boldsymbol{P}^T\boldsymbol{v} = 0$. Pero $\boldsymbol{P}^T = \boldsymbol{P}$ de donde $\boldsymbol{P}\boldsymbol{v} = 0$. Tenemos pues que $\boldsymbol{P}$ es la matriz de proyección sobre $C(\boldsymbol{P})$.

Supongamos que $\boldsymbol{P}$ es la matriz de proyección sobre $C(\boldsymbol{P})$. Si $\boldsymbol{v} \in \mathbb{R}^p$ entonces $\boldsymbol{v} = \boldsymbol{v}_1 + \boldsymbol{v}_2$ con $\boldsymbol{v}_1 \in C(\boldsymbol{P})$ y $\boldsymbol{v}_2 \in C(\boldsymbol{P})^\perp$. Tenemos

$$\boldsymbol{P}^2\boldsymbol{v} = \boldsymbol{P}(\boldsymbol{P}(\boldsymbol{v}_1 + \boldsymbol{v}_2)) = P\boldsymbol{v}_1 = \boldsymbol{v}_1 = P\boldsymbol{v},$$

de donde, $\boldsymbol{P}^2 = \boldsymbol{P}$. Veamos que $\boldsymbol{P}$ es simétrica, $\boldsymbol{P}^T = \boldsymbol{P}$. Tomamos $\boldsymbol{w} \in \mathbb{R}^p$ y lo descomponemos $\boldsymbol{w} = \boldsymbol{w}_1 + \boldsymbol{w}_2$ con $\boldsymbol{w}_1 \in C(\boldsymbol{P})$ y $\boldsymbol{w}_2 \in C(\boldsymbol{P})^\perp$. Pero $\boldsymbol{I} - \boldsymbol{P}$ es la matriz de proyección sobre $C(\boldsymbol{P})^\perp$ de donde

$$\boldsymbol{w}^T \boldsymbol{P}^T (\boldsymbol{I} - \boldsymbol{P}) \boldsymbol{v} = \boldsymbol{w}_1^T \boldsymbol{v}_2 = 0,$$

y esto es cierto para cualesquiera $\boldsymbol{v}, \boldsymbol{w} \in \mathbb{R}^p$ de donde $\boldsymbol{P}^T(\boldsymbol{I} - \boldsymbol{P}) = \boldsymbol{0}$ y $\boldsymbol{P}^T = \boldsymbol{P}^T \boldsymbol{P}$. Pero $\boldsymbol{P}^T \boldsymbol{P}$ es simétrica y también lo son $\boldsymbol{P}^T$ y $\boldsymbol{P}$. □

Proposición A.37. *Los valores propios de una matriz de proyección son* 0 *o* 1.

Prueba. Si λ es un valor propio de $\boldsymbol{P}$ entonces $\boldsymbol{P}\boldsymbol{v} = \lambda \boldsymbol{v}$. Pero si $\boldsymbol{v} \in C(\boldsymbol{P})$ entonces $\boldsymbol{P}\boldsymbol{v} = \boldsymbol{v}$ y tendríamos $\lambda = 1$. Si $\boldsymbol{v} \in C(\boldsymbol{P})^\perp$ entonces $\boldsymbol{P}\boldsymbol{v} = \boldsymbol{0}$ y tendríamos $\lambda = 0$. □

Proposición A.38. *Si $\boldsymbol{P}$ es una matriz de proyección entonces su rango coincide con su traza.*

Prueba. La traza coincide con la suma de sus valores propios. Si la matriz es simétrica su rango es el número de valores propios no nulos. Los valores propios son 0 o 1 y en consecuencia la suma de sus valores propios coincide con el número de valores propios no nulos. □

Proposición A.39. *Supongamos $\boldsymbol{P}_i \in \mathbb{R}^{n \times n}$ simétricas y tales que $\sum_{i=1}^k \boldsymbol{P}_i = \boldsymbol{I}_n$. Esto es equivalente con que*

1. *$\boldsymbol{P}_i$ es idempotente para cada i.*
2. *$\boldsymbol{P}_i \boldsymbol{P}_j = \boldsymbol{0}$ para $i \neq j$.*
3. *$\sum_{i=1}^k rank(\boldsymbol{P}_i) = n$.*

A.10 Matrices idempotentes

Teorema A.1. *Si $\boldsymbol{P}$ es simétrica entonces $\boldsymbol{P}$ es idempotente y de rango r si y solamente si tiene r valores propios iguales a uno y $n - r$ valores propios iguales a cero.*

Prueba. Si $\boldsymbol{P}$ es idempotente entonces $\boldsymbol{P}^2 = \boldsymbol{P}$. Si $\boldsymbol{P}\boldsymbol{x} = \lambda \boldsymbol{x}$ con $\boldsymbol{x} \neq \boldsymbol{0}$ implica que $\lambda \boldsymbol{x}^T \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{P} \boldsymbol{x} = \boldsymbol{x}^T \boldsymbol{P}^2 \boldsymbol{x} = (\boldsymbol{P}\boldsymbol{x})^T (\boldsymbol{P}\boldsymbol{x}) = \lambda^2 \boldsymbol{x}^T \boldsymbol{x}$. Por tanto $\lambda(\lambda - 1) = 0$. Los valores propios pueden ser cero o uno. Pero por ser simétrica su rango coincide con el número de valores propios no nulos. La matriz $\boldsymbol{P}$ ha de tener r valores propios iguales a uno y $(n - r)$ iguales a cero.

Si los valores propios son 0 o 1 entonces podemos asumir los r primeros iguales a uno. Tenemos una matriz ortogonal $\boldsymbol{T}$ tal que

$$\boldsymbol{T}^T \boldsymbol{P} \boldsymbol{T} = \begin{bmatrix} \boldsymbol{I}_r & \boldsymbol{0} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} = \boldsymbol{\Lambda}$$

o bien

$$\boldsymbol{P} = \boldsymbol{T} \boldsymbol{\Lambda} \boldsymbol{T}^T.$$

Tenemos que $\boldsymbol{P}^2 = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T\boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T = \boldsymbol{T}\boldsymbol{\Lambda}^2\boldsymbol{T}^T = \boldsymbol{P}$ y $rank(\boldsymbol{P}) = r$ pues por ser simétrica su rango coincide con la suma de sus valores propios. □

A.11 Transformaciones de Householder y descomposición QR

A.11.1 Descomposición QR

Toda matriz real $\boldsymbol{X} \in \mathbb{R}^{n\times p}$ con $p \leq n$ se puede descomponer como

$$\boldsymbol{X} = \boldsymbol{Q}\begin{bmatrix}\boldsymbol{R}\\ \boldsymbol{0}_{(n-p)\times p}\end{bmatrix}$$

el producto de una matriz ortogonal $\boldsymbol{Q}$ por una matriz $\boldsymbol{R} \in \mathbb{R}^{p\times p}$ una triangular superior, esto es, si $\boldsymbol{R} = [r_{ij}]$ entonces $r_{ij} = 0$ para $i > j$ y $0_{(n-p)\times p} \in \mathbb{R}^{(n-p)\times p}$ es una matriz de ceros.

No hay una sola descomposición QR y se pueden utilizar distintos procedimientos. En la siguiente sección mostramos cómo hacerlo utilizando las descomposiciones de Householder.

A.12 Transformaciones de Householder

Tomamos un vector $\boldsymbol{u} \in \mathbb{R}^p$, se define una transformación de Householder como

$$\boldsymbol{H} = \boldsymbol{I} - \frac{2}{\boldsymbol{u}^T\boldsymbol{u}}\boldsymbol{u}\boldsymbol{u}^T = \boldsymbol{I} - \frac{2}{\|\boldsymbol{u}\|^2}\boldsymbol{u}\boldsymbol{u}^T. \tag{A.8}$$

Fácilmente comprobamos que es una matriz simétrica e idempotente.

Tomamos $\boldsymbol{x}, \boldsymbol{y} \in \mathbb{R}^p$ y tales que tienen la misma longitud, $\|\boldsymbol{x}\| = \|\boldsymbol{y}\|$ y definimos $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{y}$. Si $\boldsymbol{H}$ es la matriz de Householder correspondiente al vector $\boldsymbol{u}$ se verifica que $\boldsymbol{H}\boldsymbol{x} = \boldsymbol{y}$. Para probar esta igualdad notemos que

$$\begin{aligned}2\boldsymbol{u}^T\boldsymbol{x} = 2(\boldsymbol{x}-\boldsymbol{y})^T\boldsymbol{x} = 2(\boldsymbol{x}^T\boldsymbol{x} - \boldsymbol{y}^T\boldsymbol{x}) = \\ 2\|\boldsymbol{x}\|^2 - 2\boldsymbol{y}^T\boldsymbol{x} = \|\boldsymbol{x}\|^2 - 2\boldsymbol{y}^T\boldsymbol{x} + \|\boldsymbol{y}\|^2 = \\ (\boldsymbol{x}-\boldsymbol{y})^T(\boldsymbol{x}-\boldsymbol{y}) = \|\boldsymbol{x}-\boldsymbol{y}\|^2.\end{aligned} \tag{A.9}$$

Utilizando las igualdades (A.9) es inmediato

$$\begin{aligned}\boldsymbol{H}\boldsymbol{x} = \left(\boldsymbol{I} - \frac{2}{\|\boldsymbol{x}-\boldsymbol{y}\|^2}(\boldsymbol{x}-\boldsymbol{y})(\boldsymbol{x}-\boldsymbol{y})^T\right)\boldsymbol{x} = \\ \boldsymbol{x} - (\boldsymbol{x}-\boldsymbol{y})\frac{2(\boldsymbol{x}-\boldsymbol{y})^T\boldsymbol{x}}{\|\boldsymbol{x}-\boldsymbol{y}\|^2} = \boldsymbol{y}.\end{aligned} \tag{A.10}$$

Si $\boldsymbol{y} = (\|\boldsymbol{x}\|, 0, \ldots, 0)^T$ entonces para la matriz de Householder correspondiente a $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{y}$ tendríamos que $\boldsymbol{H}\boldsymbol{x} = (\|\boldsymbol{x}\|, 0, \ldots, 0)^T$. En definitiva, la transformación de Householder anula las componentes de $\boldsymbol{x}$ salvo la primera. Observermos que la primera componente de $\boldsymbol{u} = \boldsymbol{x} - \boldsymbol{y}$ es $x_1 - \|\boldsymbol{x}\|$. Si $x_1 < 0$ entonces $x_1 - \|\boldsymbol{x}\|$ no pierde dígitos significativos. Sin embargo, si $x_1 > 0$ y el valor de esta componente es próximo a $\|\boldsymbol{x}\|$ estamos perdiente precisión al hacer la diferencia $x_1 - \|\boldsymbol{x}\|$. Es habitual tomar

$$u_1 = x_1 - \|\boldsymbol{x}\| = \frac{x_1^2 - \|\boldsymbol{x}\|^2}{x_1 + \|\boldsymbol{x}\|}$$

de forma que

$$\frac{2}{\boldsymbol{u}^T\boldsymbol{u}} = \frac{2}{u_1^2 + x_2^2 + \ldots + x_p^2}.$$

Se tiene que si $\boldsymbol{x} \in \mathbb{R}^p$ y $\boldsymbol{H}$ es la matriz de Householder correspondiente al vector $\boldsymbol{u}$ entonces

$$\boldsymbol{H}\boldsymbol{x} = \left(I - \frac{2}{\boldsymbol{u}^T\boldsymbol{u}}\boldsymbol{u}\boldsymbol{u}^T\right)\boldsymbol{x} = \boldsymbol{x} - 2\frac{\boldsymbol{u}^T\boldsymbol{x}}{\boldsymbol{u}^T\boldsymbol{u}}\boldsymbol{u}. \tag{A.11}$$

Supongamos $\boldsymbol{A} \in \mathbb{R}^{n\times p}$ de rango p, $\boldsymbol{A} = [\boldsymbol{a}_1|\ldots|\boldsymbol{a}_p]$ (siendo $\boldsymbol{a}_j$ las columnas de la matriz $\boldsymbol{A}$. Consideramos $\boldsymbol{H}_1$ la matriz de Householder que anula todas las componentes de $\boldsymbol{a}_1$ excepto la primera. Tendremos

$$\boldsymbol{A} = \begin{bmatrix} r_{11} & r_{12} & \ldots & r_{1p} \\ 0 & & & \\ 0 & & \boldsymbol{A}_1 & \\ 0 & & & \end{bmatrix}$$

siendo $r_{11} = \|\boldsymbol{a}_1\|$ que no es nulo pues de lo contrario el rango de $\boldsymbol{A}$ sería menor a p. Tomamos

$$\boldsymbol{H}_2 = \begin{bmatrix} 1 & 0 & \ldots & 0 \\ 0 & & & \\ 0 & & \boldsymbol{K}_2 & \\ 0 & & & \end{bmatrix}$$

con $\boldsymbol{K}_2$ la matriz de Householder que anula todas las componentes de la primera columan de $\boldsymbol{A}_1$ salvo la primera. Se tiene

$$\boldsymbol{H}_2\boldsymbol{H}_1\boldsymbol{A} = \begin{bmatrix} r_{11} & r_{12} & \ldots & r_{1p} \\ 0 & r_{22} & \ldots & r_{2p} \\ 0 & 0 & & \\ 0 & & \boldsymbol{A}_2 & \\ 0 & 0 & & \end{bmatrix}$$

Además $r_{22} \neq 0$ pues de lo contrario la segunda columna de $\boldsymbol{H}_2\boldsymbol{H}_1\boldsymbol{A}$ depende linealmente de la primera pero $rank(\boldsymbol{H}_2\boldsymbol{H}_1\boldsymbol{A}) = rank(\boldsymbol{H}_1\boldsymbol{A}) = rank(\boldsymbol{A}) = p$ ya que las matrices $\boldsymbol{H}_i$ (i=1,2) son ortogonales. Aplicando el procedimiento anterior p veces obtenemos

$$\boldsymbol{H}_p \ldots \boldsymbol{H}_1\boldsymbol{A} = \begin{bmatrix} \boldsymbol{R} \\ \boldsymbol{0}_{(n-p)\times p} \end{bmatrix}$$

siendo $\boldsymbol{R}$ una matriz triangular superior $p \times p$ con todos los elementos de la diagonal principal positivos. Si definimos

$$\boldsymbol{Q} = (\boldsymbol{H}_p \ldots \boldsymbol{H}_1)^T = \boldsymbol{H}_1 \ldots \boldsymbol{H}_p, \tag{A.12}$$

entonces tenemos la descomposición QR de la matriz $\boldsymbol{A}$ dada r

$$\boldsymbol{A} = \boldsymbol{Q}\begin{bmatrix} \boldsymbol{R}_{p\times p} \\ \boldsymbol{0}_{(n-p)\times p} \end{bmatrix}$$

La matriz $\boldsymbol{Q}$ es ortogonal por ser producto de ortogonales. Podemos descomponer la matriz $\boldsymbol{Q}$ en dos submatrices

$$\boldsymbol{Q} = \begin{bmatrix} \boldsymbol{Q}_n \times p & \boldsymbol{Q}_{p\times(n-p)} \end{bmatrix}$$

A.13 Descomposición de Cholesky

Consideramos $\boldsymbol{A}$ una matriz definida positiva ($\boldsymbol{x}^T\boldsymbol{A}\boldsymbol{x} > 0$ si $\boldsymbol{x} \neq \boldsymbol{0}$. Una matriz definida positiva es simétrica y tienes sus valores propios extrictamente positivos. La descomposición de Cholesky nos dice que existe una matriz triangular superior $\boldsymbol{R}$ de las mismas dimensiones de $\boldsymbol{A}$ tal que

$$\boldsymbol{A} = \boldsymbol{R}^T\boldsymbol{R}.$$

Veamos cómo construir la descomposición de Cholesky. Utilizamos la convención $\sum_{k=1}^{0} x_i = 0$. Tendremos

$$R_{ii} = \sqrt{D_{ii} - \sum_{k=1}^{i-1}} \; y \; R_{ij}\frac{D_{ii} - \sum_{k=1}^{i-1} R_{ki}R_{kj}}{R_{ii}}, \; j > i.$$

Si tenemos sistemas de ecuaciones en las que intervienen matrices definidas positivas el uso de la descomposición de Cholesky es muy adecuda ya que si el sistema es $\boldsymbol{A} = \boldsymbol{y}$ siendo $\boldsymbol{A}$ definida positiva entonces trendremos

$$\boldsymbol{R}^T\boldsymbol{R} = y,$$

Teniendo en cuenta que $\boldsymbol{R}^T$ es triangular superior podemos resolver con facilidad la solución $\boldsymbol{R}\boldsymbol{x}$ y con un paso adicional podemos obtener $\boldsymbol{x}$.

A.14 Ejercicios

Ex. 38 — Demostrar que la matriz de Householder definida en A.8 es simétrica e idempotente.

Apéndice B

Algo de Probabilidad

En este tema se consideran resultados probabilísticos no tratados habitualmente en un curso básico de Probabilidad que es lo asumido para este curso.

B.1 Función generatriz de momentos

B.1.1 Variable aleatoria

Definición B.1 (Función generatriz de momentos)**.** *Si X es una variable aleatoria entonces definimos la función generatriz de momentos como*

$$M(t) = E\left[e^{tX}\right].$$

Notemos que $M(0) = 1$. Además, asumiendo condiciones que permitan el intercambio de la derivada con la integral, se tiene

$$\frac{dM(t)}{dt} = E[Xe^{tX}].$$

Por tanto:

$$\left.\frac{dM(t)}{dt}\right|_{t=0} = E[X].$$

Aplicando el mismo procedimiento recursivamente tendremos que

$$\left.\frac{dM(t)}{dt^k}\right|_{t=0} = E[X^k],$$

para $k = 1, 2, \ldots$ asumiendo que existen las correspondientes derivadas. De hecho, asumiendo la existencia de la función generatriz de momentos en un intervalo abierto que contenga al origen, se puede probar que la función generatriz de momentos caracteriza la distribución de probabilidad.

B.1.2 Vector aleatorio

Si suponemos $\boldsymbol{X}$ un vector aleatorio n-dimensional entonces definimos la función generatriz de momentos del vector como

Definición B.2 (Función generatriz de momentos)**.** *Si X es una variable aleatoria entonces definimos la función generatriz de momentos como*

$$M(t) = E\left[e^{\boldsymbol{t}^T \boldsymbol{X}}\right] = E\left[e^{\sum_{i=1}^{n} t_i X_i}\right]$$

con $\boldsymbol{t} \in \mathbb{R}^n$.

B.2 Función característica

No siempre existe la función generatriz de momentos. La *función característica* sí que está siempre definida y es una herramienta de gran utilidad en Probabilidad. Caracteriza la distribución de la variable aleatoria y nos permite obtener con facilidad la distribución de probabilidad de sumas de variables aleatorias y la distribución del límite de sucesiones de variables aleatorias.

El concepto de *función característica* fue introducido por Lyapunov en una de la primeras versiones del Teorema Central del Límite.

Definición B.3. *Sea X una variable aleatoria y sea $t \in \mathbb{R}$. La función característica de X, $\phi_X(t)$, se define como*

$$E\left[e^{itX}\right],$$

y puesto que $|e^{itX}| \leq 1$, $\forall t$, $\phi_X(t)$ existe siempre y está definida $\forall t \in \mathbb{R}$.

Podemos ver su expresión para variables discretas y continuas. Si X es una variable aleatoria discreta con soporte D_X y función de probabilidad $f_X(x)$ entonces

$$\phi_X(t) = \sum_{x \in D_X} e^{itx} f_X(x) = \sum_{x \in D_X} e^{itx} P(X = x). \tag{B.1}$$

Si X es una variable aleatoria continua con función de densidad de probabilidad $f_X(x)$ entonces la función característica sería

$$\phi_X(t) = \int_{-\infty}^{+\infty} e^{itx} f_X(x) dx = \int_{-\infty}^{+\infty} \cos(tx) f_X(x) dx + i \int_{-\infty}^{+\infty} \sin(tx) f_X(x) dx \tag{B.2}$$

De la definición se derivan las siguientes propiedades.

Proposición B.1. *1. $\phi_X(0) = 1$*

2. *$|\phi_X(t)| \leq 1$*
3. *$\phi_X(t)$ es uniformemente continua.*
4. *Si definimos $Y = aX + b$,*

$$\phi_Y(t) = E(e^{itY}) = E\left(e^{it(aX+b)}\right) = e^{itb}\phi_X(at)$$

5. *Si $E(X^n)$ existe, la función característica es n veces diferenciable y $\forall k \leq n$ se verifica $\phi_X^{(k)}(0) = i^k E(X^k)$*

Prueba. Veamos el apartado 3.

$$\phi_X(t+h) - \phi_X(t) = \int_\Omega e^{itX}(e^{ihX} - 1)\, dP.$$

Al tomar módulos,

$$|\phi_X(t+h) - \phi_X(t)| \leq \int_\Omega |e^{ihX} - 1|\, dP, \tag{B.3}$$

pero $|e^{ihX} - 1| \leq 2$ y (B.3) será finito, lo que permite intercambiar integración y paso al límite, obteniendo

$$\lim_{h\to 0} |\phi_X(t+h) - \phi_X(t)| \leq \int_\Omega \lim_{h\to 0} |e^{ihX} - 1|\, dP = 0.$$

La propiedad 5 establece un interesante relación entre las derivadas de $\phi_X(t)$ y los momentos de X cuando estos existen, relación que permite desarrollar $\phi_X(t)$ en serie de potencias. En efecto, si $E(X^n)$ existe $\forall n$, entonces,

$$\phi_X(t) = \sum_{k\geq 0} \frac{i^k E(X^k)}{k!} t^k. \tag{B.4}$$

□

Una de las propiedades fundamentales de la función característica tiene que ver con la distribución de la suma de variables aleatorias independientes.

Proposición B.2. *Si $X_1, X_2, \ldots, X_n$ son variables aleatorias independientes y definimos $Y = X_1 + X_2 + \cdots + X_n$ tenemos que*

$$\phi_Y(t) = E\left(e^{it(X_1+X_2+\cdots+X_n)}\right) = E\left(\prod_{k=1}^n e^{itX_k}\right) = \prod_{k=1}^n E\left(e^{itX_k}\right) = \prod_{k=1}^n \phi_{X_k}(t). \tag{B.5}$$

Es interesante ver cómo son las funciones características de algunas distribuciones importantes.

Ejemplo B.1 (Bernoulli)**.** *Si $X \sim B(1,p)$*

$$\phi_X(t) = e^0 q + e^{it} p = q + pe^{it}.$$

Ejemplo B.2 (Binomial)**.** *Si $X \sim B(n,p)$*

$$\phi_X(t) = \prod_{k=1}^n (q + pe^{it}) = (q + pe^{it})^n.$$

Ejemplo B.3 (Poisson)**.** *Si $X \sim P(\lambda)$*

$$\phi_X(t) = \sum_{x\geq 0} e^{itx} \frac{e^{-\lambda}\lambda^x}{x!} = e^{-\lambda} \sum_{x\geq 0} \frac{(\lambda e^{it})^x}{x!} = e^{\lambda(e^{it}-1)}.$$

Ejemplo B.4 (Normal)**.** *Si $Z \sim N(0,1)$, sabemos que existen los momentos de cualquier orden y en particular, $E(Z^{2n+1}) = 0$, $\forall n$ y $E(Z^{2n}) = \frac{(2n)!}{2^n n!}$, $\forall n$.*

$$\phi_Z(t) = \sum_{n\geq 0} \frac{i^{2n}(2n)!}{2^n(2n)!n!} t^{2n} = \sum_{n\geq 0} \frac{\left(\frac{(it)^2}{2}\right)^n}{n!} = \sum_{n\geq 0} \frac{\left(-\frac{t^2}{2}\right)^n}{n!} = e^{-\frac{t^2}{2}}.$$

Para obtener $\phi_X(t)$ si $X \sim N(\mu, \sigma^2)$, podemos utilizar el resultado anterior. En efecto, recordemos que X puede expresarse en función de Z mediante $X = \mu + \sigma Z$,

$$\phi_X(t) = e^{i\mu t}\phi_Z(\sigma t) = e^{i\mu t - \frac{\sigma^2 t^2}{2}}.$$

Notemos que $Im(\phi_Z(t)) = 0$. Esto lo verifica cualquier variable aleatoria que tenga distribución simétrica, esto es, que X y $-X$ tengan la misma distribución de probabilidad.

Ejemplo B.5 (Gamma)**.** *Si $X \sim G(\alpha, \beta)$, su función de densidad de probabilidad viene dada por*

$$f_X(x) = \begin{cases} \dfrac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x} & \text{si } x > 0, \\ \\ 0 & \text{si } x \leq 0, \end{cases}$$

por lo que

$$\phi_X(t) = \frac{\beta^\alpha}{\Gamma(\alpha)} \int_0^\infty e^{itx} x^{\alpha-1} e^{-\beta x} dx,$$

que con el cambio $y = x(1 - it/\beta)$ conduce a

$$\phi_X(t) = \left(1 - \frac{it}{\beta}\right)^{-\alpha}.$$

Ejemplo B.6 (Exponencial)**.** *La distribución exponencial puede ser considerada un caso particular de $G(\alpha, \beta)$ cuando $\alpha = 1$. A partir de aquí,*

$$\phi_X(t) = \frac{\beta}{\beta - it}.$$

Ejemplo B.7 (Ji-cuadrado)**.** *Si consideramos una distribución gamma con $\alpha = n/2$ y $\beta = 1/2$, decimos que X tiene una distribución χ^2 con n grados de libertad y se denota $X \sim \chi^2_n$. Su función característica será*

$$\phi_X(t) = (1 - 2it)^{-\frac{n}{2}}.$$

B.3 Vectores y matrices aleatorias

[112] [79], página 5 y siguientes.

[112] Consideremos una colección de variables aleatorias Z_{ij} con $i = 1, \ldots, m$ y $j = 1, \ldots, n$. Tenemos la matriz aleatoria $\boldsymbol{Z} = [Z_{ij}]$.

Definición B.4. $E[\boldsymbol{Z}] = [EZ_{ij}]$

Teorema B.1. *Si $\boldsymbol{A} \in \mathbb{R}^{l\times m}$, $\boldsymbol{B} \in \mathbb{R}^{n\times p}$ y $\boldsymbol{C} \in \mathbb{R}^{l\times p}$ entonces*

$$E[\boldsymbol{AZB} + \boldsymbol{C}] = \boldsymbol{A}E[\boldsymbol{Z}]\boldsymbol{B} + \boldsymbol{C}.$$

La prueba es consecuencia inmediata de la linealidad de la media. Si $\boldsymbol{X}$ and $\boldsymbol{Y}$ son $n \times 1$ vectores aleatorios y $\boldsymbol{A}, \boldsymbol{B} \in \mathbb{R}^{m\times n}$ entonces aplicando linealidad de la esperanza se tiene que

$$E[\boldsymbol{AX} + \boldsymbol{BY}] = \boldsymbol{A}E[\boldsymbol{X}] + \boldsymbol{B}E[\boldsymbol{Y}]. \tag{B.6}$$

Definición B.5. *Si $\boldsymbol{X}$ and $\boldsymbol{Y}$ son $m \times 1$ y $n \times 1$ vectores aleatorios podemos definir. $cov[\boldsymbol{X}, \boldsymbol{Y}] = [cov(X_i, Y_j)]$.*

Teorema B.2. *Si $\boldsymbol{\mu_X} = E[\boldsymbol{X}]$ y $\boldsymbol{\mu_Y} = E[\boldsymbol{Y}]$ entonces*

$$cov[\boldsymbol{X}, \boldsymbol{Y}] = E[(\boldsymbol{X} - \mu_{\boldsymbol{X}})(\boldsymbol{Y} - \mu_{\boldsymbol{Y}})^T].$$

Prueba. Simplemente en la posición (i, j) de $cov[\boldsymbol{X}, \boldsymbol{Y}]$ tenemos $E[(X_i - \mu_{X_i})(Y_j - \mu_{Y_j})]$ que coincide con el elemento en la misma posición del $(\boldsymbol{X} - \mu_{\boldsymbol{X}})(\boldsymbol{Y} - \mu_{\boldsymbol{Y}})^T$. □

Definición B.6. $var(\boldsymbol{X}) = cov[\boldsymbol{X}, \boldsymbol{X}]$.

La matriz de varianzas también es habitualmente denotada com $V_{\boldsymbol{X}}$ o $\Sigma_{\boldsymbol{X}}$. A la matriz $var(\boldsymbol{X})$ se le denomina matriz de varianzas, matriz de covarianzas, matriz de varianzas-covarianzas o matriz de dispersión. Notemos que $var(\boldsymbol{X})$ es simétrica y tiene en la diagonal principal las varianzas de las variables. Tenemos que

$$var(\boldsymbol{X}) = E[(\boldsymbol{X} - \mu_{\boldsymbol{X}})(\boldsymbol{X} - \mu_{\boldsymbol{X}})^T]. \tag{B.7}$$

Fácilmente se comprueba que

$$var(\boldsymbol{X}) = E[(\boldsymbol{X}\boldsymbol{X}^T)] - \mu_{\boldsymbol{X}}\mu_{\boldsymbol{X}}^T. \tag{B.8}$$

Corolario B.1. *Si $\boldsymbol{a} \in \mathbb{R}^n$ entonces $var[\boldsymbol{X} - \boldsymbol{a}] = var[\boldsymbol{X}]$.*

La prueba es inmediata ya que $X_i - a_i - E[X_i - a_i] = X_i - E[X_i]$.

Teorema B.3. *Si $\boldsymbol{X}$ es un vector aleatorio sobre $\mathbb{R}^m$ e $\boldsymbol{Y}$ es un vector aleatorio sobre $\mathbb{R}^n$ y Si $\boldsymbol{A} \in \mathbb{R}^{l\times m}$, $\boldsymbol{B} \in \mathbb{R}^{p\times n}$ entonces*

$$cov(\boldsymbol{AX}, \boldsymbol{BY}) = \boldsymbol{A}cov(\boldsymbol{X}, \boldsymbol{Y})\boldsymbol{B}^T.$$

Prueba.

$$\begin{aligned} cov(\boldsymbol{AX}, \boldsymbol{BY}) &= E[(\boldsymbol{AX} - \boldsymbol{A}\mu_{\boldsymbol{X}})(\boldsymbol{BY} - \boldsymbol{B}\mu_{\boldsymbol{Y}})^T] = \\ E[\boldsymbol{A}(\boldsymbol{X} - \mu_{\boldsymbol{X}})(\boldsymbol{Y} - \mu_{\boldsymbol{Y}})^T\boldsymbol{B}^T] &= \boldsymbol{A}E[(\boldsymbol{X} - \mu_{\boldsymbol{X}})(\boldsymbol{Y} - \mu_{\boldsymbol{Y}})^T]\boldsymbol{B}^T = \\ &\boldsymbol{A}cov(\boldsymbol{X}, \boldsymbol{Y})\boldsymbol{B}^T. \end{aligned} \tag{B.9}$$

□

Y en particular se verifica que

Corolario B.2. *1.* $cov(\boldsymbol{AX}, \boldsymbol{Y}) = \boldsymbol{A}cov(\boldsymbol{X}, \boldsymbol{Y})$.

2. $cov(\boldsymbol{X}, \boldsymbol{BY}) = cov(\boldsymbol{X}, \boldsymbol{Y})\boldsymbol{B}^T$.

3. $var[\boldsymbol{AX}] = \boldsymbol{A}var[\boldsymbol{X}]\boldsymbol{A}^T$.

Teorema B.4. *Sea $\boldsymbol{X}$ es un vector aleatorio que verifica que no existen $\boldsymbol{a}(\neq \boldsymbol{0})$ y b tales $\boldsymbol{a}^T\boldsymbol{X} = b$ con probabilidad uno. Entonces $var[\boldsymbol{X}]$ es una matriz definida positiva.*

Prueba. Tenemos para cualquier $\boldsymbol{c}$ que $0 \leq var[\boldsymbol{c}^T\boldsymbol{X}] = \boldsymbol{c}^T var[\boldsymbol{X}]\boldsymbol{c}$. La igualdad a cero de la varianza supone que la variable $\boldsymbol{c}^T\boldsymbol{X}$ es constante con probabilidad uno: $\boldsymbol{c}^T\boldsymbol{X} = d$ (a.s.) Por las hipótesis del teorema se sigue que $\boldsymbol{c} = \boldsymbol{0}$ y la matriz $var[\boldsymbol{X}]$ es definida positiva. □

Teorema B.5. *Sea $\boldsymbol{X}$ es un vector aleatorio (en $\mathbb{R}^n$) y $\boldsymbol{A} \in \mathbb{R}^{n\times n}$ simétrica. Si denotamos $E[\boldsymbol{X}] = \mu$ entonces*

$$E[\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X}] = tr(\boldsymbol{A}\boldsymbol{v}ar(X)) + \mu^T\boldsymbol{A}\mu.$$

[113] tr denota la traza de la matriz.

[113]

Prueba.

$$\begin{aligned} E[\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X}] = tr(E[\boldsymbol{X}^T\boldsymbol{A}\boldsymbol{X}]) = E[tr(\boldsymbol{A}\boldsymbol{X}\boldsymbol{X}^T)] = \\ tr(E[\boldsymbol{A}\boldsymbol{X}\boldsymbol{X}^T]) = tr(\boldsymbol{A}E[\boldsymbol{X}\boldsymbol{X}^T]) = \\ tr(\boldsymbol{A}[var(\boldsymbol{X}) + \boldsymbol{\mu}\boldsymbol{\mu}^T]) = tr(\boldsymbol{A}var(\boldsymbol{X})) + tr(\boldsymbol{A}\boldsymbol{\mu}\boldsymbol{\mu}^T) = \\ tr(\boldsymbol{A}var(\boldsymbol{X})) + tr(\boldsymbol{\mu}^T\boldsymbol{A}\boldsymbol{\mu}). \quad \text{(B.10)} \end{aligned}$$

□

Tenemos dos casos de interés. Si tomamos $\boldsymbol{Y} = \boldsymbol{X} - \boldsymbol{b}$ y tenemos en cuenta que $var(\boldsymbol{Y}) = var(\boldsymbol{X})$ entonces

$$E[(\boldsymbol{X} - \boldsymbol{b})^T\boldsymbol{A}(\boldsymbol{X} - \boldsymbol{b})] = tr(\boldsymbol{A}\boldsymbol{v}ar(X)) + (\boldsymbol{\mu} - \boldsymbol{b})^T\boldsymbol{A}(\boldsymbol{\mu} - \boldsymbol{b}). \quad \text{(B.11)}$$

Si tenemos que $var(X) = \sigma^2\boldsymbol{I}_n$ entonces $tr(\boldsymbol{A}\boldsymbol{v}ar(X)) = \sigma^2 tr(\boldsymbol{A})$.

B.4 Distribución normal multivariante

Definición B.7. *Se dice que el vector $\boldsymbol{Y} = (Y_1, Y_2, \ldots, Y_n)^T$ tiene una distribución normal n-variante cuando su densidad se puede expresar como*

$$f(\boldsymbol{y}) = \frac{1}{(2\pi)^{\frac{n}{2}} det(\boldsymbol{\Sigma})^{\frac{1}{2}}} e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{y}-\boldsymbol{\mu})} \quad \text{(B.12)}$$

siendo $\boldsymbol{y} = (y_1, \ldots, y_n)^T$, $\boldsymbol{\mu} = (\mu_1, \ldots, \mu_n)^T \in \mathbb{R}^n$ y $\boldsymbol{\Sigma} = [\sigma_{ij}]_{i,j=1,\ldots,n} \in \mathbb{R}^{n\times n}$ definida positiva.

La matriz $\boldsymbol{\Sigma}$ es definida positiva por lo que existe una matriz cuadrada definida positiva que es su raiz cuadrada (proposición A.30), esto es, verificando $\boldsymbol{\Sigma}^{1/2}\boldsymbol{\Sigma}^{1/2} = (\boldsymbol{\Sigma}^{1/2})^2 = \boldsymbol{\Sigma}$.

Teorema B.6. *La función definida en (B.12) es una función de densidad de probabilidad.*

Prueba. Trivialmente es no negativa. Veamos que integra uno. Consideremos la transformación $\boldsymbol{z} = \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{y} - \boldsymbol{\mu})$ con la transformación inversa dada por $\boldsymbol{y} = \boldsymbol{\Sigma}^{1/2}\boldsymbol{z} + \boldsymbol{\mu}$. El jacobiano de la transformación inversa es $det(\boldsymbol{\Sigma}^{1/2}) = det(\boldsymbol{\Sigma})^{1/2}$. Aplicando el cambio de variable tenemos

$$\begin{aligned} \int_{\mathbb{R}^n} e^{-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\mathbf{y}-\boldsymbol{\mu})} d\boldsymbol{y} = \\ \int_{\mathbb{R}^n} e^{-\frac{1}{2}\boldsymbol{z}^T\boldsymbol{\Sigma}^{1/2}\boldsymbol{\Sigma}^{-1}\boldsymbol{\Sigma}^{1/2}\boldsymbol{z}} det(\boldsymbol{\Sigma})^{1/2} d\boldsymbol{z} = \\ \int_{\mathbb{R}^n} e^{-\frac{1}{2}\boldsymbol{z}^T\boldsymbol{z}} det(\boldsymbol{\Sigma})^{1/2} d\boldsymbol{z} = det(\boldsymbol{\Sigma})^{1/2} \prod_{i=1}^{n} (2\pi)^{1/2}. \quad \text{(B.13)} \end{aligned}$$

□

Teorema B.7. *Si* $\boldsymbol{Y}$ *tiene la densidad dada en* (B.12) *entonces* $E[\boldsymbol{Y}] = \boldsymbol{\mu}$ *y* $var(\boldsymbol{Y}) = \boldsymbol{\Sigma}$.

Prueba. Aplicando otra vez el cambio de variable $\boldsymbol{z} = \boldsymbol{\Sigma}^{-1/2}(\boldsymbol{y} - \boldsymbol{\mu})$ tenemos como densidad conjunta

$$f(z_1, \ldots, z_n) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}z_i^2}.$$

En resumen, $\boldsymbol{Z}$ tiene por componentes variables normales independientes y con distribución normal estándar. Por tanto $E[\boldsymbol{Z}] = \boldsymbol{0}$ y $var[\boldsymbol{Z}] = \boldsymbol{I}_n$. Pero

$$E[\boldsymbol{Y}] = E[\boldsymbol{\Sigma}^{1/2}\boldsymbol{Z} + \boldsymbol{\mu}] = \boldsymbol{\Sigma}^{1/2}E[\boldsymbol{Z}] + \boldsymbol{\mu} = \boldsymbol{\mu}.$$

Por otra parte

$$var[\boldsymbol{Y}] = var[\boldsymbol{\Sigma}^{1/2}\boldsymbol{Z} + \boldsymbol{\mu}] = var[\boldsymbol{\Sigma}^{1/2}\boldsymbol{Z}] = \boldsymbol{\Sigma}^{1/2}var[\boldsymbol{Z}]\boldsymbol{\Sigma}^{1/2} = \boldsymbol{\Sigma}^{1/2}\boldsymbol{I}_n\boldsymbol{\Sigma}^{1/2} = \boldsymbol{\Sigma}. \quad \text{(B.14)}$$

□

Por ello denotamos, de un modo análogo al caso unidimensional, que $\boldsymbol{Y} \sim N_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$.

Ejemplo B.8 (Componentes independientes)**.** *Cuando las componentes del vector son independientes, las covarianzas son todas nulas y* $\boldsymbol{\Sigma}$ *es una matriz diagonal cuyos elementos son las varianzas de cada componente:* $\boldsymbol{\Sigma} = diag(\sigma_1^2, \ldots, \sigma_n^2)$*. Por tanto,*

$$|\boldsymbol{\Sigma}|^{-1} = \frac{1}{\sigma_1^2\sigma_2^2\cdots\sigma_n^2}.$$

Además, la forma cuadrática que aparece en el exponente de (B.12) *se simplifica y la densidad sería*

$$f(y_1, y_2, \ldots, y_n) = \frac{1}{\prod_{i=1}^{n}(2\pi\sigma_i^2)^{\frac{1}{2}}} e^{-\frac{1}{2}\sum_{i=1}^{n}\left(\frac{y_i-\mu_i}{\sigma_i}\right)^2} = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma_i^2}} e^{-\frac{1}{2}\left(\frac{y_i-\mu_i}{\sigma_i}\right)^2}, \quad \text{(B.15)}$$

que no es más que el producto de las densidades marginales de las componentes.

Si $\boldsymbol{Z} \sim N_n(\boldsymbol{0}, \boldsymbol{I}_n)$ entonces su función generatriz de momentos es

$$Ee^{\boldsymbol{t}^T\boldsymbol{Z}} = Ee^{\sum_{i=1}^{n} t_i Z_i} = E\prod_{i=1}^{n} e^{t_i Z_i} = \prod_{i=1}^{n} Ee^{t_i Z_i} = \prod_{i=1}^{n} e^{\frac{1}{2}t_i^2} = e^{\frac{1}{2}\boldsymbol{t}^T\boldsymbol{t}}. \quad \text{(B.16)}$$

Si consideramos $\boldsymbol{Y} \sim N_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ entonces $\boldsymbol{Y} = \boldsymbol{\Sigma}^{1/2}\boldsymbol{Z} + \boldsymbol{\mu}$ con $\boldsymbol{Z} \sim N_n(\boldsymbol{0}, \boldsymbol{I}_n)$. Utilizando la ecuación B.16 con $\boldsymbol{s} = \boldsymbol{\Sigma}^{1/2}\boldsymbol{t}$ tenemos

$$E[e^{\boldsymbol{t}^T\boldsymbol{Y}}] = E[e^{\boldsymbol{t}^T(\boldsymbol{\Sigma}^{1/2}\boldsymbol{Z}+\boldsymbol{\mu})}] = E[e^{\boldsymbol{s}^T\boldsymbol{Z}}]e^{\boldsymbol{t}^T\boldsymbol{\mu}} = E[e^{\frac{1}{2}\boldsymbol{t}^T\boldsymbol{\Sigma}^{1/2}\boldsymbol{\Sigma}^{1/2}\boldsymbol{t}+\boldsymbol{t}^T\boldsymbol{\mu}}] = e^{\boldsymbol{t}^T\boldsymbol{\mu}+\frac{1}{2}\boldsymbol{t}^T\boldsymbol{\Sigma}\boldsymbol{t}}. \quad \text{(B.17)}$$

Utilizando estas funciones generatrices podemos probar el siguiente teorema.

Teorema B.8. *Sea $\boldsymbol{Y} \sim N_n(\mu, \boldsymbol{\Sigma})$, $\boldsymbol{C} \in \mathbb{R}^{m\times n}$ de rango m, y $\boldsymbol{d} \in \mathbb{R}^m$. Entonces $\boldsymbol{C}\boldsymbol{Y} + \boldsymbol{d} \sim N_m(\boldsymbol{C}\boldsymbol{\mu} + \boldsymbol{d}, \boldsymbol{C}\boldsymbol{\Sigma}\boldsymbol{C}^T)$.*

Prueba. Determinemos la función generatriz de momentos de $\boldsymbol{C}\boldsymbol{Y}+\boldsymbol{d}$.

$$E[e^{\boldsymbol{t}^T(\boldsymbol{C}\boldsymbol{Y}+\boldsymbol{d})}] = E[e^{(\boldsymbol{C}^T\boldsymbol{t})^T\boldsymbol{Y}+\boldsymbol{t}^T\boldsymbol{d}}] = E[e^{(\boldsymbol{C}^T\boldsymbol{t})^T\boldsymbol{\mu}+\frac{1}{2}(\boldsymbol{C}^T\boldsymbol{t})^T\boldsymbol{\Sigma}\boldsymbol{C}^T\boldsymbol{t}+\boldsymbol{t}^T\boldsymbol{d}}] = E[e^{\boldsymbol{t}^T(\boldsymbol{C}\boldsymbol{\mu}+\boldsymbol{d})+\frac{1}{2}\boldsymbol{t}^T\boldsymbol{C}\boldsymbol{\Sigma}\boldsymbol{C}^T\boldsymbol{t}}]. \quad \text{(B.18)}$$

La matriz $\boldsymbol{C}\boldsymbol{\Sigma}\boldsymbol{C}^T$ es definida positiva (proposición A.25). La función que acabamos de obtener corresponde a la función generatriz de momentos de una $N_m(\boldsymbol{C}\boldsymbol{\mu} + \boldsymbol{d}, \boldsymbol{C}\boldsymbol{\Sigma}\boldsymbol{C}^T)$. Es importante notar que $\boldsymbol{C}$ ha de tener rango completo para que $\boldsymbol{C}\boldsymbol{\Sigma}\boldsymbol{C}^T$ sea definida positiva. □

Corolario B.3. *Si $\boldsymbol{Y} = \boldsymbol{A}\boldsymbol{Z} + \boldsymbol{\mu}$ con $\boldsymbol{A}$ no singular y $\boldsymbol{Z} \sim N_n(\boldsymbol{0}, \boldsymbol{I})$ entonces $\boldsymbol{Y} \sim N_n(\boldsymbol{\mu}, \boldsymbol{A}\boldsymbol{A}^T)$.*

Ejemplo B.9. *Supongamos $\boldsymbol{Y} \sim N_n(\boldsymbol{0}, \boldsymbol{I}_n)$ y $\boldsymbol{T}$ es una matriz ortogonal. Entonces por el teorema B.8, $\boldsymbol{Z} = \boldsymbol{T}^T\boldsymbol{Y} \sim N_n(\boldsymbol{0}, \boldsymbol{I}_n))$ ya que $\boldsymbol{T}^T\boldsymbol{T} = I_n$.*

En esta sección nos hemos ocupado de las propiedades básicas que necesitamos de la distribución normal multivariante. Es importante poder trabajar a nivel práctico con esta importante distribución. Un paquete de R imprescindible es [49, mvtnorm].

B.5 Distribución de las formas cuadráticas

[114] [79], pág. 27 y siguientes.

[114] Suponemos $\boldsymbol{Y} \sim N_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ siendo $\boldsymbol{\Sigma}$ definida positiva. En esta sección se estudia la distribución de la forma cuadrática

$$\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y} = \sum_{i=1}^{n}\sum_{j=1}^{n} a_{ij}Y_iY_j.$$

Podemos asumir simétrica la matriz $\boldsymbol{A}$ ya que si no lo fuera, sustituimos a_{ij}, sin que cambie la forma cuadrática, por $(a_{ij} + a_{ji})/2$. Por tanto, $\boldsymbol{A}$ es simétrica. Aplicando el teorema de descomposición espectral tenemos una matriz ortogonal y una matriz diagonal tales que $\boldsymbol{A} = \boldsymbol{T}^T\boldsymbol{\Lambda}\boldsymbol{T}$ con $\boldsymbol{\Lambda} = diag(\lambda_1, \ldots, \lambda_n)$ donde los λ_i son los valores propios (reales) de $\boldsymbol{A}$. Tenemos que

$$\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y} = \boldsymbol{Y}^T\boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T\boldsymbol{Y} = \boldsymbol{Z}^T\boldsymbol{\Lambda}\boldsymbol{Z} = \sum_{i=1}^{n}\lambda_i Z_i^2.$$

Si $\boldsymbol{Y} \sim N_n(\boldsymbol{0}_n, \boldsymbol{I}_n)$ entonces $\boldsymbol{Z} = \boldsymbol{T}^T\boldsymbol{Y} \sim N_n(\boldsymbol{0}_n, \boldsymbol{I}_n)$ y la forma cuadrática $\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y}$ es una combinación lineal de variables independientes con distribución χ^2_1. Si r de los valores propios son 1 y los demás $n-r$ son nulos entonces la distribución será χ^2_r. ¿Cuándo ocurre esto?

Teorema B.9. *Sea $\boldsymbol{A}$ una matriz simétrica. Entonces $\boldsymbol{A}$ tiene r valores propios iguales a 1 y los demás $n-r$ iguales a 0 si y solo si $\boldsymbol{A}^2 = \boldsymbol{A}$ y el rango de $\boldsymbol{A}$ es igual a r.*

Prueba. Supongamos que $\boldsymbol{A}$ tiene r valores propios iguales a 1 y los demás $n-r$ iguales a 0. Realizamos la descomposición espectral de $\boldsymbol{A}$, $\boldsymbol{A} = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T$. Pero Λ es una matriz diagonal con los valores propios en su diagonal. Podemos suponer que son los r primeros son iguales a 1 y los demás son cero: $\boldsymbol{\Lambda} = diag(\boldsymbol{1}_r, \boldsymbol{0}_{n-r})$. Tenemos $rank(\boldsymbol{A}) = rank(\boldsymbol{\Lambda}) = r$ y $\boldsymbol{A}^2 = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T\boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T = \boldsymbol{T}\boldsymbol{\Lambda}^2\boldsymbol{T}^T = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T$ y en la última igualdad usamos que los valores propios son 0 o 1.

Supongamos $\boldsymbol{A}^2 = \boldsymbol{A}$ y el rango de $\boldsymbol{A}$ es igual a r. Por ser $\boldsymbol{A}$ simétrica tenemos una matriz de proyección (proposición A.36). Pero una matriz de proyección tiene valores propios que son 0 o 1 (proposición A.37). La descomposición espectral de $\boldsymbol{A}$, $\boldsymbol{A} = \boldsymbol{T}\boldsymbol{\Lambda}\boldsymbol{T}^T$, nos indica que $rank(\boldsymbol{A}) = rank(\boldsymbol{\Lambda})$. Pero $rank(\boldsymbol{\Lambda})$ es el número de valores propios iguales a uno que han de ser r. Los demás son nulos. □

Tenemos pues el siguiente teorema.

Teorema B.10. *Sea $\boldsymbol{Y} \sim N_n(\boldsymbol{0}, \boldsymbol{I}_n)$ y $\boldsymbol{A}$ una matriz simétrica. Entonces $\boldsymbol{Y}^T A\boldsymbol{Y} \sim \chi^2_r$ si y solo si $\boldsymbol{A}$ es idempotente de rango r.*

Prueba. Por el teorema B.9 vemos que si la matriz $\boldsymbol{A}$ es simétrica, idempotente y con rango r entonces $\boldsymbol{Y}^T A\boldsymbol{Y} \sim \chi^2_r$.

Supongamos que $\boldsymbol{A}$ es simétrica y $\boldsymbol{Y}^T A\boldsymbol{Y} \sim \chi^2_r$. Utilizando el teorema B.9 solamente hemos de probar que la matriz $\boldsymbol{A}$ tiene r valores propios iguales a uno y los demás a cero. La función generatriz de momentos de $\boldsymbol{Y}^T A\boldsymbol{Y}$, teniendo en cuenta que $\boldsymbol{Y}^T A\boldsymbol{Y} = \sum_{i=1}^n \lambda_i Z_i^2$ con los Z_i i.i.d. con distribución normal estándar, es igual a $\prod_{i=1}^n (1-2\lambda_i t)^{-1/2}$. Pero asumimos $\boldsymbol{Y}^T A\boldsymbol{Y} \sim \chi^2_r$ por tanto

$$\prod_{i=1}^{n} (1-2\lambda_i t)^{-1/2} = (1-2t)^{-r/2},$$

esto es,

$$\prod_{i=1}^{n} (1-2\lambda_i t) = (1-2t)^{r}.$$

Por la unicidad de la factorización se ha de dar que r de los λ_i son iguales a uno y los demás a cero. □

Como una consecuencia podemos obtener un resultado muy conocido.

Teorema B.11. *Supongamos $Y_1, \ldots, Y_n$ i.i.d. $Y_i \sim N(\mu, \sigma^2)$ y $S^2 = \frac{1}{n-1}\sum_{i=1}^n (Y_i - \bar{Y}_n)^2$ entonces*

$$\frac{n-1}{\sigma^2} S^2 \sim \chi^2_{n-1}.$$

Prueba.

$$\frac{n-1}{\sigma^2} S^2 = \frac{1}{\sigma^2} \boldsymbol{Y}^T (\boldsymbol{I}_n - \frac{1}{n}\boldsymbol{J}_n)\boldsymbol{Y},$$

donde $\boldsymbol{J}_n = \boldsymbol{1}_n \boldsymbol{1}_n^T$. Tomamos $\boldsymbol{Z} = \frac{1}{\sigma}(\boldsymbol{Y} - \mu \boldsymbol{1}_n) \sim N_n(\boldsymbol{0}, \boldsymbol{I}_n)$. Tenemos

$$\frac{n-1}{\sigma^2} S^2 = \boldsymbol{Z}^T (\boldsymbol{I}_n - \frac{1}{n}\boldsymbol{J}_n)\boldsymbol{Z},$$

con $\boldsymbol{I}_n - \frac{1}{n}\boldsymbol{J}_n$ es simétrica e idempotente. Su rango coincide con su traza por A.38 y la traza es igual a

$$tr(\boldsymbol{I}_n - \frac{1}{n}\boldsymbol{J}_n) = n - 1.$$

Utilizando el teorema B.10 se sigue el resultado. □

Corolario B.4. *Sea* $\boldsymbol{Y} \sim N_n(\boldsymbol{0}, \boldsymbol{I}_n)$ *y* $\boldsymbol{A}$ *una matriz simétrica. Si* $\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y} \sim \chi^2_r$ *entonces* $\boldsymbol{Y}^T(\boldsymbol{I}_n - \boldsymbol{A})\boldsymbol{Y} \sim \chi^2_{n-r}$.

Prueba. Por el teorema B.10 tenemos que $\boldsymbol{A}$ es idempotente pero entonces $\boldsymbol{I}_n - \boldsymbol{A}$ también es idempotente. Además es simétrica por lo que su rango coincide con su traza (A.38) que es igual a $tr(\boldsymbol{I}_n) - tr(\boldsymbol{A}) = n - r$. Se sigue que $\boldsymbol{Y}^T(\boldsymbol{I}_n - \boldsymbol{A})\boldsymbol{Y} \sim \chi^2_{n-r}$. □

Corolario B.5. *Si* $\boldsymbol{A}$ *y* $\boldsymbol{B}$ *son matrices simétricas,* $\boldsymbol{Y} \sim N_n(\boldsymbol{0}, \boldsymbol{I}_n)$ *y tales que* $\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y}$ *y* $\boldsymbol{Y}^T\boldsymbol{B}\boldsymbol{Y}$ *tienen distribuciones ji-cuadrado. Entonces* $\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y}$ *y* $\boldsymbol{Y}^T\boldsymbol{B}\boldsymbol{Y}$ *son independientes si y solo si* $\boldsymbol{A}\boldsymbol{B} = \boldsymbol{0}$.

Prueba. Supongamos que $\boldsymbol{A}\boldsymbol{B} = \boldsymbol{0}$. Por el teorema B.10 tenemos que $\boldsymbol{A}$ y $\boldsymbol{B}$ son idempotentes de donde $\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y} = \boldsymbol{Y}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{Y} = \| \boldsymbol{A}\boldsymbol{Y} \|^2$ y $\boldsymbol{Y}^T\boldsymbol{B}\boldsymbol{Y} = \boldsymbol{Y}^T\boldsymbol{B}^T\boldsymbol{B}\boldsymbol{Y} = \| \boldsymbol{B}\boldsymbol{Y} \|^2$. Pero $\boldsymbol{A}\boldsymbol{Y}$ y $\boldsymbol{B}\boldsymbol{Y}$ son independientes de donde $\boldsymbol{Y}^T\boldsymbol{A}\boldsymbol{Y}$ y $\boldsymbol{Y}^T\boldsymbol{B}\boldsymbol{Y}$ también lo son.

Veamos el recíproco. Suponemos las formas cuadráticas independientes. Ambas asumimos que tienen distribuciones ji-cuadrado. Su suma también tendrá una distribución ji-cuadrado con los grados de libertad igual a la suma de los grados de libertad. En consecuencia $\boldsymbol{Y}^T(\boldsymbol{A} + \boldsymbol{B})\boldsymbol{Y}$ tiene una distribución ji-cuadrado. $\boldsymbol{A} + \boldsymbol{B}$ debe ser idempotente por el teorema B.10. Se verifica pues que

$$\boldsymbol{A} + \boldsymbol{B} = (\boldsymbol{A} + \boldsymbol{B})^2 = \boldsymbol{A}^2 + \boldsymbol{A}\boldsymbol{B} + \boldsymbol{B}\boldsymbol{A} + \boldsymbol{B}^2 = \boldsymbol{A} + \boldsymbol{A}\boldsymbol{B} + \boldsymbol{B}\boldsymbol{A} + \boldsymbol{B},$$

es decir,

$$\boldsymbol{A}\boldsymbol{B} + \boldsymbol{B}\boldsymbol{A} = \boldsymbol{0}.$$

Si en la igualdad anterior multiplicamos por la izquierda por $\boldsymbol{A}$ tenemos $\boldsymbol{A}\boldsymbol{B} + \boldsymbol{A}\boldsymbol{B}\boldsymbol{A}$. Multiplicando la igualdad anterior por la derecha por la matriz $\boldsymbol{A}$ tenemos $\boldsymbol{A}\boldsymbol{B}\boldsymbol{A} + \boldsymbol{B}\boldsymbol{A}$. Se sigue pues que $\boldsymbol{A}\boldsymbol{B} = \boldsymbol{B}\boldsymbol{A} = \boldsymbol{0}$. □

Teorema B.12. *Supongamos* $\boldsymbol{Y} \sim N_n(\boldsymbol{\mu}, \boldsymbol{\Sigma})$ *con* Σ *definida positiva. Entonces* $Q = (\boldsymbol{Y} - \boldsymbol{\mu})^T\boldsymbol{\Sigma}^{-1}(\boldsymbol{Y} - \boldsymbol{\mu}) \sim \chi^2_n$.

Prueba. Hacemos la transformación $\boldsymbol{Y} = \Sigma^{-1/2}\boldsymbol{Z} + \boldsymbol{\mu}$. Tenemos

$$Q = \boldsymbol{Z}^T\Sigma^{-1/2}\Sigma^{-1}\Sigma^{-1/2}\boldsymbol{Z} = \boldsymbol{Z}^T\boldsymbol{Z} = \sum_{i=1}^{n} Z_i^2.$$

Puesto que $Z_i^2 \sim \chi^2_1$ y son independientes entonces $Q \sim \chi^2_n$. □

B.6 Función gamma

Recordamos la definición de función gamma. Definimos como función gamma para todos los números positivos como

$$\Gamma(z) = \int_0^\infty t^{z-1}e^{-t}dt.$$

Encontramos para todos los n enteros para los cuales esta función está definida que $\Gamma(n) = (n-1)!$. Esto se puede demostrar fácilmente a partir de la definición. Sea n entero y positivo tenemos que la función gamma para este valor es:

$$\Gamma(n) = \int_0^\infty t^{n-1}e^{-t}dt = \left[t^{n-1}(-e^{-t})\right]_0^\infty + \int_0^\infty (n-1)t^{n-2}e^{-t}dt = 0+(n-1)\Gamma(n-1). \tag{B.19}$$

Aplicando el método de integración por partes y sustituyendo. Demostrando ahora que $\Gamma(1) = 1$ completamos la demostración por inducción.

$$\Gamma(1) = \int_0^\infty t^0 e^{-t}dt = \int_0^\infty e^{-t}dt = \left[-e^{-t}\right]_0^\infty = 0-(-1) = 1. \tag{B.20}$$

Necesitaremos más adelante los valores de la función gamma cuando $\alpha = n$ o $\alpha = n + \frac{1}{2}$, n natural. Es fácil comprobar, mediante sucesivas integraciones por partes, que

$$\Gamma(\alpha) = (\alpha-1)\Gamma(\alpha-1) = (\alpha-1)(\alpha-2)\Gamma(\alpha-2),$$

lo que para $\alpha = n$ da lugar a

$$\Gamma(n) = (n-1)(n-2)\ldots 2\Gamma(1).$$

Pero

$$\Gamma(1) = \int_0^\infty e^{-x}dx = 1 \quad y \quad \Gamma(n) = (n-1)!.$$

Para el caso en que $\alpha = n + \frac{1}{2}$ deberemos calcular $\Gamma(\frac{1}{2})$,

$$\Gamma\left(\frac{1}{2}\right) = \int_0^\infty e^{-x}x^{-1/2}dx = \left[x = \frac{t^2}{2}\right] = \sqrt{2}\int_0^\infty e^{-t^2/2}dt = \frac{\sqrt{2}\sqrt{2\pi}}{2} = \sqrt{\pi}. \tag{B.21}$$

La última integral en (B.21), dividida por $\sqrt{2\pi}$, es la mitad del área que cubre la función de densidad de probabilidad de la normal estándar $N(0,1)$.

B.7 Distribuciones de probabilidad

En esta sección se consideran algunas distribuciones de probabilidad de interés en el texto.

B.7.1 Distribución binomial negativa

Conocemos que un experimento de Bernoulli consiste en lo siguiente: dos únicos posibles resultados para el experimento, éxito o fracaso, con probabilidad de éxito p y probabilidad de fracaso $1-p$ con $p \in [0,1]$.

Nos fijamos en el número de pruebas adicionales que hemos de realizar hasta obtener el éxito r-ésimo. Notemos que como son pruebas adicionales, el valor mínimo será 0 indicando que las r primeras

pruebas han sido un éxito. Si X es la variable que nos da este número de pruebas adicionales entonces

$$P(X=x)=\binom{r+x-1}{x}p^r(1-p)^x,\ \text{si}\ \ x\geq 0.$$

Esta distribución de probabilidad recibe el nombre de *distribución binomial negativa* y la variable se dice que tiene una distribución binomial negativa.

El nombre de binomial negativa se justifica a partir de la expresión alternativa que admite la función de cuantía,

$$f_X(x)=\begin{cases}\binom{-r}{x}p^r(-(1-p))^x, & si\ x\geq 0\\ \\ 0, & en\ el\ resto,\end{cases}$$

obtenida al tener en cuenta que

$$\begin{aligned}\binom{-r}{x} &= \frac{(-r)(-r-1)\cdots(-r-x+1)}{x!}\\ &= \frac{(-1)^x r(r+1)\cdots(r+x-1)}{x!}\\ &= \frac{(-1)^x(r-1)!r(r+1)\cdots(r+x-1)}{(r-1)!x!}\\ &= (-1)^x\binom{r+x-1}{x}.\end{aligned}$$

Veamos que suma la unidad. Recordemos el desarrollo en serie de potencias de la función $f(x)=(1-x)^{-n}$,

$$\frac{1}{(1-x)^n}=\sum_{i\geq 0}\binom{n+i-1}{i}x^i,\quad |x|<1.$$

En nuestro caso,

$$\begin{aligned}f_X(x)=\sum_{x\geq 0}\binom{r+x-1}{x}p^r(1-p)^x=\\ p^r\sum_{x\geq 0}\binom{r+x-1}{x}(1-p)^x=p^r\frac{1}{(1-(1-p))^r}=1.\end{aligned}\quad (B.22)$$

Ejemplo B.10 (El problema de las cajas de cerillas de Banach)**.** *En un acto académico celebrado en honor de Banach, H. Steinhaus contó una anécdota acerca del hábito de fumar que aquél tenía. La anécdota se refería a la costumbre de Banach de llevar una caja de cerillas en cada uno de los bolsillos de su chaqueta, de manera que cuando necesitaba una cerilla elegía al azar uno de los bolsillos. El interés de la anécdota residía en calcular las probabilidades asociadas al número de cerillas que habría en una caja cuando, por primera vez, encontrara vacía la otra.*

Si cada caja contiene N cerillas, en el momento de encontrar una vacía la otra puede contener $0,1,2,\ldots,N$ cerillas. Designemos por A_r ={el bolsillo no vacío contiene r cerillas}*. Supongamos que la caja vacía es la del bolsillo izquierdo, para que ello ocurra $N-r$*

fracasos *(elecciones del bolsillo derecho) deben haber precedido al* $N+1$-*ésimo* éxito *(elección del bolsillo derecho). En términos de una variable aleatoria* $X \sim BN(N+1, 1/2)$ *se trata de obtener* $P(X = N - r)$. *El mismo argumento puede aplicarse si la caja vacía es la del bolsillo derecho. Así pues,*

$$p_r = P(A_r) = 2P(X = N - r) = 2\binom{2N-r}{N-r}\left(\frac{1}{2}\right)^{N+1}\left(\frac{1}{2}\right)^{N-r} = \binom{2N-r}{N-r}2^{-2N+r}. \quad \text{(B.23)}$$

Por ejemplo, para $N = 50$ *y* $r = 4$, $p_r = 0.074790$*; para* $r = 29$, $p_r = 0.000232$.

La distribución binomial negativa es una distribución discreta de probabilidad que modeliza el número de éxitos en un conjunto de experimentos de Bernoulli independientes entre sí que se va ampliando hasta alcanzar un determinado y preespecificado número de fracasos y. Aunque esta es la concepción más general, otra manera de reescribir esta modelización es tomando como limitante el número de éxitos ϕ y como fracasos la variable a observar y. La función de probabilidad de la distribución es

$$f(y; \phi, p) = \binom{y+\phi-1}{y}(1-p)^y p^\phi = \frac{(y+\phi-1)!}{y!(\phi-1)!}(1-p)^y p^\phi. \quad \text{(B.24)}$$

En la expresión anterior podemos sustituir la función factorial por la función gamma $\Gamma(\cdot)$ que la generaliza. La función de probabilidad anterior quedaría

$$f(y; \phi, p) = \frac{\Gamma(y+\phi)}{\Gamma(y+1)\Gamma(\phi)}(1-p)^y p^\phi. \quad \text{(B.25)}$$

Sea μ la media de la distribución binomial negativa, esta se expresa como $\mu = \frac{p\phi}{1-p}$, cosa que podemos reescribir como $p = \frac{\phi}{\phi+\mu}$. Sustituyendo la última expresión en nuestra función de probabilidad tenemos que

$$f(y|\phi, p) = \frac{\Gamma(y+\phi)}{\Gamma(y+1)\Gamma(\phi)}\left(\frac{\mu}{\phi+\mu}\right)^y\left(\frac{\phi}{\mu+\phi}\right)^\phi. \quad \text{(B.26)}$$

En (B.32) y (B.33) vemos que si μ denota la media de Y entonces su varianza viene dada por $var(Y) = \mu + \frac{1}{\phi}\mu^2$.

En ocasiones, en lugar del parámetro ϕ se toma el inverso del mismo de modo que la función de probabilidad adoptaría la siguiente expresión

$$f(y|\phi, p) = \frac{\Gamma(y+\phi^{-1})}{\Gamma(y+1)\Gamma(\phi^{-1})}\left(\frac{\mu}{\phi^{-1}+\mu}\right)^y\left(\frac{\phi^{-1}}{\mu+\phi^{-1}}\right)^{\phi^{-1}} \quad \text{(B.27)}$$

Distribución binomial negativa como mixtura de distribuciones Poisson

Se puede obtener la función de probabilidad de una distribución binomial negativa como una mixtura de distribuciones de Poisson. Supongamos que Y sigue una distribución de Poisson con parámetro λ

$$f(y|\lambda) = e^{-\lambda}\frac{\lambda^y}{y!}, \quad \text{(B.28)}$$

con $y = 0, 1, \ldots$. Vamos a asumir que el valor del parámetro λ es aleatorio con distribución gamma que suponemos parametrizada con los parámetros de forma y media (B.36), $\lambda \sim Gamma(\alpha, \mu)$. Entonces la función de probabilidad de Y vendrá dada por

$$f(y|\alpha,\mu) = \int_0^{+\infty} f(y|\lambda)g(\lambda|\alpha,\mu)d\lambda = \int_0^{+\infty} e^{-\lambda}\frac{\lambda^y}{y!}\frac{1}{\Gamma(\alpha)}\left(\frac{\alpha}{\mu}\right)^{\alpha}\lambda^{\alpha-1}e^{-\frac{\alpha}{\mu}\lambda} = \frac{1}{y!\Gamma(\alpha)}\left(\frac{\alpha}{\mu}\right)^{\alpha}\int_0^{+\infty} e^{-\lambda(1+\alpha/\mu)}\lambda^{y+\alpha-1}d\lambda. \quad \text{(B.29)}$$

Notemos que

$$\int_0^{+\infty} e^{-\lambda(1+\alpha/\mu)}\lambda^{y+\alpha-1}d\lambda \quad = \quad \Gamma(y\ +\ \alpha)\left(\frac{\mu}{\mu+\alpha}\right)^{y+\alpha}, \quad \text{(B.30)}$$

de donde inmediatamente se sigue

$$f(y|\alpha,\mu) = \frac{\Gamma(y+\alpha)}{\Gamma(\alpha)\Gamma(y+1)}\left(\frac{\mu}{\mu+\alpha}\right)^{y}\left(\frac{\alpha}{\mu+\alpha}\right)^{\alpha}. \quad \text{(B.31)}$$

Sin comparamos la ecuación anterior (B.31) con B.26 vemos que tenemos la misma expresión con $\phi = \alpha$.

Obtener la media y la varianza de la distribución binomial negativa es simple usando propiedades de la media y esperanza de las distribuciones condicionadas. En concreto tenemos

$$EY = E_\lambda[E[Y|\lambda]] = E_\lambda[\lambda] = \mu, \quad \text{(B.32)}$$

ya que $\lambda \sim Gamma(\alpha, \mu)$. En cuanto a la varianza tenemos

$$var(Y) = E_\lambda(var(Y|\lambda)) + var_\lambda(E(Y|\lambda)) = E_\lambda(\lambda) + var_\lambda(\lambda) = \mu + \frac{1}{\alpha}\mu^2. \quad \text{(B.33)}$$

Si denotamos, como es habitual, $\phi = \alpha$ tenemos

$$var(Y) = \mu + \frac{1}{\phi}\mu^2. \quad \text{(B.34)}$$

** **Ex. 39** — Consideremos el problema de las cajas de cerillas propuesto en ejemplo B.10. Se propone encontrar la solución considerando cada una de las variantes siguientes:

1. Que el número de cerillas de cada una de las cajas no es el mismo: una de ellas supondremos que tiene N cerillas y la otra M cerillas.
2. Que el fumador elige su bolsillo derecho (con M cerillas) con el doble de probabilidad que su bolsillo izquierdo (con N cerillas).

B.7.2 Distribución gamma

[115] Distribución gamma

[115] Diremos que la variable aleatoria X tiene una *distribución gamma de parámetros* α *y* β, $X \sim Ga(\alpha, \beta)$, donde $\alpha, \beta > 0$ si su función de densidad es de la forma

$$f_X(x) = \begin{cases} 0, & si\ x \le 0 \\ \dfrac{1}{\Gamma(\alpha)\beta^\alpha}x^{\alpha-1}e^{-x/\beta}, & si\ x > 0 \end{cases} \quad \text{(B.35)}$$

donde $\Gamma(\alpha)$ es el valor de la función Gamma en α.[116] La función definida en B.35 es una función de densidad de probabilidad. Obviamente es no negativa. Para comprobar que su integral sobre toda la recta es uno bastará hacer el cambio $y = x/\beta$ en la correspondiente integral

[116] La función Gamma se define como $\Gamma(\alpha) = \int_0^\infty y^{\alpha-1}e^{-y}dy$ donde $\alpha > 0$.

$$\int_0^\infty \frac{1}{\Gamma(\alpha)\beta^\alpha}x^{\alpha-1}e^{-x/\beta}dx = \frac{1}{\Gamma(\alpha)\beta^\alpha}\int_0^\infty y^{\alpha-1}e^{-y}\beta^\alpha dy = \frac{1}{\Gamma(\alpha)}\Gamma(\alpha) = 1.$$

Obsérvese que la distribución exponencial de parámetro λ es un caso particular de la distribución gamma. En concreto, una exponencial de parámetro λ es una distribución gamma con parámetros 1 y $1/\lambda$.

Es habitual llamar a α parámetro de forma (shape) y a β parámetro de escala.

La media y varianza de la distribución gamma se obtiene fácilmente que tienen la expresión $EX = \alpha\beta$ y $var(X) = \alpha\beta^2$. Esto nos permite obtener una reparametrización de uso habitual. Tenemos $\mu = EX = \alpha\beta$ de donde $\beta = \mu/\alpha$. De este modo tenemos la densidad de la gamma reparametrizada que adopta la siguiente expresión

$$f(x) = \frac{1}{\Gamma(\alpha)}\left(\frac{\alpha}{\mu}\right)^\alpha x^{\alpha-1}e^{-\frac{\alpha}{\mu}x}, \tag{B.36}$$

donde $\alpha, \mu > 0$. Pasamos del vector (α, β) (forma y escala) a (α, μ) (forma y media).

B.7.3 Distribución Ji-cuadrado

[117] Una *variable aleatoria Ji-cuadrado de parámetro* r, $X \sim \chi^2_r$, es una variable aleatoria gamma con $\alpha = r/2$ y $\beta = 2$ siendo r entero no negativo. Por tanto, es una familia de densidades de probabilidad que dependen de un solo parámetro r. Su función de densidad tiene la expresión

[117] Distribución Ji-cuadrado

$$f_X(x) = \begin{cases} 0, & si\ x \le 0 \\ \dfrac{1}{\Gamma(r/2)2^{r/2}}x^{r/2-1}e^{-x/2}, & si\ x > 0. \end{cases}$$

El parámetro r es también conocido como el *número de grados de libertad* de la distribución.

B.7.4 Distribución ji-cuadrado no central

Definición B.8. *Supongamos $X_1, \ldots, X_k$ variables aleatorias independientes y con distribución normal, $X_i \sim N(\mu_i, 1)$. Entonces la variable $Y = \sum_{i=1}^k X_i^2$ sigue una distribución ji-cuadrado no central con k grados de libertad y parámetro de no centralidad $\lambda = \sum_{i=1}^k \mu_i^2$ y lo denotamos $Y \sim \chi^2_{k,\lambda}$.*

Un estudio más detallado lo podemos encontrar en https://en.wikipedia.org/wiki/Noncentral_chi-squared_distribution.

Apéndice C

Código sin más

Incluimos en este capítulo código al que queremos referenciar desde el resto del manual sin demasiados comentarios.

C.1 GSE198668

Fueron inicialmente analizados por Aarón García Blázquez en el curso 2021-22.

C.1.1 Construcción del ExpressionSet

Se encuentran en https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE198668. Cargamos paquetes.

```
pacman::p_load(Biobase,GEOquery,affy)
```

Descargamos datos.

```
dir = getwd()
gcel = getGEOSuppFiles("GSE198668")
setwd("GSE198668/")
system("tar xvf GSE198668_RAW.tar")
GSE198668raw = affy::ReadAffy()
setwd("../")
system("rm -fr GSE198668")
setwd(dir)
```

Normalizados los datos.

```
GSE198668 = affy::rma(GSE198668raw)
```

Construimos el `Biobase::ExpressionSet`.

```
#Añadimos los datos fenotípicos a partir de un documento de texto
pd0 = read.csv(paste0(dirTamiData,"GSE198668_metadata.csv"),
                header=TRUE,row.names ="ID")

#Añadimos los datos del experimento
exd0 = new("MIAME",name="Hellerud et al.",lab = "Medical Biochemistry",
           contact = "o.k.olstad@medisin.uio.no",
           title = "Massive Organ Inflammation in Experimental and in Clinical
           Meningococcal Septic Shock",
           abstract = "The aims of the present study were to investigate the
           inflammatory responses in various organs as compared with the
           systemic circulation in an experimental porcine model of
           meningococcal sepsis using wild-type N. meningitidis;
           to study the role of LPS versus non-LPS microbial molecules as
           triggers of organ inflammation in meningococcal sepsis",
```

```
            url = "https://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=
                ↪ GSE198668")
metadata = data.frame(labelDrescription =
                              c("Órgano del que procede la muestra",
                                "Cepa inoculada", "Dosis"),
                          row.names = colnames(pd0))
datosfenotipo = AnnotatedDataFrame(data = pd0, varMetadata = metadata)
GSE198668 = ExpressionSet(assayData=exprs(GSE198668), phenoData =
    ↪ datosfenotipo,
                          experimentData = exd0, annotation = "porcine.db")
```

Modificamos las variables fenotípicas.

```
pData(GSE198668)$Organ = factor(pData(GSE198668)$Organ)
pData(GSE198668)$Strand = factor(pData(GSE198668)$Strand)
```

El chip empleado en este estudio no posee como clave la entrada de ENSEMBL pero se ha añadido el símbolo del gen a su sonda.

```
pacman::p_load(AnnotationDbi,porcine.db)
probeid2entrez =
    AnnotationDbi::select(porcine.db,keys = featureNames(GSE198668),
                          columns=c("ENTREZID","SYMBOL"),
                          keytype = "PROBEID")
indices = match(featureNames(GSE198668),probeid2entrez$PROBEID)
fData(GSE198668) = probeid2entrez[indices,]
all.equal(fData(GSE198668)$PROBEID,featureNames(GSE198668))
```

Eliminación de las correspondencias múltiples.

```
a = probeid2entrez
c1 = match(unique(a[,1]),a[,1])
a1 = a[c1,]
c2 = match(unique(a1[,2]),a1[,2])
a2 = a1[c2,]
fData(GSE198668) = a2
```

Y lo guardamos.

```
save(GSE198668,file=paste0(dirTamiData,"GSE198668.rda"))
```

C.2 gse80200

Datos procesados inicialmente por Daniel González Gamblor en el curso 2021-22. Se incluye su código con levels modificaciones.

Elegí el experimento GSE80200, un estudio de expresión en Arabidopsis thaliana. Este estudio utiliza la plataforma ATH1-121501 GPL21063 de Affymetrix.

```
pacman::p_load(GEOquery,affy,Biobase,ath1121501.db)
download = getGEOSuppFiles("GSE80200")
untar("GSE80200/GSE80200_RAW.tar")
gse80200_raw = ReadAffy(sampleNames =
                            c("S1_control", "S2_control",
                              "S3_control", "S4_treatment",
                              "S5_treatment", "S6_treatment"))
gse80200 = rma(gse80200_raw)

data = new("MIAME",
           name="Willems P, Denecker J, Van Breusegem F",
           lab="PSB",
           contact="pawil@psb.ugent.be",
           title="Transcriptional responses in Arabidopsis seedlings after
hydrogen peroxide treatment",
abstract="Excessive levels of reactive oxygen species (ROS) cause cellular
```

```
stress through damage to all classes of macromolecules and result in cell
death. However, ROS can also act as signaling molecules in various biological
 processes. In plants, ROS signaling has been documented in environmental
stress perception, plant development and cell death amongst others.
Knowledge on the regulatory events governing ROS signal transduction is
however still scratching the surface. To further elucidate the transcriptional
 response and regulation upon ROS accumulation we supplemented Arabidopsis
 seedlings with a 10mM hydrogen peroxide (H2O2) solution to trigger oxidative
 stress.",
url="https://www-ncbi-nlm-nih-gov.ezproxy.u-pec.fr/geo/query/acc.cgi?acc
    ↪ =GSE80200",
pubMedIds="27246095"
)
experimentData(gse80200) = data

## pData
type = factor(c(0, 0, 0, 1, 1, 1), levels = 0:1, labels=c("H2O", "H2O2"))
pd = data.frame(type)
rownames(pd) = colnames(gse80200)
pData(gse80200) = pd

## ath1121501.db tiene opciones limitadas, no contiene ENSEMBL
## Utiliza `Biomart`
pacman::p_load(biomaRt)

#Select mart
mart = useMart(biomart="plants_mart", host="plants.ensembl.org")
mart = useDataset("athaliana_eg_gene", mart)
# Annotation
info = getBM(mart=mart,
             attributes=c("affy_ath1_121501","entrezgene_id",
                          "ensembl_gene_id","external_gene_name"),
             filters="affy_ath1_121501",
              values = featureNames(gse80200_rma), uniqueRows=TRUE)

##La columna de IDs de ensembl corresponde en realidad con los
    ↪ identificadores
##de la base de datos más popular para *Arabidopsis thaliana*: la
    ↪ Arabidopsis
##Infor#mation Resource ([TAIR](https://www.arabidopsis.org/index.jsp))
    ↪ .

m <-match(featureNames(gse80200_rma), info[,"affy_ath1_121501"])
fData(gse80200) = info[m,]

## Utilizo como identificadores primarios los ENSEMBL
ids <-fData(gse80200)
ids <-ids[,c("affy_ath1_121501", "ensembl_gene_id")]
m1 <-match(unique(ids[,1]), ids[,1])
gse80200 = gse80200[m1,]
ids <-fData(gse80200)
m2 <-match(unique(ids[,2]), ids[,2])
gse80200 = gse80200[m2,]
names(fData(gse80200)) = c("PROBEID","ENTREZID","ENSEMBL","
    ↪ GENENAME")
pData(gse80200)$type = as.factor(pData(gse80200)$type)
save(gse80200, file =paste0(dirTamiData,"gse80200.rda"))
```

C.3 gse21443

Lo que sigue está modificado a partir de un código original de Gonzalo Antón Bernat (20.06.23). La finalidad del estudio es comprender cómo las plantas adquieren y procesan nutrientes como el hierro. Los grupos experimentales son los siguientes: arabidopsis WT con hierro (2 réplicas), arabidopsis WT sin hierro (2 réplicas), arabidopsis pye

con hierro (2 réplicas) y arabidopsis pye sin hierro (2 réplicas).

```
## gse21443
pacman::p_load(GEOquery,affy)
getGEOSuppFiles("GSE21443")
setwd("GSE21443")
system("tar xvf GSE21443_RAW.tar")
gse21443raw = ReadAffy()
gse21443 = affy::rma(gse21443raw)
setwd("../")
datosexperimento ←new('MIAME',
                          name = 'Terri Anita Long',
                          lab = 'Philip Benfey',
                          contact = ' tlong@duke.edu',
                          title = 'Expression analysis of pye−1 mutants
and root pericycle cells to iron sufficient or iron deficient conditions',
pubMedIds = '20675571'
)

experimentData(gse21443) = datosexperimento

muestras = rownames(pData(gse21443))
condicion = factor(rep(0:1,each=4),
                   levels = 0:1, labels = c("Control", "Popeye"))
medio = factor(c(0, 0, 1, 1, 0, 0, 1, 1), levels = 0:1,
               labels = c("Con_hierro", "Sin_hierro"))
replica = factor(c(0, 1, 0, 1, 0, 1, 0, 1), levels = 0:1,
                 labels = c("Replica_1", "Replica_2"))
pd = data.frame(muestras, condicion, medio, replica)
rownames(pd) = colnames(exprs(gse21443))
pData(gse21443) = pd
save(gse21443,file="gse21443.rda")
```

```
pacman::p_load("EnrichmentBrowser")
arabi_go = getGenesets(org='ath', # El organismo debe estar en código
    ↪ KEGG
                       db='go',
                       onto='BP')
save(arabi_go, file = 'arabi_go.rda')

arabi_KEGG = getGenesets(org = 'ath',
                         db='kegg')
save(arabi_KEGG, file = 'arabi_KEGG.rda')
```

C.4 bcrneg

```
pacman::p_load("Biobase","ALL")
data(ALL)
bcell = grep("^B",as.character(ALL$BT))
types = c("NEG","BCR/ABL")
moltyp = which(as.character(ALL$mol.biol) %in% types)
bcrneg = ALL[,intersect(bcell,moltyp)]
save(bcrneg,file=paste0(dirTamiData,"bcrneg.rda"))
```

Bibliografía

[1] A. Agresti. *Categorical Data Analysis*. Second. Wiley, 2002.

[2] A. Agresti. *Categorical Data Analysis*. Third. Wiley-Interscience, 2013.

[3] Alan Agresti. *An Introduction to Categorical Data Analysis*. Third edition. Wiley John + Sons, 2019. 400 págs. ISBN: 1119405262.

[4] Alan Agresti. *Foundations of Linear and Generalized Linear Models*. Wiley, 2015. 480 págs. ISBN: 1118730038.

[5] Alan Agresti y Maria Kateri. *Foundations of Statistics for Data Scientists*. Chapman y Hall/CRC, 2021. DOI: 10.1201/9781003159834.

[6] JJ Allaire et al. *rmarkdown: Dynamic Documents for R*. R package version 2.25, https://pkgs.rstudio.com/rmarkdown/. 2023.

[7] Guillermo Ayala. *tami: Statistical Bioinformatics*. R package version 1.0. 2023.

[8] Guillermo Ayala. *tami: Statistical Bioinformatics*. R package version 1.0. 2023.

[9] Guillermo Ayala. *tamidata: Data sets for Statistical Bioinformatics*. R package version 0.5. 2022.

[10] Guillermo Ayala. *tamidata2: Data sets for Statistical Bioinformatics*. R package version 0.2. 2023.

[11] Guillermo Ayala. *tamidata3: Omics data sets*. R package version 0.3. 2023.

[12] M. Baker y D. Penny. «Is there a reproducibility crisis?» En: *Nature* 533.7604 (2016). cited By 44, págs. 452-454. DOI: 10.1038/533452A.

[13] Sudipto Banerjee y Anindya Roy. *Linear Algebra and Matrix Analysis for Statistics (Chapman & Hall/CRC Texts in Statistical Science)*. Chapman y Hall/CRC, 2014. ISBN: 978-1-4822-4824-1.

[14] Yoav Benjamini y Yosef Hochberg. «Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing». English. En: *Journal of the Royal Statistical Society. Series B (Methodological)* 57.1 (1995), págs. 289-300. ISSN: 00359246.

[15] Yoav Benjamini y Daniel Yekutieli. «The control of the false discovery rate in multiple testing under dependency». En: *The Annals of Statistics* 29.4 (2001), págs. 1165-1188.

[16] M. Carlson et al. *GenomicFeatures: Conveniently import and query gene models*. R package version 1.52.2. 2023. DOI: 10.18129/B9.bioc.GenomicFeatures.

[17] Marc Carlson. *ath1121501.db: Affymetrix Affymetrix ATH1-121501 Array annotation data (chip ath1121501)*. R package version 3.13.0. 2021.

[18] Marc Carlson. *GO.db: A set of annotation maps describing the entire Gene Ontology*. R package version 3.17.0. 2023.

[19] Marc Carlson. *hgu133a.db: Affymetrix Affymetrix HG-U133A Array annotation data (chip hgu133a)*. R package version 3.13.0. 2021.

[20] Marc Carlson. *hgu133plus2.db: Affymetrix Affymetrix HG-U133 Plus 2 Array annotation data (chip hgu133plus2)*. R package version 3.13.0. 2021.

[21] Marc Carlson. *hgu95av2.db: Affymetrix Affymetrix HG U95Av2 Array annotation data (chip hgu95av2)*. R package version 3.13.0. 2021.

[22] Marc Carlson. *org.Hs.eg.db: Genome wide annotation for Human*. R package version 3.17.0. 2023.

[23] Marc Carlson. *org.Mm.eg.db: Genome wide annotation for Mouse*. R package version 3.17.0. 2023.

[24] Yunshun Chen, Aaron T. L. Lun y Gordon K. Smyth. «From reads to genes to pathways: differential expression analysis of RNA-Seq experiments using Rsubread and the edgeR quasi-likelihood pipeline». En: *F1000Research* 5 (2016), pág. 1438. DOI: 10.12688/f1000research.8987.2.

[25] Yunshun Chen et al. *edgeR: Empirical Analysis of Digital Gene Expression Data in R*. R package version 3.42.4. 2023. DOI: 10.18129/B9.bioc.edgeR.

[26] W. G. Cochran. «The distribution of quadratic forms in a normal system, with applications to the analysis of covariance». En: *Mathematical Proceedings of the Cambridge Philosophical Society* 30.2 (1934), págs. 178-191. DOI: 10.1017/S0305004100016595.

[27] Antonio Colaprico et al. «TCGAbiolinks: an R/Bioconductor package for integrative analysis of TCGA data». En: *Nucleic Acids Research* 44.8 (2015), e71-e71. ISSN: 0305-1048. DOI: 10.1093/nar/gkv1507.

[28] Ana Conesa et al. «A survey of best practices for RNA-seq data analysis». En: *Genome Biology* 17.1 (2016), págs. 1-19. ISSN: 1474-760X. DOI: 10.1186/s13059-016-0881-8.

[29] Sean Davis. *GEOquery: Get data from NCBI Gene Expression Omnibus (GEO)*. R package version 2.68.0. 2023. DOI: 10.18129/B9.bioc.GEOquery.

[30] S. Dudoit, J.P. Shaffer y J.C. Boldrick. «Multiple hypothesis testing in microarray experiments». En: *Statistical Science* 18 (2003). Microarrays, págs. 71-103.

[31] Peter K. Dunn y Gordon K. Smyth. *Generalized Linear Models With Examples in R*. Springer New York, 2018. DOI: 10.1007/978-1-4419-0118-7.

[32] Steffen Durinck y Wolfgang Huber. *biomaRt: Interface to BioMart databases (i.e. Ensembl)*. R package version 2.56.1. 2023. DOI: 10.18129/B9.bioc.biomaRt.

[33] Brad Efron y R. Tibshirani. *GSA: Gene Set Analysis*. R package version 1.03.2. 2022.

[34] Bradley Efron y Robert Tibshirani. «On testing the significance of sets». En: *Annals of Applied Statistics* 1.1 (2007). Gene set analysis, págs. 107-129. DOI: 10.1214/07-AOAS101.

[35] B. Ewing y P. Green. «Base-calling of automated sequencer traces using phred. II. Error probabilities.» En: *Genome research* 8 (3 1998), págs. 186-194. ISSN: 1088-9051. ppublish.

[36] Brent Ewing et al. «Base-Calling of Automated Sequencer Traces UsingiPhred./i I. Accuracy Assessment». En: *Genome Research* 8.3 (1998), págs. 175-185. DOI: 10.1101/gr.8.3.175.

[37] S Falcon y R Gentleman. «Using GOstats to test gene lists for GO term association.» En: *Bioinformatics* 23.2 (2007), págs. 257-8.

[38] S. Falcon y R. Gentleman. «Using GOstats to test gene lists for GO term association». En: *Bioinformatics* 23.2 (2007), págs. 257-258. DOI: 10.1093/bioinformatics/btl567. eprint: http://bioinformatics.oxfordjournals.org/content/23/2/257.full.pdf+html.

[39] John Fox. *Regression diagnostics*. Newbury Park, Calif: Sage Publications, 1991. ISBN: 080393971X.

[40] John Fox et al. *effects: Effect Displays for Linear, Generalized Linear, and Other Models*. R package version 4.2-2, https://-socialsciences.mcmaster.ca/jfox/. 2022.

[41] Ludwig Geistlinger, Gergely Csaba y Ralf Zimmer. «Bioconductor's EnrichmentBrowser: seamless navigation through combined results of set- network-based enrichment analysis». En: *BMC Bioinformatics* 17 (2016), pág. 45. DOI: 10.1186/s12859-016-0884-1.

[42] R. Gentleman. *annotate: Annotation for microarrays*. R package version 1.78.0. 2023. DOI: 10.18129/B9.bioc.annotate.

[43] R. Gentleman et al. *Biobase: Base functions for Bioconductor*. R package version 2.60.0. 2023. DOI: 10.18129/B9.bioc.Biobase.

[44] R. Gentleman et al., eds. *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Springer, 2005.

[45] Robert Gentleman. *Category: Category Analysis*. R package version 2.66.0. 2023. DOI: 10.18129/B9.bioc.Category.

[46] Robert Gentleman. «Reproducible Research: A Bioinformatics Case Study». En: *Statistical Applications in Genetics and Molecular Biology* 4.1 (2005), pág. 2. DOI: 10.2202/1544-6115.1034.

[47] Robert Gentleman y Duncan Temple Lang. *Statistical Analyses and Reproducible Research*. Inf. téc. Bioconductor Project. Bioconductor Project Working Papers, 2004.

[48] Robert Gentleman et al. *genefilter: genefilter: methods for filtering genes from high-throughput experiments*. R package version 1.82.1. 2023. DOI: 10.18129/B9.bioc.genefilter.

[49] Alan Genz et al. *mvtnorm: Multivariate Normal and t Distributions*. R package version 1.2-4. 2023.

[50] J. J. Goeman y P. Buhlmann. «Analyzing gene expression data in terms of gene sets: methodological issues». En: *Bioinformatics* 23.8 (2007), págs. 980-987. DOI: 10.1093/bioinformatics/btm051. eprint: http://bioinformatics.oxfordjournals.org/content/23/8/980.full.pdf+html.

[51] Cedric Gondro. *Primer to Analysis of Genomic Data Using R*. Springer International Publishing, 2015. DOI: 10.1007/978-3-319-14475-7.

[52] Malachi Griffith et al. «Informatics for RNA Sequencing: A Web Resource for Analysis on the Cloud». En: *PLOS Computational Biology* 11.8 (2015), págs. 1-20. DOI: 10.1371/journal.pcbi.1004393.

[53] Haglund et al. «Evidence of a functional estrogen receptor in parathyroid adenomas». En: *J. Clin. Endocrinol. Metab.* 97.12 (2012), págs. 4631-4639.

[54] Florian Hahne et al. *Bioconductor Case Studies*. Use R! Springer, 2008.

[55] Melanie A. Huntley et al. «ReportingTools: an automated result processing and presentation toolkit for high-throughput genomic analyses». En: *Bioinformatics* 29.24 (2013), págs. 3220-3221. ISSN: 1367-4803. DOI: 10.1093/bioinformatics/btt551.

[56] Rafael A. Irizarry et al. *affy: Methods for Affymetrix Oligonucleotide Arrays*. R package version 1.78.2. 2023. DOI: 10.18129/B9.bioc.affy.

[57] Eija Korpelainen et al. *RNA-seq Data Analysis A Practical Approach*. CRC Press, 2015.

[58] Nan M. Laird y Christoph Lange. *The Fundamentals of Modern Statistical Genetics*. Springer New York, 2011. DOI: 10.1007/978-1-4419-7338-2.

[59] Xiaochun Li. *ALL: A data package*. R package version 1.42.0. 2023. DOI: 10.18129/B9.bioc.ALL.

[60] Daniel Lüdecke. *ggeffects: Create Tidy Data Frames of Marginal Effects for ggplot from Model Outputs*. R package version 1.3.4. 2023.

[61] Daniel Lüdecke. «ggeffects: Tidy Data Frames of Marginal Effects from Regression Models». En: *Journal of Open Source Software* 3.26 (2018), pág. 772. DOI: 10.21105/joss.00772.

[62] A. T. L. Lun, Y. Chen y G. K. Smyth. «It's DE-licious: A Recipe for Differential Expression Analyses of RNA-seq Experiments Using Quasi-Likelihood Methods in edgeR.» En: *Statistical Genomics. Methods and Protocols*. Ed. por E. Mathé y S. Davis. Vol. 1418. 2016. Cap. 19. DOI: 10.1007/978-1-4939-3578-9_19.

[63] Davis J. McCarthy, Yunshun Chen y Gordon K. Smyth. «Differential expression analysis of multifactor RNA-Seq experiments with respect to biological variation». En: *Nucleic Acids Research* 40.10 (2012), págs. 4288-4297. DOI: 10.1093/nar/gks042. eprint: http://nar.oxfordjournals.org/content/40/10/4288.full.pdf+html.

[64] P. McCullagh y John A. Nelder. *Generalized Linear Models*. Taylor & Francis Ltd, 11989. 532 págs. ISBN: 0412317605.

[65] V.K . Mootha et al. «PGC-1alpha-responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes». En: *Nature Genetics* 34 (2003), págs. 267-73.

[66] Martin Morgan, Seth Falcon y Robert Gentleman. *GSEABase: Gene set enrichment data structures and methods*. R package version 1.62.0. 2023. DOI: 10.18129/B9.bioc.GSEABase.

[67] Martin Morgan y Marcel Ramos. *BiocManager: Access the Bioconductor Project Package Repository*. R package version 1.30.22. 2023.

[68] Martin Morgan et al. *SummarizedExperiment: SummarizedExperiment container*. R package version 1.30.2. 2023. DOI: 10.18129/B9.bioc.SummarizedExperiment.

[69] Tobias Oetiker et al. *La introducción no-tan-corta a LATeX 2e*. 2010.

[70] Assaf Oron y Robert Gentleman. *GSEAlm: Linear Model Toolset for Gene Set Enrichment Analysis*. R package version 1.60.0. 2023. DOI: 10.18129/B9.bioc.GSEAlm.

[71] Alicia Oshlack, Mark Robinson y Matthew Young. «From RNA-seq reads to differential expression results». En: *Genome Biology* 11.12 (2010), pág. 220. ISSN: 1465-6906. DOI: 10.1186/gb-2010-11-12-220.

[72] Hervé Pagès et al. *AnnotationDbi: Manipulation of SQLite-based annotations in Bioconductor*. R package version 1.62.2. 2023. DOI: 10.18129/B9.bioc.AnnotationDbi.

[73] Katherine S. Pollard, Sandrine Dudoit y Mark J. van der Laan. *Multiple Testing Procedures: R multtest Package and Applications to Genomics, in Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Springer, 2005.

[74] «Preliminary Report: Findings from the Aspirin Component of the Ongoing Physicians' Health Study». En: *New England Journal of Medicine* 318.4 (1988), págs. 262-264. DOI: 10.1056/nejm198801283180431.

[75] C.R. Rao. *Linear Statistical Inference and Its Applications*. Wiley, 1967.

[76] Anat Reiner, Daniel Yekutieli y Yoav Benjamini. «Identifying differentially expressed genes using false discovery rate controlling procedures». En: *Bioinformatics* 19.3 (2003), págs. 368-375. DOI: 10.1093/bioinformatics/btf877. eprint: http://bioinformatics.oxfordjournals.org/content/19/3/368.full.pdf+html.

[77] Mark D. Robinson y Gordon K. Smyth. «Moderated statistical tests for assessing differences in tag abundance». En: *Bioinformatics* 23.21 (2007), págs. 2881-2887. DOI: 10.1093/bioinformatics/btm453. eprint: http://bioinformatics.oxfordjournals.org/content/23/21/2881.full.pdf+html.

[78] Mark D. Robinson y Gordon K. Smyth. «Small-sample estimation of negative binomial dispersion, with applications to SAGE data». En: *Biostatistics* 9.2 (2008), págs. 321-332. DOI: 10.1093/biostatistics/kxm030. eprint: http://biostatistics.oxfordjournals.org/content/9/2/321.full.pdf+html.

[79] George A. F. Seber y Alan J. Lee. *Linear Regression Analysis*. Wiley, 2003. ISBN: 0-471-41540-5.

[80] P.P. Sinha. *Bioinformatics with R Cookbook*. Packt Publishing, 2014.

[81] Gordon Smyth et al. *limma: Linear Models for Microarray Data*. R package version 3.56.2. 2023. DOI: 10.18129/B9.bioc.limma.

[82] Gordon K. Smyth. «Linear Models and Empirical Bayes Methods for Assessing Differential Expression in Microarray Experiments». En: *Statistical Applications in Genetics and Molecular Biology* 1 (2004), pág. 3.

[83] Aravind Subramanian et al. «Gene set enrichment analysis: A knowledge-based approach for interpreting genome-wide expression profiles». En: *Proceedings of the National Academy of Sciences of the United States of America* 102.43 (2005), págs. 15545-15550. DOI: 10.1073/pnas.0506580102. eprint: http://www.pnas.org/content/102/43/15545.full.pdf+html.

[84] Dan Tenenbaum y Bioconductor Package Maintainer. *KEGGREST: Client-side REST access to the Kyoto Encyclopedia of Genes and Genomes (KEGG)*. R package version 1.40.1. 2023. DOI: 10.18129/B9.bioc.KEGGREST.

[85] Laura A. Thomson. *R (and S-PLUS) Manual to Accompany Agresti's Categorical Data Analysis (2002) 2nd edition*. 2009.

[86] Lu Tian et al. «Discovering statistically significant pathways in expression profiling studies». En: *Proceedings of the National Academy of Sciences of the United States of America* 102.38 (2005), págs. 13544-13549. DOI: 10.1073/pnas.0506577102. eprint: http://www.pnas.org/content/102/38/13544.full.pdf+html.

[87] R. Tibshirani et al. *samr: SAM: Significance Analysis of Microarrays*. R package version 3.0. 2018.

[88] Virginia Goss Tusher, Robert Tibshirani y Gilbert Chu. «Significance analysis of microarrays applied to the ionizing radiation response». En: *Proceedings of the National Academy of Sciences* 98.9 (2001). Gene set analysis, págs. 5116-5121. DOI: 10.1073/pnas.091062498. eprint: http://www.pnas.org/content/98/9/5116.full.pdf+html.

[89] H. Wickham. *Advanced R*. CRC, 2014.

[90] Hadley Wickham. *reshape: Flexibly Reshape Data*. R package version 0.8.9. 2022.

[91] Hadley Wickham. «Reshaping Data with the reshape Package». En: *Journal of Statistical Software* 21.1 (2007), págs. 1-20. ISSN: 1548-7660. DOI: 10.18637/jss.v021.i12.

[92] Hadley Wickham et al. *ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. R package version 3.4.4, https://github.com/tidyverse/ggplot2. 2023.

[93] G. N. Wilkinson y C. E. Rogers. «Symbolic Description of Factorial Models for Analysis of Variance». En: *Applied Statistics* 22.3 (1973), pág. 392. DOI: 10.2307/2346786.

[94] Simon N. Wood. *Generalized Additive Models: An Introduction with R*. second. Chapman y Hall/CRC, 2017. DOI: 10.1201/9781315370279.

[95] Di Wu y Gordon K. Smyth. «Camera: a competitive gene set test accounting for inter-gene correlation». En: *Nucleic Acids Research* 40.17 (2012), e133. DOI: 10.1093/nar/gks461. eprint: http://nar.oxfordjournals.org/content/40/17/e133.full.pdf+html.

[96] Yihui Xie. *Dynamic Documents with R and knitr*. 2nd. Chapman & Hall/CRC The R Series. Chapman y Hall/CRC, 2015.

[97] Yihui Xie. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.45. 2023.

[98] Yihui Xie, Joe Cheng y Xianying Tan. *DT: A Wrapper of the JavaScript Library DataTables*. R package version 0.31. 2023.

[99] Guangchuang Yu. *clusterProfiler: A universal enrichment tool for interpreting omics data*. R package version 4.8.3. 2023. DOI: 10.18129/B9.bioc.clusterProfiler.

[100] Hao Zhu. *kableExtra: Construct Complex Table with 'kable' and Pipe Syntax*. R package version 1.3.4. 2021.

[101] Hao Zhu. *kableExtra: Construct Complex Table with kable and Pipe Syntax*. R package version 1.3.4, https://github.com/haozhu233/kableExtra. 2021.

Índice alfabético

Glosario

Affymetrix http://www.affymetrix.com/. 180, 228, 237, 238

ASCII Sistema de codificación de caracteres: American Standard Code for Information Interchange. https://en.wikipedia.org/wiki/ASCII. 21

Bash https://en.wikipedia.org/wiki/Bash_(Unix_shell). ix, 221

Bioconductor Red de espejos con paquetes de R para análisis de datos ómicos: https://www.bioconductor.org/. ix–xii, 8, 14, 21, 129, 146, 176, 206, 231, 233

BioMart http://www.biomart.org Es un sistema de almacenamiento de datos orientado a las consultas. Ha sido desarrollado por el European Bioinformatics Institute (EBI) y el Cold Spring Harbor Laboratory (CSHL). . 248

CDS Es la región codificante de un gen: **C**oding **D**NA **S**equence. Es la parte del gen (DNA o RNA) compuesta por los exones que codifican proteína. El CDS es la porción de un transcrito que es trasladado por un ribosoma. . 245, 246

Debian Una distribución Linux. Posiblemente la mejor de todas ellas y la que está debajo de otras muchas. Consultar https://www.debian.org/ y https://en.wikipedia.org/wiki/Debian.. xi, 301

emacs Editor de textos que puede ser configurado para trabajar con R y Markdown o LaTeXy otros lenguajes de programación. https://www.gnu.org/software/emacs/ . 222

Ensembl https://en.wikipedia.org/wiki/Ensembl. 17, 180, 225, 226, 228, 240, 250–252

Entrez https://en.wikipedia.org/wiki/Entrez. 14, 178, 180, 183, 225, 228, 238, 240, 250–252

FASTA https://en.wikipedia.org/wiki/FASTA_format. 19

FASTQ Es un formato para guardar datos de secuencias. Consiste de cuatro líneas. La primera contiene el nombre de la secuencia. La segunda línea contiene a la propia secuencia. La tercera línez contiene información opcional sobre la secuencia. La cuarta línea cuantifica la confianza o calidad en la determinación de cada base recogida en la segunda línea. https://en.wikipedia.org/wiki/FASTQ_format . 20

FDR False discovery rate o tasa de falso rechazo.. 207, 224

función Gamma $\Gamma(\alpha) = \int_0^\infty y^{\alpha-1}e^{-y}dy$. 281

Gene Ontology Abreviadamente GO. Definen conceptos o clases que son utilizados para describir la función de un gen y relaciones entre estos conceptos. Se clasifican las funciones atendiendo a tres aspectos: MF o función molecular que describe las actividades moleculares de los productos de los genes; CC o componente celular indicando dónde los productos de los genes son activos; BP o procesos biológicos que indican las rutas y procesos indicando actividad de los productos de varios genes: https://en.wikipedia.org/wiki/Gene_ontology y en http://geneontology.org/. Los términos GO están organizados en un grafo acíclico dirigido. . 175, 177, 178, 180, 182–184, 189–191, 195, 196, 199, 200, 203, 211, 213, 225, 226, 228, 240, 241, 252

GEO NCBI Gene Expression Omnibus. Su dirección es http://www.ncbi.nlm.nih.gov/geo/. Es un repositorio público con datos experimentales de alto rendimiento. Tenemos experimentos basados en microarrays de uno o dos canales que miden mRNA, DNA genómico, presencia de proteinas. También hay otras técnicas no basadas de arrays como análisis serial de expresión de genes (SAGE), datos proteómicos obtenidos son espectrometría de masas o datos de secuenciación de alto rendimiento. Hay cuatro tipos de entidades básicas en GEO: Sample, Platform, Series, DataSet. En esta sección mostramos cómo obtener datos de esta gran base de datos pública. Los datos que nos bajamos los utilizaremos en los siguientes temas. Cuando accedemos a una entrada de GEO podemos ver el enlace **Analyze with GEO2R**. Si accedemos a este enlace podemos ver que nos ofrece algunos análisis de los datos utilizando R/Bioconductor.. 21

HTML https://en.wikipedia.org/wiki/HTML. 220, 223, 301

i.i.d. Independientes y con la misma distribución. En definitiva que tenemos una muestra aleatoria de una variable o vector aleatorio.. 30, 85

KEGG Es una colección de mapas de rutas representando interacciones moleculares y grafos de interacción. Estas rutas cubren muchos procesos bioquímicos que se pueden dividir en: metabolismo, proceso de información genética, proceso de información medioambiental, procesos celulares, sistemas de organismos, enfermedades humanas, desarrollo de medicamentos: http://www.genome.jp/kegg/, https://en.wikipedia.org/wiki/KEGG. . 175, 184, 196, 214, 226

Markdown Es un lenguaje de marcas ligero (minimal sería más correcto). Pretende ser una forma rápida de escribir HTML. Según su autor, John Gruber, "HTML is a publishing format; Markdown is a writing format." La dirección indicada contiene una exposición de este lenguaje de marcas. Al ser tan limitado en su

sintaxis han aparecido diferentes variaciones que lo extienden (entre otras cosas para tablas) de las cuales la más interesante y usada es Pandoc Markdown.. xi

NCBI-SRA NCBI SRA (**S**equence **Read A**rchive) es una base de datos con datos de secuenciación de DNA en forma de lecturas cortas generadas mediante secuenciación de alto rendimiento. Su dirección es http://www.ncbi.nlm.nih.gov/sra. En http://www.ncbi.nlm.nih.gov/books/NBK47528/ tenemos más información. Si no funciona el enlace buscamos en Google *SRA handbook*.. 21

Pandoc Es un programa que convierte entre distintos lenguajes de marcas inicialmente desarrollado por John McFarlane. En particular, los paquetes [97], [6, rmarkdown] y Quarto lo utilizan. https://pandoc.org/.. 301

pdf https://en.wikipedia.org/wiki/PDF.. 223

Python https://en.wikipedia.org/wiki/Python_(programming_language). ix

Quarto Es un sistema de escritura de documentos científicos en la forma de documentos dinámicos y que utiliza Pandoc y se puede https://quarto.org/.. xi, 220, 222, 301

R R es un entorno de programación estadístico. https://cran.r-project.org/, https://www.r-project.org/ . ix–xii, 40, 41, 44, 50, 85, 129, 161, 176, 206, 220, 221, 231, 233, 274, 301

RMA Robust multichip average **??**. Ver **??**.. 13, 17

RMarkdown El paquete [6, rmarkdown] incorpora una implementación del lenguaje de marcas Markdown que utilizado conjuntamente con R permite generar informes. La página http://rmarkdown.rstudio.com/ es el mejor lugar para aprender a manejarlo. . 221

RNA-Seq https://en.wikipedia.org/wiki/RNA-Seq. . 19

t-test https://en.wikipedia.org/wiki/Student%27s_t-test. 50

TCGA **T**he **C**ancer **G**enome **A**tlas es una plataforma creada por el NCI (*National Cancer Institute*) y la National Human Genome Research Institute. Contiene bases de datos online con estudios de cáncer utilizando distintas técnicas https://cancergenome.nih.gov/. . 23

Ubuntu Distribución Linux basada en la distribución Debian y que resulta de más fácil instalación. https://www.ubuntu.com/ y https://en.wikipedia.org/wiki/Ubuntu.. xi