



---

**Guillermo Ayala** Universidad de Valencia

# Bioinformática Estadística

Análisis estadístico de datos ómicos



*Copyright ©9 de mayo de 2018*

*Guillermo Ayala*

*[Guillermo.Ayala@uv.es](mailto:Guillermo.Ayala@uv.es)*

*This work is free. You can redistribute it and/or modify it under the terms of the*

*Do What The Fuck You Want To Public License, Version 2, as published by Sam*

*Hocevar. See <http://www.wtfpl.net/> for more details.*



# Índice general

<b>I Introducción</b>	<b>1</b>
<b>1 Estadística y datos ómicos</b>	<b>3</b>
1.1 Estructura de los datos . . . . .	3
1.2 Análisis de datos . . . . .	4
1.3 Sobre herramientas online . . . . .	5
1.4 Paquetes transversales . . . . .	6
1.5 Jerga . . . . .	6
1.6 Datos ordenados . . . . .	6
1.7 Bibliografía . . . . .	6
<b>2 R y Bioconductor</b>	<b>7</b>
2.1 Sobre R y su instalación . . . . .	7
2.1.1 De cómo instalarlo . . . . .	7
2.2 Lo primero con R . . . . .	8
2.3 Una sola muestra . . . . .	9
2.3.1 Varias muestras . . . . .	12
2.4 Datos golub . . . . .	14
2.5 La función apply . . . . .	19
2.6 Listas . . . . .	22
2.7 Funciones con R . . . . .	23
2.8 Sobre Bioconductor . . . . .	26
2.9 Paquetes para tami . . . . .	26
<b>3 Anotación</b>	<b>27</b>
3.1 AnnotationDbi . . . . .	28
3.2 ChipDb . . . . .	30
3.3 OrgDb . . . . .	32
3.4 TxDb . . . . .	34
3.5 BSgenome . . . . .	44
3.6 OrganismDb . . . . .	47
3.7 biomaRt . . . . .	48
3.8 KEGGREST . . . . .	49
3.9 Tareas habituales con anotaciones . . . . .	49
3.9.1 Cambiar identificadores de un ExpressionSet . . . . .	49
3.10 Ejercicios . . . . .	52
<b>II Datos</b>	<b>53</b>
<b>4 Microarrays</b>	<b>55</b>
4.1 Introducción . . . . .	55
4.2 Affymetrix GeneChip . . . . .	55
4.3 Obteniendo los datos . . . . .	56

4.4	Sonda y conjunto de sondas . . . . .	56
4.5	Variabilidad intra y entre arrays . . . . .	57
4.6	Control de calidad . . . . .	60
4.6.1	Dibujo de la media-diferencia de Tukey o dibujo de Bland-Altman . . . . .	60
4.6.2	MA plots . . . . .	61
4.6.3	Densidades y diagramas de cajas . . . . .	63
4.6.4	Utilizando arrayQualityMetrics . . . . .	64
4.6.5	Corrigiendo errores técnicos . . . . .	64
4.7	Método MAS5 . . . . .	65
4.7.1	Corrección de fondo . . . . .	65
4.7.2	Cálculo del valor de expresión . . . . .	66
4.8	Robust multichip average (RMA) . . . . .	67
4.8.1	Corrección de fondo . . . . .	67
4.8.2	Normalización de cuantiles . . . . .	68
4.8.3	Resumen . . . . .	70
4.9	MAS5 y RMA con affy . . . . .	71
4.10	Otros métodos posibles . . . . .	73
4.11	Cómo hacer un preprocesado sencillo a partir de datos de expresión . . . . .	74
4.12	Lecturas complementarias . . . . .	74
<b>5</b>	<b>Datos de microarrays</b>	<b>75</b>
5.1	Introducción . . . . .	75
5.2	sample.ExpressionSet . . . . .	75
5.3	Datos golub . . . . .	77
5.4	Datos ALL . . . . .	79
5.5	Construyendo un ExpressionSet . . . . .	80
5.6	Un experimento con levadura . . . . .	86
5.7	De cómo utilizar un ExpressionSet . . . . .	87
5.8	NCBI GEO . . . . .	90
5.9	GSE21779 . . . . .	90
5.10	GSE1397 . . . . .	91
5.11	Datos GSE20986 . . . . .	94
5.12	GSE34764 . . . . .	95
5.13	ArrayExpress . . . . .	95
5.14	Varios . . . . .	96
5.14.1	Guardando la matriz de expresión . . . . .	96
5.15	Ejercicios . . . . .	96
<b>6</b>	<b>Datos de RNA-seq</b>	<b>99</b>
6.1	Introducción . . . . .	99
6.2	Flujo de trabajo con RNA-seq . . . . .	99
6.3	Repositorios . . . . .	101
6.4	Formato FASTA . . . . .	101
6.5	Formato FASTQ . . . . .	101
6.6	De SRA a fastq . . . . .	106
6.7	Control de calidad de un experimento RNA-seq con EDA-Seq . . . . .	106
6.8	STAR y samtools . . . . .	107
6.9	Bowtie2 . . . . .	108
6.10	Utilizando tophat y samtools . . . . .	111
6.11	GSE37211: paquete parathyroidSE . . . . .	111
6.12	GSE64099 . . . . .	113

6.13 GSE63776 a partir de conteos . . . . .	114
6.14 Normalización de RNA-seq . . . . .	115
6.14.1 RPKM . . . . .	117
6.14.2 Método TMM: Media ajustada de M-valores . . . . .	117
6.14.3 Mediana de los cocientes . . . . .	119
6.14.4 Homogeneizando los percentiles . . . . .	121
6.15 Estudios de caso . . . . .	121
6.15.1 SRP064411 . . . . .	121
<b>III Expresión diferencial</b>	<b>125</b>
<b>7 Expresión diferencial marginal</b>	<b>127</b>
7.1 Introducción . . . . .	127
7.2 Algo de notación . . . . .	127
7.3 Selección no específica o filtrado independiente . . . . .	128
7.3.1 Utilizando apply . . . . .	129
7.3.2 Utilizando genefilter . . . . .	131
7.3.3 Utilizando nsFilter . . . . .	133
7.4 Fold-change . . . . .	133
7.5 Los peligros de la selección no específica . . . . .	134
7.6 Expresión diferencial de un solo gen . . . . .	135
7.7 Comparamos dos condiciones . . . . .	137
7.8 Comparando más de dos condiciones . . . . .	139
7.9 Ejercicios . . . . .	141
<b>8 Comparaciones múltiples</b>	<b>143</b>
8.1 Introducción . . . . .	143
8.2 Control fuerte y control débil del error . . . . .	146
8.3 Relación entre las tasas de error tipo I . . . . .	147
8.4 p valores y p valores ajustados . . . . .	147
8.5 Métodos que controlan la FWER . . . . .	148
8.5.1 Los métodos en un solo paso . . . . .	148
8.5.2 Los métodos por pasos de bajada . . . . .	149
8.5.3 Método por pasos de subida . . . . .	150
8.6 Métodos que controlan el FDR . . . . .	150
8.7 Utilizando genefilter y p.adjust . . . . .	151
8.7.1 Cálculo de p-valores ajustados . . . . .	151
8.7.2 Genes con expresión diferencial . . . . .	151
8.8 El q-valor . . . . .	152
8.9 Ejercicios . . . . .	156
<b>9 Expresión diferencial con microarrays</b>	<b>157</b>
9.1 Método SAM . . . . .	157
9.1.1 El método . . . . .	157
9.1.2 Cálculo de $s_0$ . . . . .	160
9.1.3 SAM con samr . . . . .	160
9.1.4 SAM con otro tipo de covariables . . . . .	161
9.1.5 Ejercicios . . . . .	162
9.2 Limma . . . . .	162
9.2.1 El modelo limma . . . . .	163
9.2.2 Datos gse44456 con limma . . . . .	166
9.2.3 Un diseño cruzado con limma . . . . .	168
9.3 Un comentario técnico importante . . . . .	171

<b>10 Expresión diferencial con datos RNASeq</b>	<b>173</b>
10.1 Introducción . . . . .	173
10.2 Una muestra por condición . . . . .	173
10.3 edgeR . . . . .	175
10.3.1 Estimación de una dispersión común . . . . .	175
10.3.2 Un test exacto . . . . .	176
10.3.3 Dispersiones posiblemente distintas . . . . .	178
10.3.4 edgeR: PRJNA297664 . . . . .	179
10.4 Un método de cuasi-verosimilitud . . . . .	180
10.5 SAMseq . . . . .	180
10.5.1 SAMseq: PRJNA297664 . . . . .	182
10.6 Un método de cuasi-verosimilitud . . . . .	183
10.7 SAMseq . . . . .	184
10.7.1 SAMseq: PRJNA297664 . . . . .	186
10.8 DESeq2 . . . . .	186
10.8.1 Método . . . . .	186
10.8.2 Paquete . . . . .	190
10.9 Material adicional . . . . .	190
10.10 Ejercicios . . . . .	191
<b>11 Tamaño muestral y potencia</b>	<b>193</b>
<b>12 Generando un informe</b>	<b>195</b>
12.1 Identificando el grupo de genes . . . . .	196
12.2 R2HTML . . . . .	197
12.3 ReportingTools . . . . .	198
12.4 Ejercicios . . . . .	199
<b>IV Reducción de dimensión y clasificación</b>	<b>201</b>
<b>13 Componentes principales</b>	<b>203</b>
13.1 Introducción . . . . .	203
13.2 Componentes principales . . . . .	203
13.3 Componentes principales de los datos golub . . . . .	208
13.4 ¿Muestras o genes? . . . . .	211
13.5 ¿Tipificamos los datos? . . . . .	213
13.6 Ejemplos . . . . .	214
13.7 Componentes principales . . . . .	216
<b>14 Análisis cluster</b>	<b>219</b>
14.1 Introducción . . . . .	219
14.2 Datos . . . . .	220
14.2.1 Un ejemplo artificial . . . . .	220
14.2.2 Un ejemplo con muestras . . . . .	221
14.2.3 Un ejemplo con genes . . . . .	221
14.3 Disimilaridades . . . . .	222
14.3.1 Distancias euclídea y de Manhattan . . . . .	222
14.3.2 Disimilaridades utilizando coeficientes de correlación . . . . .	223
14.3.3 Matriz de disimilaridades . . . . .	224
14.4 Disimilaridades entre grupos de observaciones . . . . .	226
14.5 Cluster jerárquico . . . . .	226
14.6 Distancia cofenética . . . . .	229
14.7 Métodos de particionamiento . . . . .	230

14.7.1 Método de las k-medias . . . . .	231
14.7.2 Particionamiento alrededor de los mediodes . . . . .	232
14.8 Silueta . . . . .	233
14.9 Análisis cluster y datos de expresión de gen . . . . .	235
14.10 Ejemplos . . . . .	236
14.10.1 Un análisis de los datos ALL . . . . .	236
14.10.2 Análisis cluster de GSE20986 . . . . .	239
14.11 Otras aproximaciones y problemas . . . . .	239
14.12 Ejercicios . . . . .	240
<b>V Análisis de grupos de genes</b>	<b>241</b>
<b>15 Grupos de genes</b>	<b>243</b>
15.1 Introducción . . . . .	243
15.2 Homo sapiens . . . . .	244
15.3 Grupos con levadura . . . . .	246
15.4 Grupos para GSE1397 . . . . .	247
15.5 Grupos GO para levadura . . . . .	249
15.6 Grupos GO para humanos . . . . .	250
15.7 Grupos para Arabidopsis thaliana . . . . .	250
15.8 Ejercicios . . . . .	250
<b>16 Test de Fisher unilateral</b>	<b>251</b>
16.1 Introducción . . . . .	251
16.2 fisher.test . . . . .	252
16.3 Sobre la elección del universo de genes . . . . .	253
16.4 Utilizando Category y GOstats . . . . .	255
16.4.1 GSE1397 . . . . .	255
16.4.2 GSE20986 . . . . .	257
16.4.3 ALL . . . . .	259
16.5 Ejercicios . . . . .	261
<b>17 Análisis de conjuntos de genes</b>	<b>263</b>
17.1 Introduction . . . . .	263
17.2 Sobre la distribución de la matriz de expresión . . . . .	263
17.3 Conjunto(s) de genes . . . . .	264
17.4 Ejemplos . . . . .	265
17.4.1 Un ejemplo simulado de Efron y Tibshirani . . . . .	265
17.5 Cuantificando asociación gen-fenotipo . . . . .	267
17.6 Enriqueciendo el conjunto de genes . . . . .	267
17.7 Distribuciones condicionadas a los datos . . . . .	268
17.7.1 Distribución de permutación para un gen . . . . .	268
17.7.2 Distribución de aleatorización . . . . .	269
17.8 Limma::genSetTest . . . . .	270
17.9 Limma::wilcoxGST . . . . .	270
17.10 Limma::CAMERA . . . . .	271
17.11 GSA . . . . .	272
17.12 GSEA: Gene set enrichment analysis . . . . .	275
17.13 Estudiando interacciones entre genes individuales y grupos de genes . . . . .	277
17.14 Un método simple pero efectivo . . . . .	278
17.15 Un método basado en poblaciones finitas . . . . .	279
17.16 Análisis de grupos de genes por muestra . . . . .	279
17.16.1 GSVA . . . . .	280

17.1 Otros paquetes y bibliografía complementaria . . . . .	281
<b>18 Enriquecimiento de grafos . . . . .</b>	<b>283</b>
18.1 Grafos . . . . .	283
18.2 NEAT . . . . .	283
18.3 DEGraph . . . . .	284
<b>VI Agregación . . . . .</b>	<b>287</b>
<b>19 Listas de características ordenadas . . . . .</b>	<b>289</b>
19.1 Estabilidad de la lista ordenada . . . . .	290
19.1.1 Diferentes criterios de ordenación . . . . .	290
19.1.2 Estabilidad frente a cambios en los datos . . . . .	294
19.2 Agregación de listas . . . . .	297
<b>VII Probabilidad y Estadística . . . . .</b>	<b>299</b>
<b>20 Estadística descriptiva . . . . .</b>	<b>301</b>
20.1 Datos . . . . .	301
20.2 Introducción . . . . .	302
20.3 Descriptivas numéricas . . . . .	302
20.3.1 Media muestral . . . . .	303
20.3.2 Media ajustada . . . . .	303
20.3.3 Percentiles . . . . .	304
20.3.4 Varianza y desviación estándar muestrales . . . . .	305
20.3.5 Rango . . . . .	306
20.3.6 Rango intercuartílico . . . . .	306
20.3.7 La función genérica summary . . . . .	307
20.4 MAD . . . . .	307
20.5 Ejercicios . . . . .	307
20.6 Descripciones gráficas de los datos . . . . .	308
20.6.1 Añadimos variables y seleccionamos casos o variables . . . . .	308
20.6.2 Frecuencias . . . . .	308
20.6.3 Histograma . . . . .	309
20.6.4 Diagramas de cajas . . . . .	310
20.6.5 Estimadores kernel de la densidad . . . . .	311
20.6.6 Buscando datos anómalos . . . . .	311
20.7 Ejercicios . . . . .	312
<b>21 Probabilidad . . . . .</b>	<b>315</b>
21.1 Experimento, suceso y probabilidad . . . . .	315
21.1.1 Contando: variaciones, permutaciones y combinaciones . . . . .	320
21.1.2 Un poco de teoría . . . . .	323
21.2 Ejercicios . . . . .	324
21.3 Variable aleatoria . . . . .	325
21.3.1 Variable aleatoria discreta . . . . .	325
21.3.2 Ejercicios . . . . .	326
21.3.3 Variable aleatoria continua . . . . .	327
21.3.4 Ejercicios . . . . .	328
21.3.5 Función de distribución . . . . .	329
21.3.6 Función de distribución muestral o empírica . . . . .	330

21.3.7 Ejercicios . . . . .	330
21.4 Media y varianza . . . . .	331
21.4.1 Media de una variable aleatoria discreta . . . . .	331
21.4.2 Varianza y desviación típica . . . . .	333
21.5 Variable Bernoulli . . . . .	334
21.6 Variable binomial . . . . .	334
21.6.1 Ejercicios . . . . .	337
21.7 Distribución Poisson . . . . .	338
21.8 Distribuciones geométrica y binomial negativa . . . . .	340
21.8.1 Distribución geométrica . . . . .	340
21.8.2 Distribución binomial negativa . . . . .	340
21.9 Distribución normal . . . . .	342
21.10 Ejercicios . . . . .	346
<b>22 Distribución muestral . . . . .</b>	<b>347</b>
22.1 Población y muestra aleatoria . . . . .	347
22.2 Distribución muestral de una variable binomial . . . . .	347
22.2.1 Ejercicios . . . . .	349
22.3 Distribución muestral de la media bajo normalidad . . . . .	349
22.3.1 Ejercicios . . . . .	352
22.4 Distribución muestral de la media en poblaciones no normales. Teorema central del límite . . . . .	352
22.4.1 Aproximación de la distribución binomial . . . . .	352
22.4.2 Ilustración del teorema central del límite . . . . .	353
22.4.3 Ejercicios . . . . .	353
<b>23 Estimación . . . . .</b>	<b>355</b>
23.1 Introducción . . . . .	355
23.2 La población . . . . .	355
23.3 Estimación puntual . . . . .	356
23.4 Algunas definiciones . . . . .	358
23.5 Estimación puntual de la media . . . . .	358
23.6 Intervalo de confianza para la media . . . . .	358
23.6.1 Asumimos que conocemos la varianza . . . . .	358
23.6.2 No asumimos la varianza conocida . . . . .	361
23.7 Error absoluto y tamaño de la muestra . . . . .	367
23.8 Estimación de la varianza en poblaciones normales . . . . .	370
<b>24 Contraste de hipótesis . . . . .</b>	<b>373</b>
24.1 Introducción . . . . .	373
24.2 Contrastes para una muestra . . . . .	373
24.2.1 Un contraste unilateral . . . . .	373
24.2.2 Y, finalmente, el contraste bilateral . . . . .	379
24.3 Intervalo de confianza y contraste de hipótesis . . . . .	381
24.4 Contraste de normalidad . . . . .	382
24.4.1 Gráficos para evaluar la normalidad . . . . .	382
24.5 Contrastes de normalidad . . . . .	386
24.5.1 Test de Shapiro-Wilk . . . . .	386
24.5.2 Test ji-cuadrado . . . . .	387
24.5.3 Test de Kolmogorov-Smirnov . . . . .	387

<b>25 Comparación de dos poblaciones</b>	<b>389</b>
25.1 Introducción . . . . .	389
25.2 Comparación descriptiva de las muestras . . . . .	389
25.3 Comparando las medias de dos poblaciones normales . . .	393
25.3.1 Estimación de la diferencia de medias . . . . .	393
25.3.2 Contraste de hipótesis . . . . .	396
25.3.3 Los contrastes unilaterales o direccionales . . .	398
25.4 Inferencia sobre las varianzas de dos poblaciones normales	399
25.4.1 Estimación del cociente de varianzas . . . . .	399
25.4.2 Contraste de hipótesis para el cociente de varian- zas . . . . .	400
25.5 Comparación de medias con muestras apareadas . . . . .	401
25.6 Test de Kolmogorov-Smirnov para dos muestras . . . . .	404
25.7 Test de Wilcoxon . . . . .	408
<b>26 Datos categóricos</b>	<b>409</b>
26.1 Estimación de una proporción . . . . .	409
26.1.1 Ejercicios . . . . .	410
26.2 Tamaño de la muestra en la estimación de una proporción	411
26.2.1 Exercises . . . . .	412
26.3 Varias variables categóricas . . . . .	413
26.3.1 Probabilidad y tabla de contingencia . . . . .	413
26.3.2 Comparación de proporciones . . . . .	414
26.3.3 Test de Fisher . . . . .	416
<b>27 Modelos lineales</b>	<b>419</b>
27.1 Coeficiente de correlación de Pearson . . . . .	419
27.2 Matriz de covarianzas y matriz de correlaciones . . . . .	423
27.3 Regresión lineal simple . . . . .	424
27.4 Regresión lineal múltiple . . . . .	426
27.5 Estimación de $\beta$ . . . . .	427
27.6 Algunos casos particulares . . . . .	428
27.7 Verosimilitud . . . . .	429
27.8 Distribución muestral de $\hat{\beta}$ . . . . .	429
27.9 Bondad de ajuste . . . . .	430
27.10 Valoración de las hipótesis del modelo . . . . .	431
27.10.1 Homogeneidad de la varianza . . . . .	431
27.10.2 Normalidad . . . . .	434
27.10.3 Incorrelación de los errores . . . . .	439
27.10.4 Observaciones anómalas y observaciones influ- yentes . . . . .	440
27.11 Inferencia sobre el modelo . . . . .	443
27.12 Selección de variables . . . . .	447
27.12.1 Procedimientos que comparan modelos . . . . .	447
27.12.2 Procedimientos basados en criterios . . . . .	448
<b>28 Conceptos fundamentales de Estadística</b>	<b>449</b>
28.1 Verosimilitud . . . . .	449
28.2 Estimación . . . . .	451
28.2.1 Estimación insesgada de media y varianza . . .	451
28.2.2 Estimación insesgada del vector de medias y la matriz de covarianzas . . . . .	452
28.3 Estimador máximo verosímil . . . . .	453
28.4 Contraste de hipótesis . . . . .	455
28.4.1 Contraste de la media en la poblaciones normales	455

28.4.2 Test del cociente de verosimilitudes . . . . .	456
28.4.3 Test de Wald . . . . .	456
28.4.4 Intervalos de confianza . . . . .	457
<b>29 Modelos lineales generalizados</b>	<b>459</b>
29.1 Un modelo lineal generalizado . . . . .	459
29.2 Desviación . . . . .	461
29.3 Modelos lineales generalizados para datos binarios . . . . .	461
29.4 Modelo Poisson loglineal . . . . .	462
29.4.1 GLM Poisson e independencia en tablas de contingencia $I \times J$ . . . . .	463
29.5 Verosimilitud de modelos lineales generalizados . . . . .	463
29.5.1 Familia de dispersión exponencial . . . . .	463
29.6 Inferencia para modelos lineales generalizados . . . . .	465
29.6.1 Desviación y bondad de ajuste . . . . .	465
29.6.2 Comparación de modelos utilizando la desviación . . . . .	466
29.6.3 Residuos en un GLM . . . . .	467
29.6.4 Ajuste de un GLM mediante Newton-Raphson . . . . .	467
29.6.5 IRLS . . . . .	468
29.7 ML como mínimos cuadrados reponderados iterativamente	468
29.8 Cuasiverosimilitud Wedderburn(1974) . . . . .	469
29.8.1 Sobredispersión en GLM Poisson y cuasiverosimilitud . . . . .	470
<b>30 Meta-análisis</b>	<b>471</b>
30.1 Método de Fisher . . . . .	471
30.2 Método de Stouffer . . . . .	471
<b>31 Miseclánea</b>	<b>473</b>
31.1 Algoritmo bipesado de Tukey de un solo paso . . . . .	473
31.2 Median polish . . . . .	474
31.3 Datos faltantes . . . . .	476
31.3.1 Algoritmo de normalización del k-vecino más próximo . . . . .	476
31.4 Datos multimodales . . . . .	477
31.4.1 TCGA . . . . .	477
<b>VIII Investigación reproducible</b>	<b>479</b>
<b>32 Por favor, no publiquéis más</b>	<b>481</b>
32.1 Sobre la gran cantidad de publicaciones ¿innecesarias? .	481
32.2 Investigación reproducible . . . . .	482
32.3 Markdown . . . . .	483
32.4 RMarkdown . . . . .	483
32.5 RStudio y RMarkdown . . . . .	483
32.6 emacs y RMarkdown . . . . .	484
<b>33 Lo que no se debe hacer pero se hace</b>	<b>485</b>
33.1 Mejor no usarlo . . . . .	485
33.2 Combinando información de distintos experimentos . . .	486
<b>A Soluciones ejercicios</b>	<b>489</b>
<b>Glosario</b>	<b>515</b>



# Prólogo

**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 prerequisito 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 dispositivos 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.<sup>1</sup> 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 análisis en nuestro trabajo. También<sup>2</sup> 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 § 4 está dedicado a este tipo de datos, sus características y técnicas de preprocessado. 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.

**Sobre la parte biológica.** El objetivo es que estas notas ayuden a quien lee y no que demuestren que sabe el que las escribe.<sup>3</sup> 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, Bioquímica o Medicina fundamentalmente. Hay que tener interés en la Estadística y la Informática en cualquier caso para poder seguir las. En § ?? se incluye un repaso de procedimientos estadísticos de modo que podamos referenciarlos en el resto de temas según se utilizan. En el apéndice ?? se introduce lo básico del uso de R mientras que en apéndice ?? comentamos cómo definir funciones y proponemos ejemplos sencillos.

**Sobre R/Bioconductor.** Estas notas utilizan sistemáticamente R/Bioconductor. Y ningún otro software.<sup>4</sup>

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

<sup>1</sup> Uno no está obligado más que a hacer las cosas lo mejor que pueda y no más.

<sup>2</sup> En la medida en que me voy encontrando con nuevos tipos de datos que entienda.

<sup>3</sup> Sobre esta cuestión yo diría que que el autor se apaña como puede (que no es mucho) con los conceptos biológicos.

<sup>4</sup> En [73] podemos ver un trabajo interesante sobre las prestaciones de Bioconductor.

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 otras ocasiones proceden de empresas para las cuales les resulta interesante que se disponga de software que permita utilizar su hardware.

En lo que sigue hay distintos niveles de lectura en un mismo texto. Es un material que se utiliza en dos cursos claramente diferenciados.

**TAMI** Uno es una breve introducción en 20 horas lectivas en tercer curso del grado de Biotecnología 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.

**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.

**Sobre la investigación reproducible.** Es ingente la cantidad de publicaciones científicas que llevan tratamientos estadísticos. En muchas de ellas no es claro qué es lo que realmente han hecho. Citan un procedimiento estadístico 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<sup>5</sup> 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<sup>6</sup> y, de un modo más genérico, a la investigación reproducible<sup>7</sup>. R/Bioconductor incorpora muchas herramientas para realizar investigación reproducible.<sup>8</sup> En particular, este texto está realizado utilizando [159, 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 R/Bioconductor o, simplemente, porque son bonitos de estudiar.

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

**Sobre la generación simple de informes.** 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.

<sup>5</sup> 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.

<sup>6</sup> [http://en.wikipedia.org/wiki/Literate\\_programming](http://en.wikipedia.org/wiki/Literate_programming)

<sup>7</sup> <http://en.wikipedia.org/wiki/Reproducibility>

<sup>8</sup> <http://cran.r-project.org/web/views/ReproducibleResearch.html>

**Cómo usar el texto.** 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.<sup>9</sup>

**¿Bioconductor es la única opción?** Por supuesto que no. Y tampoco tengo muy claro que sea la mejor. Indudablemente es una de las mejores. En estas notas el énfasis está en el análisis de datos y, en particular, en el análisis estadístico de datos. Y desde el punto de vista de la Estadística sí que podemos afirmar que R es la mejor opción y Bioconductor es su opción natural.<sup>10</sup> En este texto utilizamos R/Bioconductor. Es muy recomendable consultar la dirección <http://www.bioconductor.org/help/workflows/> en donde podemos encontrar análisis completos de datos.

**¿Qué necesitamos instalar para poder utilizar estas notas?** En <http://www.uv.es/ayala/docencia/tami/AllTami.R> tenemos el modo de instalar todos los paquetes que se utilizan en estas notas.<sup>11</sup>

**Funciones y paquetes.** A lo largo de las notas utilizamos muchas funciones que corresponden a paquetes distintos. Una función no puede ser utilizada si no hemos cargado previamente el paquete correspondiente. Si intentamos utilizar la función sin cargar previamente el paquete entonces R nos indica que no la encuentra. Con frecuencia indicaremos el nombre de la función y su paquete conjuntamente. Por ejemplo, si en el texto vemos `annotate::annotation` estamos refiriéndonos a la función `annotation` que se encuentra en el paquete [54, `annotate`].

**Sobre la estructura del documento.** Empezamos en I indicando qué se entiende por Estadística de datos ómicos y con una breve introducción a R. Acabamos esta parte hablando de anotación. Necesitamos tener bases de datos (locales u online) que nos permitan saber qué a qué se refieren nuestras características (variables). ¿Qué gen o qué transcripto o qué exón tengo? ¿Qué relaciones hay entre ellos? Por ejemplo: ¿qué exones componen un exon? O bien: ¿qué isoformas tengo de un gen?

En II 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 el paquete [11, `tamidata`].<sup>12</sup> En el momento actual consideramos datos de expresión obtenidos con microarrays y con la técnica RNASeq.<sup>13</sup>

Expresión diferencial, III, es el tema fundamental desde el punto de vista estadístico que es la opción de este manual. Se estudian los problemas de comparaciones múltiples así como técnicas de expresión diferencial en microarrays y RNASeq.

En IV hablamos de componentes principales y clustering. Desde mi (muy) personal punto de vista no son técnicas multivariantes que deban

<sup>9</sup> Solamente tener en cuenta que los caminos hay que modificarlos adaptándolos a donde tengamos los datos en nuestro ordenador.

<sup>10</sup> Sin embargo, en [http://neurolex.org/wiki/Category:Resource:Gene\\_Ontology\\_Tools](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.

<sup>11</sup> 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.

<sup>12</sup> Es un paquete propio que lo aloja en una página de la Universidad de Valencia.

<sup>13</sup> Aunque se pretende ampliar a otros datos ómicos.

<sup>14</sup> Faltaría un tema de clasificación supervisada que no acabo de ver su interés en este contexto.

de usar en exceso en este contexto. Sin embargo, se usan y por ello se tratan aquí.<sup>14</sup>

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

<sup>15</sup> El material se desarrolló en origen para esta asignatura del grado de Biotecnología de la Universidad de Valencia. Se imparte en 25 horas.

Este manual se utiliza para distintos cursos. El primero es **Tecnologías de análisis molecular integrado**.<sup>15</sup> Se compone de los siguientes apartados indicando su contenido.

1. § 1 presenta el problema del análisis de datos ómicos.
2. En § 2 se considera la instalación de R así como la instalación de los paquetes que utilizamos en este manual. Se han de consultar las secciones de § 2.1 hasta § 2.5.
3. En § 4 se trata de datos ómicos obtenidos utilizando arrays. Nos centramos en el caso de arrays de Affymetrix. Hay que leer de § 4.1 a § 4.5 aunque el código no es necesario utilizarlo.
4. En § 5 se comentan los datos que utilizamos en el resto del manual y muchos de ellos están incluidos en [11, tamidata]. También se muestra cómo trabajar con un **ExpressionSet**. Las secciones que utilizamos son § 5.2, § 5.6 y § 5.7.
5. De § 7 se utilizan las secciones § 7.1, § 7.2, § 7.4, § 7.6 y § 7.7.
6. De § 8 utilizamos § 8.1, § 8.3, § 8.4. De § 8.5.1 solamente el método de Bonferroni. De § 8.6 solamente el método de Benjamini-Hochberg. Secciones § 8.7 y § 8.8.
7. De § 16 utilizamos § 16.1 y § 16.2.

A todo el material anterior hay que añadir las viñetas incluidas en el paquete [10]. Estas viñetas cubren el material de uso de R/Bioconductor.

El segundo curso que se imparte con estas notas es **Bioinformática estadística**<sup>16</sup> y utilizamos todo el material.

<sup>16</sup> Se imparte en 45 horas dentro del máster de Bioinformática de la Universidad de Valencia.



# **Parte I**

# **Introducción**



# Capítulo 1

## Estadística y datos ómicos

Estas notas se ocupan de revisar y aplicar técnicas estadísticas a datos ómicos. En lo fundamental nos ocuparemos de datos de transcripción y proteomas. 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**. En este sentido es interesante recordar una frase (que subscribo) de [Brian D. Ripley](#).

```
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)
```

### 1.1 Estructura de los datos

En lo que sigue tendremos distintos tipos de datos pero con una estructura similar. En concreto, observaremos un gran número de características<sup>17</sup> sobre un pequeño número de muestras. Cada característica medirá la abundancia de moléculas utilizando distintos procedimientos. Esta abundancia 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, a un péptido, a una proteína. Además esta abundancia 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<sup>18</sup> 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

<sup>17</sup> Features.

<sup>18</sup> Y a lo largo de las notas lo repetiremos muchas veces.

Estadística de alto rendimiento esto no es así. Y eso da novedad a los procedimientos. Obviamente limita las posibilidades pero abre un nuevo campo de trabajo.

<sup>19</sup> Que podemos llamar matriz de expresión aunque no necesariamente hablamos de expresión de un gen pero no parece malo utilizar esta nomenclatura.

<sup>20</sup> 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.

<sup>21</sup> Algunos procedimientos de procesado previo de los datos conocidos como normalización pueden dar lugar a expresiones negativas.

<sup>22</sup> Expression profile.

<sup>23</sup> Lo cual no deja de ser paradójico.

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

<sup>25</sup> Entendido en un sentido amplio: Por variable fenotípica podemos estar entendiendo el tiempo en que se ha observado la evolución de una muestra.

<sup>26</sup> Los valores 0 y 1 son arbitrarios. Podemos tomar cualquier otro par de valores.

Las abundancias las recogemos en una matriz<sup>19</sup> que denotaremos por

$$x = [x_{ij}]_{i,j=1,\dots,n}$$

donde el valor  $x_{ij}$  nos cuantifica la abundancia de la característica  $i$  en la muestra  $j$ .<sup>20</sup> El valor  $x_{ij}$  será un valor (usualmente) positivo. Si corresponde un DNA microarray entonces mide un nivel de fluorescencia y tomará valores positivos.<sup>21</sup> Sea positivo a no un valor mayor indicará una mayor expresión del gen. Si trabajamos con datos obtenidos con RNASeq 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 un *perfil*<sup>22</sup> (de un modo genérico perfil de expresión).

En la matriz de expresión los valores observados para las distintas muestras son *independientes* aunque posiblemente observados bajo distintas circunstancias. No son pues réplicas de una misma condición experimental pero sí se observan independientemente. Los valores de expresión para las distintas filas ya no son independientes. Por ejemplo, los genes actúan de un modo coordinado.

Los datos de la matriz de expresión no son directamente comparables. El nivel de ruido es grande y por ello se han desarrollado técnicas para corregirlo. Son métodos de corrección de fondo y normalización. Veremos algunos. En sentido estricto cuando normalizamos 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.<sup>23</sup>

De cada muestra tendremos información. Por ejemplo, si es una muestra control o bien corresponde a una muestra tomada bajo un tratamiento.<sup>24</sup> A esta información o variables que nos describen a las muestras las llamaremos los *metadatos* o *variable fenotípicas*<sup>25</sup> Usualmente tendremos varias variables fenotípicas aunque usualmente trabajaremos con una cada vez. Denotaremos por  $y = (y_1, \dots, y_n)$  los valores observados de una variable en las  $n$  muestras. El caso más frecuente de variable fenotípica será cuando tengamos dos grupos de muestras (casos y controles). En este caso tendremos  $y_i = 1$  si es un caso e  $y_i = 0$  si es un control.<sup>26</sup> Si tenemos más de dos grupos de muestras a comparar, por ejemplo  $k$  grupos, entonces  $y_i \in \{1, \dots, k\}$  para  $i = 1, \dots, n$ .

## 1.2 Análisis de datos

¿Y qué vamos a hacer con la matriz de expresión 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.

El problema más importante es el que se conoce como **expresión diferencial**. Nos fijamos en una variable fenotípica y en una característica (gen por ejemplo). ¿Hay asociación entre el perfil de expresión 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.<sup>27</sup>

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 los 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.

<sup>27</sup> Este problema puede encontrarse bajo distintas denominaciones como: Gene set analysis, gene set enrichment analysis, functional enrichment testing.

### 1.3 Sobre herramientas online

En este texto la opción elegida es R/Bioconductor. Una opción alternativa<sup>28</sup> es la utilización de aplicaciones online. Una buena opción que cubre muchos de los contenidos de este manual es [157, NetworkAnalyst] y la aplicación se localiza en <http://www.networkanalyst.ca>. En particular es de interés la tabla 1 en [157, pág. 824] en donde se muestra una comparativa entre distintas herramientas online. En particular se consideran:

<sup>28</sup> Que puede ser complementaria.

**NetworkAnalyst** <http://www.networkanalyst.ca>.

**DAVID** <http://david.abcc.ncifcrf.gov>.

**g:Profiler** <http://biit.cs.ut.ee/gprofiler/>.

**InnateDB con Cytoscape** <http://www.innatedb.ca> y <http://www.cytoscape.org>.

**Gitoools** <http://www.gitoools.org>.

## 1.4 Paquetes transversales

Algunos paquetes que vamos a manejar pretenden cubrir una gran cantidad de contenidos que vamos a tratar: lectura de la información generando las clases adecuadas (como Biobase::ExpressionSet o SummarizedExperiment::RangedSummarizedExperiment); normalización; análisis de expresión diferencial marginal o gen a gen; análisis de enriquecimiento para grupos o bien para grafos. En fin, tienen una pretensión de análisis global. Usualmente lo que hacen es utilizar distintos paquetes según realiza una u otra funcionalidad. Dentro de esta categoría entran [52, EnrichmentBrowser] o [142, piano]. Los usaremos aunque siempre suponen una limitación en el análisis que se realiza. Soy más partidario de utilizar funcionalidades concretas del paquete más que utilizarlo para todo el análisis.

## 1.5 Jerga

En un campo como es la Bioinformática un problema mayor es el lenguaje que se utiliza. Un texto de Estadística o un texto de Genética tiene una jerga consolidada. Son campos de trabajo maduros.<sup>29</sup> Prácticamente cada campo de investigación va desarrollando una jerga<sup>30</sup> en donde suele haber una parte interesada. Impedir el acceso a los que no son de este campo. Aquí el problema es grave porque se mezclan jergas al mismo tiempo que se desarrolla una nueva. Para intentar evitar esta barrera he incluido un glosario al final del texto que intenta aclarar los términos y sirva de referencia rápida. Sin embargo, quizás la mayor dificultad de esta disciplina sea este, entender la jerga de los demás.<sup>31</sup>

<sup>29</sup> Sin ser peyorativo, campos de investigación viejos.

<sup>30</sup> Mejor una jerigonza.

<sup>31</sup> En mi caso de los biólogos. A los informáticos los entiendo. Hablan menos de hecho.

## 1.6 Datos ordenados

Un problema que comparte el análisis de datos ómicos con cualquier otro problema de análisis de datos es tener unos datos de entrada *ordenados*<sup>32</sup> así como unas salidas de las distintas funciones que los analizan que sean ordenadas, fácilmente interpretables y que puedan ser una entrada ordenada para un procedimiento posterior. El término utilizado en [149] es **tidy data**. En este texto utilizaremos distintas herramientas que prestan atención a este aspecto del análisis que ahorra mucho tiempo de trabajo y muchísimos quebraderos de cabeza. Entre otros paquetes utilizaremos [13, biobroom] y [118, broom].

## 1.7 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.<sup>33</sup> Un libro que

<sup>33</sup> Hay una peligrosa tendencia a creer que leyendo un resumen se conoce la técnica. No es cierto. Las referencias bibliográficas siempre hay que consultarlas.

trata cómo hacer las cosas con R/Bioconductor pero no lo que se hace o porqué se hace es [126]. Es una guía de uso muy bien elaborada. Un texto con un objetivo similar al nuestro es [64]. Un manual online que sigue una línea muy similar a este es <http://genomicsclass.github.io/book/>.

# Capítulo 2

## R y Bioconductor

Nuestra opción de trabajo es la utilización de R y Bioconductor. En adelante, hablaremos de R/Bioconductor. No es la única opción. Dos buenas opciones de software libre son Babelomics o DAVID Bioinformatics. Y una opción comercial es Partek Genomics Suite. Sin embargo, cuando hablamos de análisis estadístico de datos de alto rendimiento sí que, en el momento actual, es en mi opinión la mejor opción. Y en este manual sobre todo hablamos de análisis de datos.

### 2.1 Sobre R y su instalación

#### 2.1.1 De cómo instalarlo

**Instalación** Los pasos a seguir son los siguientes:

1. Bajamos el programa de la siguiente dirección <http://cran.r-project.org/>.
2. Elegimos versión.<sup>34</sup>
3. Una vez hemos bajado el paquete se instala ejecutándolo con las opciones por defecto.

**Inicio de una sesión** En el escritorio tenemos el icono de R. Simplemente clicando<sup>1</sup> el icono iniciamos la sesión de trabajo.

<sup>34</sup> Se recomienda cualquier versión de Linux. Si utilizas Windows (en esto el mundo es bastante libre) elige la correspondiente versión para Windows.

**Instalación de un paquete** R tiene muchos paquetes que extienden el R base que acabamos de instalar. De hecho, es casi imposible realizar un análisis estadístico por sencillo que sea sin utilizar paquetes adicionales de R. Vamos a instalar el paquete [144, UsingR]. Es un paquete con herramientas para la enseñanza de Estadística básica. La opción más simple es escribir en línea de comandos.

```
install.packages(``UsingR'')
```

**Cargando un paquete** Una vez instalado el paquete, para poder usar las funciones o datos que contenga, debemos *cargarlo* mediante

```
library(UsingR)
```

<sup>1</sup> Dios nos perdone por usar esta palabra aunque no creo que lo haga.

Ahora podemos utilizar las extensiones que proporciona a R este paquete.

A lo largo de estas notas iremos aprendiendo a utilizar (lo que necesitamos de) R según lo necesitemos. Sin embargo, una referencia rápida a la sintaxis de R la podemos encontrar en <http://cran.r-project.org/doc/contrib/Short-refcard.pdf>.<sup>2</sup>

Otra opción para instalar es utilizar [117, pacman]. La instalación

<sup>35</sup> Elegimos como repositorio <https://ftp.cixug.es/>

de [144, UsingR] sería<sup>35</sup>

```
pacman::p_install("UsingR", repos = "https://ftp.cixug.es/CRAN/")
```

## 2.2 Lo primero con R

### De cómo trabajar con R

Hay dos formas de trabajar con R. La primera opción es utilizar un editor en el que escribimos código de R. Lo copiamos y luego lo pegamos en la línea de comandos de R.<sup>3</sup> Dentro de esta manera de trabajar podemos utilizar distintas editores (que llevan herramientas para facilitarnos el trabajo).

**Con el editor de R** El propio programa lleva un editor incorporado. Es básico pero suficiente. Es la opción que utilizaremos en las clases prácticas.

**RStudio** Quizás ahora mismo sea la mejor opción. Es un programa que incorpora el editor, nos muestra las salidas, los gráficos y la historia previa. De manejo muy simple. <http://rstudio.org/>.

### De cómo conseguir ayuda con R

Supongamos que buscamos ayuda sobre las opciones de la función que nos dibuja un histograma, `hist`. Lo más simple es utilizar la ayuda en html. Utilizamos la siguiente función.

```
help.start()
```

Vemos que nos abre el navegador y nos ofrece distintas opciones. Quizás la opción más simple sea utilizar la herramienta de búsqueda. Otra opción es

```
?hist
```

O simplemente,

```
help(hist)
```

<sup>2</sup>Una buena idea es imprimirla y tenerla a mano.

<sup>3</sup>Es la opción más educativa en donde aprendemos realmente a trabajar con el programa.

## 2.3 Una sola muestra

Supongamos que consideramos una técnica ómica (microarray, RNA-seq). Estamos estudiando la expresión de 10 genes y los valores observados son los siguientes.

```
## [1] 3 6 3 6 7 1 4 7 4 4
```

Podemos leer estos datos con

```
x = c(3,6,3,6,7,1,4,7,4,4)
```

Tenemos ahora un objeto llamado x

```
class(x)
```

```
## [1] "numeric"
```

que es un vector formado por números. ¿Cómo puede ver su contenido?

```
x
## [1] 3 6 3 6 7 1 4 7 4 4
```

¿Cuál es el primer valor de este vector de datos?

```
x[1]
```

```
## [1] 3
```

¿Y el que ocupa la posición 7?

```
x[7]
```

```
## [1] 4
```

Podemos ver los datos que están entre el 3 y el 7. Para ello fijémonos en el siguiente código.

```
3:7
```

```
## [1] 3 4 5 6 7
```

Cuando ponemos dos enteros separados por : nos devuelve todos los enteros entre el primero y el segundo. Lo mismo lo podemos hacer con

```
seq(3,7,1)
```

```
## [1] 3 4 5 6 7
```

Ahora podemos ver los datos que ocupan estas posiciones en el vector x.

```
x[3:7]
```

```
## [1] 3 6 7 1 4
```

Podemos tener interés en saber los valores de los datos que ocupan las posiciones 1, 3 y de la 5 a la 8. Estas posiciones las podemos obtener con

```
c(1,3,5:8)
```

```
## [1] 1 3 5 6 7 8
```

y los valores de x serían

```
x[c(1,3,5:8)]
```

```
## [1] 3 3 7 1 4 7
```

Puede que nuestro interés en ver los datos no venga dado por la posición que ocupan sino por su valor. Por ejemplo, queremos saber cuántos de estos datos superan o son iguales a 4. ¿Cómo lo hacemos? Lo lógico es comparar los valores de x con 4. Lo hacemos con

```
x >= 4
```

```
## [1] FALSE TRUE FALSE TRUE TRUE FALSE TRUE
## [8] TRUE TRUE TRUE
```

Vemos que nos devuelve un vector diciéndonos si es cierta o no la condición que hemos preguntado, si es mayor o igual a 4. Pero: ¿qué valores son? Si hacemos

```
x[x >= 4]
```

```
## [1] 6 6 7 4 7 4 4
```

Nos devuelve los datos que ocupan las posiciones donde se daba la condición, donde la condición era cierta. Podemos saber qué valores toman los datos que son mayores que 37 con

```
x[x > 6]
```

```
## [1] 7 7
```

o bien los datos que son mayores que 4 y menores o iguales que 6.

```
x[x > 4 & x <= 6]
```

```
## [1] 6 6
```

Podemos querer que los casos que estamos seleccionando estén en un nuevo vector.

```
y = x[x > 4 & x <= 6]
```

Los valores de y son

```
y
## [1] 6 6
```

Podemos querer los que son menores o iguales a 4 o mayores que 6.

```
x[x <= 4 | x > 6]
## [1] 3 3 7 1 4 7 4 4
```

¿Y los que son iguales a 4? Observar el uso de ==.

```
x[x == 4]
## [1] 4 4 4
```

¿Y en qué posiciones del vector tenemos los valores iguales a 4.

```
which(x == 4)
## [1] 7 9 10
```

36

**De cómo ordenar un vector** Supongamos que queremos ordenar el vector  $x$ .

```
sort(x)
## [1] 1 3 3 4 4 4 6 6 7 7
```

Nos devuelve los valores ordenados. Sin embargo, con frecuencia necesitamos saber la posición que ocupaban en el vector original estos valores.

```
sort(x, index.return = TRUE)
## $x
## [1] 1 3 3 4 4 4 6 6 7 7
##
## $ix
## [1] 6 1 3 7 9 10 2 4 5 8
```

Hemos ordenado el vector  $x$  de menor a mayor. Vamos a ordenar de mayor a menor.

```
sort(x, decreasing = TRUE, index.return = TRUE)
## $x
## [1] 7 7 6 6 4 4 4 3 3 1
##
## $ix
## [1] 5 8 2 4 7 9 10 1 3 6
```

<sup>36</sup> El resto de operadores lógicos los encontramos en la ayuda de *Logical Operators*: & (&&) corresponde con la intersección, | (||) corresponde con la unión y la negación de una condición la obtenemos con !.

\* Ej. 1 — Introducir los siguientes datos en R.

```
## [1] 9 5 6 5 2 8 3 2 4 9 8 6 6 11 6
## [16] 6 5 5 4 3 9 8 6 7 1 5 6 3 4 3
## [31] 3 4 4 4 3 3 3 5 4 7 2 5 7 2 5
## [46] 3 3 6 8 4 6 2 4 4 7 5 7 7 7 4
## [61] 6 6 6 0 5 3 4 5 4 2 3 6 4 7 2
## [76] 4 10 8 8 3 3 6 4 6 4 3 7 2 5 5
## [91] 5 4 5 9 5 8 8 5 4 3
```

Se pide:

- 1.¿Cuál es el valor medio, la varianza y la desviación estándar de estos valores?
- 2.¿Cuántos de los valores son 3?
- 3.¿Cuál es el rango intercuartílico?
- 4.Ordena el vector y calcula las sumas acumuladas de los valores ordenados con la función base::cumsum.

<sup>37</sup> Estamos generando valores con distribución normal con media 45 y desviación estándar 2.3. Simplemente son unos datos para trabajar con ellos.

\* Ej. 2 — Ejecutad el siguiente código.<sup>37</sup>

```
x = rnorm(23489,mean=45,sd=2.3)
```

Se pide:

- 1.¿Cuántos valores de x son mayores o iguales de 39.4?
- 2.¿Cuántos valores de x son estrictamente menores que 46?
- 3.¿Cuántos valores de x son estrictamente menores que 46 y mayores o iguales de 39.4?
- 4.¿Cuántos valores son tales que su parte entera (por defecto) es igual a 40? Se puede utilizar también la función base::floor.

### 2.3.1 Varias muestras

Obviamente no tendremos habitualmente una sola muestra sino que tendremos más de una muestra. En cada una de ellas tendremos observada la expresión de los mismos genes. Supongamos que tenemos la expresión 10 genes en 4 muestras. Simulamos estos datos.

```
set.seed(123)
(x = matrix(rpois(10*4,lambda=5),nrow=10))

##      [,1] [,2] [,3] [,4]
## [1,]     4    9    8    9
## [2,]     7    5    6    8
## [3,]     4    6    6    6
## [4,]     8    5   11    7
## [5,]     9    2    6    1
## [6,]     2    8    6    5
## [7,]     5    3    5    6
## [8,]     8    2    5    3
## [9,]     5    4    4    4
## [10,]    5    9    3    3
```

¿Qué tipo de dato es x?

```
class(x)
## [1] "matrix"
```

Es una matriz. Podemos darle nombre a sus filas y columnas.

```
colnames(x) = c("sample1", "sample2", "sample3", "sample4")
```

<sup>38</sup> El siguiente código es equivalente al anterior y más simple.

```
colnames(x) = paste0("sample", 1:4)
```

Además sabemos que en cada fila tenemos la expresión correspondiente a un gen distinto. De momento, supondremos que son “gene1”, “gene2”, etc. Vamos a cambiarle el nombre las filas utilizando el código que acabamos de ver.

```
rownames(x) = paste0("gene", 1:10)
```

¿Qué tenemos ahora?

```
x
##      sample1 sample2 sample3 sample4
## gene1      4      9      8      9
## gene2      7      5      6      8
## gene3      4      6      6      6
## gene4      8      5     11      7
## gene5      9      2      6      1
## gene6      2      8      6      5
## gene7      5      3      5      6
## gene8      8      2      5      3
## gene9      5      4      4      4
## gene10     5      9      3      3
```

Podemos ver la expresión del gen en la fila 6 y en la columna 2 con

```
x[6, 2]
## [1] 8
```

o bien con

```
x["gene6", "sample2"]
## [1] 8
```

Si queremos ver la expresión de los genes en la muestra 2 podemos hacerlo con

```
x[, 2]
##   gene1  gene2  gene3  gene4  gene5  gene6  gene7
##      9      5      6      5      2      8      3
##   gene8  gene9 gene10
##      2      4      9
```

<sup>38</sup> La función base::c simplemente construye un vector y en cada posición tiene el nombre de la muestra. Podemos concatenar también números.

o

```
x[, "sample2"]

##   gene1   gene2   gene3   gene4   gene5   gene6   gene7
##     9      5      6      5      2      8      3
##   gene8   gene9  gene10
##     2      4      9
```

**Ej. 3** — Ejecutad el siguiente código. Nos genera una matriz donde los valores siguen una distribución normal con media 34 y desviación estándar 3.21.

```
x = matrix(rnorm(2345*122, mean=34, sd=3.21), nrow=2345)
```

Se pide:

1. ¿Cuántas columnas tiene la matriz?
2. ¿Qué columna tiene la media más alta? Utilizad las funciones base::apply y base::which.max.
3. ¿Qué fila tiene la media más pequeña? Utilizad las funciones base::apply y base::which.min.
4. ¿Qué fila tiene el menor rango intercuartílico? Utilizad las funciones base::apply y base::which.min.
5. ¿En cuántas ocasiones la primera columna es mayor o igual que la segunda columna de la matriz?
6. Determinamos la media de cada una de las filas. ¿Cuántas filas son tales que la columna 45 es mayor o igual que el vector de medias que acabamos de calcular?

## 2.4 Datos golub

Los datos golub son banco de datos de expresión de genes que vamos a utilizar. Aparecen en `multtest::golub` y fueron utilizados en [62]. Los datos son los niveles de expresión de 3051 genes (que aparecen en filas) para 38 pacientes de leucemia. De estos pacientes, 27 de ellos tienen leucemia linfoblástica aguda (ALL) y los restantes 11 tienen leucemia mieloide aguda (AML). El tipo de tumor viene indicado por el vector `golub.cl` donde ALL corresponde con 0 y AML corresponde con 1. Los nombres de los genes los tenemos en `golub.gnames`. Cargamos los datos.

```
data(golub, package = "multtest")
```

Al cargar estos datos hemos leído varias cosas:

**golub.cl** Es un vector numérico que toma valores 0 y 1 indicando si la muestra ha sido obtenida de un enfermo con leucemia linfoblástica aguda (0) o leucemia mieloide aguda (1).

**golub** Es una matriz de expresión donde cada fila corresponde con un gen y cada columna con una muestra distinta.

**golub.gnames** Es una matriz con el mismo número de filas de la matriz `golub` de modo que la fila i-ésima de `golub.gnames` nos da información sobre el gen de cuyos niveles de expresión aparecen en la fila i-ésima de la matriz `golub`.

Empezamos jugando con el vector `golub.cl`. Podemos ver los valores.

```
golub.cl
## [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [24] 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

¿Qué tipo de dato tenemos?

```
class(golub.cl)
## [1] "numeric"
```

Vemos que son números. Son números que, realmente, indican la pertenencia a un tipo de cáncer u otro. Por ello es más natural definirlo como un vector tipo `factor`.<sup>39</sup> Lo hacemos con `base::factor`.

<sup>39</sup> Como en R codificamos las variables categóricas.

```
(golub.fac = factor(golub.cl, levels=0:1, labels=c("ALL", "AML")))
## [1] ALL ALL
## [12] ALL ALL
## [23] ALL ALL ALL ALL AML AML AML AML AML AML AML AML
## [34] AML AML AML AML AML
## Levels: ALL AML
```

¿Cuántas muestras tenemos de cada tipo? Un par de opciones para hacerlo son utilizando la función `base::table`.

```
table(golub.fac)
## golub.fac
## ALL AML
## 27 11
```

o utilizando la función `base::summary`.

```
summary(golub.fac)
## ALL AML
## 27 11
```

Si queremos mostrar la distribución de los dos tipos en forma de diagrama de barras lo podemos hacer con (ver figura 2.1)

```
pacman::p_load(ggplot2)
df0 = data.frame(golub.fac)
png(paste0(dirTamiFigures, "RBio39.png"))
ggplot(df0, aes(x=golub.fac))+geom_bar()
dev.off()
```

Sustituye en el código anterior `golub.fac` por `golub.cl` y verás que el resultado no es el deseable. ¿Por qué? Quizás los números no es una buena forma de representar categorías o clases.

Nos fijamos en `golub`. ¿Qué tipo de dato es?

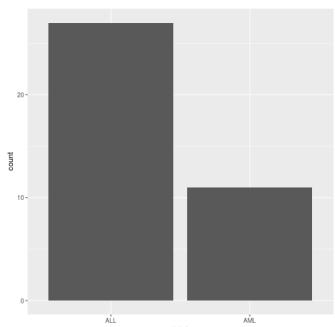


Figura 2.1: Diagrama de barras con las frecuencias absolutas de cada tipo de leucemia.

```
class(golub)
## [1] "matrix"
```

Es una matriz. En una matriz lo primero es saber cuántas filas (características aquí) tenemos

```
nrow(golub)
## [1] 3051
```

El número de columnas corresponde con el número de muestras.

```
ncol(golub)
## [1] 38
```

También podemos obtener las dimensiones de la matriz conjuntamente con

```
dim(golub)
## [1] 3051 38
```

En la fila  $i$  y columna  $j$  tiene una cuantificación de la expresión del gen  $i$ -ésimo<sup>40</sup> en la  $j$ -ésima muestra de nuestra experimentación. Por qué gen estamos hablando.

<sup>40</sup> Ya veremos más adelante que gen  $i$ -ésimo es en la  $j$ -ésima muestra de nuestra experimentación. Por ejemplo, el gen en fila 2000 y columna 12 tiene expresión

```
golub[2000,12]
## [1] 0.19595
```

<sup>41</sup> Estos datos han sido preprocesados partiendo de los datos originales. Pueden aparecer valores negativos. En § 5.3 se explica y se entiende el porqué del valor negativo. Lo importante valor menor indica menor expresión, valor mayor indica mayor expresión.

<sup>41</sup> Y todos los niveles de este gen lo tendremos con

```
golub[2000,]
## [1] 0.73124 -0.19598  0.51981 -0.54371  0.55596
## [6] 1.40683  0.79772  0.59493  0.99503  0.39529
## [11] 0.09834  0.19595  0.85017 -1.39979  1.09789
## [16] -0.74362  0.44207  0.27698 -0.04128 -1.60767
## [21] -1.06221 -1.12665  0.47863 -0.44014  0.22286
## [26]  0.42795  0.65427  0.07257 -0.28093 -0.20985
## [31]  0.05160 -1.44434 -0.17118 -1.34158  0.92325
## [36] -0.21462 -1.34579  1.17048
```

<sup>42</sup> Expression profile.

Estos valores reciben el nombre de *perfil de expresión*.<sup>42</sup> Podemos representar el perfil de expresión (figura 2.2). Definimos qué van en el eje de abscisas y en el eje de ordenadas.

```
pacman::p_load(ggplot2)
muestra = 1:ncol(golub) ## En abscisas el número de la muestra
y2000 = golub[2000,] ## En ordenadas su expresión
df = data.frame(muestra = 1:ncol(golub), y2000 = golub[2000,])
png(paste0(dirTamiFigures, "RBio47.png"))
ggplot(df, aes(x = muestra, y = y2000)) + geom_point()
dev.off()
```

Lo podemos ver en figura 2.2(a).

Si queremos ver las expresiones correspondientes a los pacientes ALL lo podemos hacer con <sup>43</sup>

```
golub[2000,golub.fac == "ALL"]  
  
## [1] 0.73124 -0.19598 0.51981 -0.54371 0.55596  
## [6] 1.40683 0.79772 0.59493 0.99503 0.39529  
## [11] 0.09834 0.19595 0.85017 -1.39979 1.09789  
## [16] -0.74362 0.44207 0.27698 -0.04128 -1.60767  
## [21] -1.06221 -1.12665 0.47863 -0.44014 0.22286  
## [26] 0.42795 0.65427
```

<sup>43</sup> Observemos que “==” se utiliza para comparar e indica TRUE O FALSE según sea cierta o falsa la condición.

y los correspondientes a los pacientes AML.

```
golub[2000,golub.fac == "AML"]  
  
## [1] 0.07257 -0.28093 -0.20985 0.05160 -1.44434  
## [6] -0.17118 -1.34158 0.92325 -0.21462 -1.34579  
## [11] 1.17048
```

Supongamos que volvemos a representar el perfil de expresión del gen en la fila 2000, y2000, pero pretendemos diferenciar con un color distinto la condición que nos indica el tipo de leucemia, golub.fac. Lo podemos hacer con (figura 2.2(b)):

```
df = data.frame(muestra = 1:ncol(golub),y2000= golub[2000,],  
                tipo = golub.fac)  
png(paste0(dirTamiFigures,"RBio53.png"))  
ggplot(df,aes(x = muestra,y= y2000,color=tipo))+geom_point()  
dev.off()
```

Otra opción sería representar separadamente<sup>44</sup> los datos correspondientes a cada tipo de leucemia (figura 2.2(c)). También podemos hacerlo con

<sup>44</sup> En facets distintas.

```
df = data.frame(muestra = 1:ncol(golub),y2000= golub[2000,],  
                tipo = golub.fac)  
png(paste0(dirTamiFigures,"RBio56.png"))  
ggplot(df,aes(x = muestra,y= y2000,colour=tipo))+geom_point() +  
  xlab("Número de muestra") + ylab("Gen en fila 2000") +  
  facet_grid(tipo~.)  
dev.off()
```

Podemos colocar los distintos subdibujos (o facets) horizontalmente (2.2(d)).

```
df = data.frame(muestra = 1:ncol(golub),y2000= golub[2000,],  
                tipo = golub.fac)  
png(paste0(dirTamiFigures,"RBio57.png"))  
ggplot(df,aes(x = muestra,y= y2000,colour=tipo))+geom_point() +  
  xlab("Número de muestra") + ylab("Gen en fila 2000") +  
  facet_grid(. ~ tipo) #Modificamos esta línea  
dev.off()
```

En gráficos (en todo) la simplicidad es un valor. De las tres figuras anteriores quizás la figura 2.2(b) sea la más adecuada.

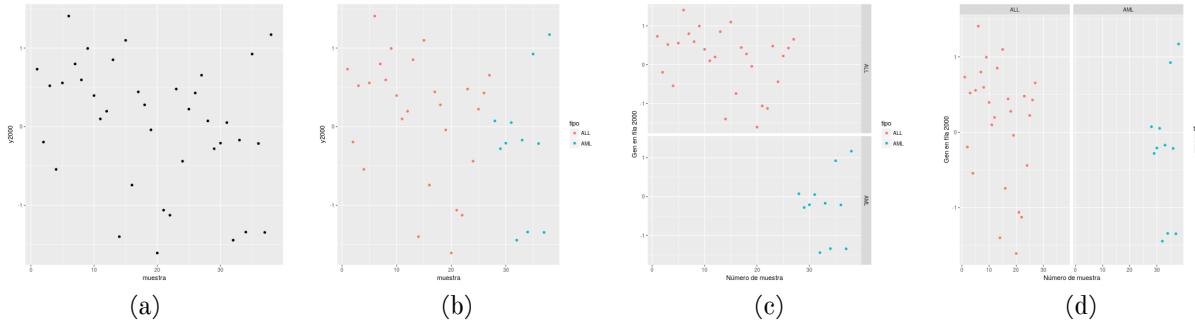


Figura 2.2: (a) Perfil de expresión del gen en la fila 2000. (b) Perfil de expresión del gen en fila 2000 diferenciando el tipo de leucemia. (c) Perfil de expresión del gen en fila 2000 diferenciando el tipo de leucemia mediante un color distinto. Utilizamos distintos dibujos (facets) alineados en vertical. (d) Perfil de expresión del gen en fila 2000 diferenciando el tipo de leucemia mediante un color distinto. Utilizamos distintos dibujos (facets) alineados en vertical.

**¿De qué genes estamos hablando? ¿A qué gen corresponde esta fila?**  
La información la podemos obtener con

```
golub.gnames[2000,]
```

¿Qué tipo de datos tenemos ahora?

```
class(golub.gnames)
## [1] "matrix"
```

Es una matriz con dimensiones

```
dim(golub.gnames)
## [1] 3051      3
```

Cada fila nos da información con el gen del cual tenemos su perfil de expresión en la matriz golub. En concreto, ¿qué tenemos en la fila 2000?

```
golub.gnames[2000,]
## [1] "4544"           "CDC21_HOMOLOG"
## [3] "X74794_at"
```

Los datos originales de la experimentación tenían más genes estudiados. Se hizo una selección ([§ 5.3](#)). Este gen ocupaba la fila 4544 en la matriz de expresión con todos los genes. Su nombre es

```
golub.gnames[2000,2]
## [1] "CDC21_HOMOLOG"
```

El identificador de la sonda Affymetrix o AffyID ([??](#)) ocupa la tercera columna y vale, para este gen,

```
golub.gnames[2000,3]
## [1] "X74794_at"
```

Vamos a modificar la matriz `golub` de modo que recoja la información que tenemos de las filas (qué genes son) y de las columnas (qué tipo de leucemia). Para ello podemos aplicar las funciones `rownames` y `colnames` a dichas matrices.

1. Identificamos las filas con el identificador Affy.

```
rownames(golub) = golub.gnames[,3]
```

2. Describimos las columnas con el tipo de leucemia.

```
colnames(golub) = golub.fac
```

Tenemos la información de un modo más compacto.

## 2.5 La función apply

Vamos a explorar los datos `golub` utilizando la función base::`apply`.  
45

¿Qué pretendemos hacer? Vamos a realizar dos dibujos. En abscisas pretendemos dar una medida de localización (media o mediana) de la expresión del gen en el primer tipo de leucemia, ALL. En ordenadas lo mismo pero con el segundo tipo, AML.

Primero hemos de calcular estos valores. La función que calcula la media (muestral) de un vector de números es base::`mean`. Por ejemplo, la media del perfil de expresión del gen en la fila 2000 sería

```
mean(golub[2000,])
## [1] 0.02080211
```

También parece una buena opción cuantificar el valor alrededor de donde se observan las expresiones mediante la mediana. La tenemos con

```
median(golub[2000,])
## [1] 0.147145
```

Vemos que son muy distintos indicando asimetría.<sup>46</sup>

¿Y cómo lo hacemos para todos los genes de una vez? La función base::`apply` lo hace. Primero seleccionamos las partes de la matriz `golub` con las muestras ALL y con las muestras AML.

```
golub.ALL = golub[,golub.fac == "ALL"]
golub.AML = golub[,golub.fac == "AML"]
```

Ahora utilizamos base::`apply`. El primer argumento es la matriz con la que trabajamos, el segundo argumento indica si la función a aplicar es por filas (1) o por columnas (2). El tercer argumento es la función que queremos aplicar. Calculamos la media y la mediana para cada submatriz de `golub`.

<sup>45</sup> Otras muy relacionadas que utilizaremos son base::`lapply` y base::`sapply`.

<sup>46</sup> Cuando para un conjunto de datos no hay coincidencia entre media y mediana lo que sucede es que no hay una disposición simétrica de los datos alrededor de estas medidas de localización.

```
ALL.mean = apply(golub[,golub.fac == "ALL"],1,mean)
AML.mean = apply(golub[,golub.fac == "AML"],1,mean)
ALL.median = apply(golub[,golub.fac == "ALL"],1,median)
AML.median = apply(golub[,golub.fac == "AML"],1,median)
```

Representamos las medias de ALL (abscisas) frente a las medias AML (ordenadas) (figura 2.3(a)).

```
df = data.frame(x0 = ALL.mean, y0 = AML.mean)
png(paste(dirTamiFigures,"RBio72.png",sep=""))
p = ggplot(df,aes(x=x0,y=y0))+geom_point()
p + xlab("Medias ALL") + ylab("Medias AML")
dev.off()
```

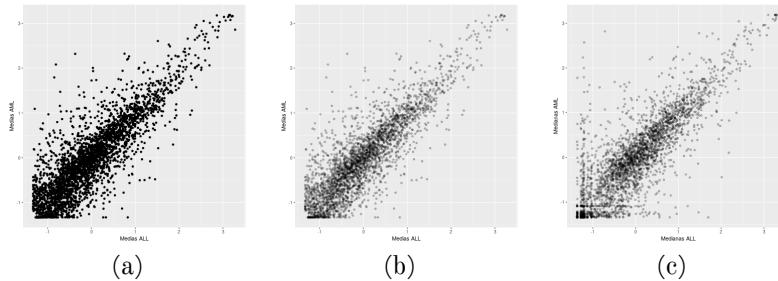


Figura 2.3: Medias en muestras AML frente a medias en muestras ALL para datos multtest::golub. En (a) tenemos los puntos tal cual y en (b) utilizamos un sombreado. (c) Medianas en muestras AML frente a medianas en muestras ALL para datos multtest::golub. sustituimos la media por la mediana.

En la figura 2.3(a) hay muchos puntos representados. Es un scatterplot o diagrama de puntos muy denso. Esto será habitual en lo que sigue pues trabajamos con grandes bancos de datos. Tantos puntos que se van superponiendo nos impide ver en qué zonas hay una mayor densidad de puntos. No es útil un dibujo así. Quizás ayude un sombreado que esté relacionado con la densidad local de puntos que representamos.

<sup>47</sup> Este argumento alpha indicado como una fracción 1/4

en el ejemplo indica que cuando se superpongan 4 puntos entonces se representa el punto completamente negro. Por debajo se utilizan grises.

```
png(paste0(dirTamiFigures,"RBio73.png"))
ggplot(df,aes(x=x0,y=y0))+geom_point(alpha=.25)+xlab("Medias ALL") +
  ylab("Medias AML")
dev.off()
```

Reproducimos el dibujo sustituyendo la media por la mediana (figura 2.3(c)).

```
df = data.frame(x0 = ALL.median, y0 = AML.median)
png(paste(dirTamiFigures,"RBio75.png",sep=""))
p = ggplot(df,aes(x=x0,y=y0))+geom_point(alpha=.25)
p + xlab("Medianas ALL") + ylab("Medianas AML")
dev.off()
```

\* Ej. 4 — Utilizando la matriz de expresión de los datos golub se pide:

- 1.Calcular para cada gen la expresión media sin considerar el tipo de cáncer.
- 2.Calcular para cada gen la desviación estándar.

3. Representar gráficamente un dibujo que, para cada gen, nos muestre en abscisas la expresión media y en ordenadas la desviación estándar.
4. Utilizando la función `grDevices::png` guardar el dibujo anterior en un fichero externo.

\* **Ej. 5** — Con los datos `golub` se pide:

1. Calcular la expresión media para cada una de las condiciones definidas por la variable `golub.cl`.
2. Representar una media frente a la otra.
3. Con la función `abline()` añadir la recta  $y = x$  así como las rectas  $y - x = \delta$  e  $y - x = -\delta$  donde  $\delta$  es un valor que iremos variando.
4. ¿Qué nos indicarían los genes tales que sus puntos asociados están fuera de la región  $\{(x, y) : |y - x| > \delta\}$ .

\* **Ej. 6** — Consideremos los datos `golub`. Se pide:

1. Determinar la desviación estándar de las expresiones de cada gen.
2. Representar un histograma de estas desviaciones estándar.
3. Calcular el percentil de orden 0.9 de las desviaciones calculadas en el punto anterior. Denotemos este valor por  $q_{0.9}$ .
4. Utilizando la función `abline` añadir al dibujo del apartado 2 una línea vertical cuya abscisa coincida con el valor  $q_{0.9}$ .
5. Seleccionar aquellos genes cuya desviación estándar sea mayor que el valor  $q_{0.9}$  del punto anterior.

\* **Ej. 7** — El siguiente código genera una matriz y la guarda en `x`.

```
set.seed(123)
x = matrix(runif(500), nrow=100)
```

Ejecuta en código anterior. Las preguntas que siguen se refieren a la matriz `x`.

1. ¿Cuántas columnas tiene la matriz anterior?
2. Calcula los valores máximos en cada una de las columnas de `x`.
3. Calcula los valores mínimos de las filas de `x`.
4. ¿En qué fila tenemos el mínimo más pequeño de los mínimos por fila calculados en 3?
5. ¿En qué fila tenemos el mínimo más grande de los mínimos por fila calculados en 3?
6. Determina el percentil de orden 0.34 de la primera columna de la matriz `x`.
7. Calcular el percentil de orden 0.23 de las medias por fila de la matriz `x`.
8. ¿Cuál es el mínimo, máximo y valor medio de `x`?
9. Representar un histograma de los datos de la primera columna de `x` utilizando [150, `ggplot2`].
10. Representar un estimador kernel de la densidad de los datos de la primera columna de `x` utilizando [150, `ggplot2`].
11. ¿Cuántos elementos de la matriz `x` son mayores o iguales a 0.12?

- 12.Para cada fila de la matriz `x` calcula cuántos valores son menores o iguales a 0.12.
- 13.De los valores calculados en determina el rango intercuartílico.
- 14.¿Cuántas filas son tales que la primera columna es mayor o igual a la segunda?

**\*\* Ej. 8 —** Construir un `data.frame` utilizando los datos `multtest::golub` que tenga las siguientes características.

- 1.Cada variable (columna) corresponda a una sonda y cada fila corresponda con una muestra distinta.
- 2.Los nombres de las variables han de ser el nombre “`probe1`”, “`probe2`”, etc.
- 3.Añadir una variable (que llamaremos `type` que indique el grupo de leucemia al que pertenece cada muestra. Esta variable ha de ser de tipo `factor` y las etiquetas (o `labels`) han de ser `AML` (leucemia mieloide aguda) y `ALL` (leucemia linfoblástica aguda).

<sup>48</sup> En lo que sigue veremos <sup>48</sup> una clase llamada `ExpressionSet` mucho más adecuada que el `data.frame` para este tipo de datos.

## 2.6 Listas

Las listas son muy importantes en R. En este manual las usaremos, entre otras cosas, cuando trabajamos con grupos de genes. Supongamos que estamos trabajando con un grupo de 10 genes y los representamos con un número. Por tanto nuestro universo de genes es el conjunto  $\{1, \dots, 10\}$ . De estos diez genes suponemos que se ha descrito asociación para los siguientes grupos:

```
geneset1 = c(1,4)
geneset2 = c(2,5,7,8)
geneset3 = c(1,5,6)
```

¿Cómo almacenamos esta información? Lo mejor es una lista. La empezamos creando.

```
(genesets = vector("list",3))

## [[1]]
## NULL
##
## [[2]]
## NULL
##
## [[3]]
## NULL
```

De momento la tenemos vacía. Hacemos que el primer elemento de la lista contenga al primer conjunto.

```
genesets[[1]] = geneset1
```

También para el segundo y tercero.

```
genesets[[2]] = geneset2
genesets[[3]] = geneset3
```

Podemos ver lo que tenemos ahora.

```
genesets

## [[1]]
## [1] 1 4
##
## [[2]]
## [1] 2 5 7 8
##
## [[3]]
## [1] 1 5 6
```

Parece natural darle un nombre a cada elemento. Supongamos que pretendemos llamarlos `set1`, `set2` y `3` respectivamente.

```
names(genesets) = paste0("set", 1:3)
```

Los genes tienen distintas denominaciones como vemos más adelante. De momento, nos conformamos con llamarlos `gene1`, etc. Construimos un vector de nombres.

```
genenames = paste0("gene", 1:10)
```

Convertimos los elementos de la lista utilizando los nombres en lugar de las posiciones. Por ejemplo, para el primero elemento sería

```
genesets[[1]] = genenames[geneset1]
```

Y lo mismo para los otros.

```
genesets[[2]] = genenames[geneset2]
genesets[[3]] = genenames[geneset3]
```

Ya tenemos una lista con los grupos de genes descritos.

## 2.7 Funciones con R

No podemos trabajar con R sin utilizar funciones.<sup>49</sup> La mayor potencia de R es que podemos definir funciones para extenderlo.

<sup>49</sup> Es un lenguaje funcional.

**La estructura básica** de una función en R es

```
nombre.funcion = function(argumentos){
  CODIGO
  resultado
}
```

Supongamos que queremos calcular la media y desviación estándar de un vector de datos. La siguiente función puede valer.

```
MediaDesviacion = function(x){
  resultado = c(mean(x),sd(x))
  resultado
}
```

<sup>50</sup> También podemos usar *return*.

La función devuelve lo que tiene en la última línea.<sup>50</sup> Una vez le declaramos la función podemos usarla.<sup>51</sup> La aplicamos a datos generados con distribución normal.

<sup>51</sup> Copiamos y pegamos en línea de comandos.

```
x = rnorm(23,mean=12,sd=1.3)
MediaDesviacion(x)

## [1] 12.274904 1.278401
```

Queremos calcular la media y desviación estándar pero de los datos que están por encima del percentil de orden p. El valor de p será un *argumento* y queremos fijarlo por defecto en 0.5.

```
MediaDesviacionPercentil = function(x,ordenpercentil = 0.5){
  qp = quantile(x,probs=ordenpercentil)
  x0 = x[x >qp]
  resultado = c(mean(x0),sd(x0))
  resultado
}
```

Y la utilizamos.

```
MediaDesviacionPercentil(x)

## [1] 13.3144744 0.8567743
```

Si queremos cambiar el valor por defecto del orden del percentil hacemos

```
MediaDesviacionPercentil(x,ordenpercentil=.7)

## [1] 13.7880029 0.6425422
```

No hemos comprobado que efectivamente es un vector. Además pretendemos que no nos devuelva nada si no lo que recibe no es un vector. En este caso que nos avise.

```
MediaDesviacionPercentilError = function(x,ordenpercentil = 0.5){
  if(!is.vector(x)) return("No es un vector")
  qp = quantile(x,probs=ordenpercentil)
  x0 = x[x >qp]
  resultado = c(mean(x0),sd(x0))
  resultado
}
```

Vamos a ver qué tal funciona. Definimos una matriz.

```
y = matrix(1:16,ncol=4)
```

Comprobamos que no es un vector.

```
is.vector(y)
## [1] FALSE
```

Y probamos nuestra función.

```
MediaDesviacionPercentilError(y)
## [1] "No es un vector"
```

Funciona y todo. No obstante queda un poco feo que cuando es un vector simplemente nos devuelve otro vector y hemos de saber que es una media lo primero y una desviación lo segundo. Podemos devolver una lista de modo que le demos un nombre a cada cosa.

```
MediaDesviacionPercentilErrorLista = function(x,ordenpercentil = 0.5){
  if(!is.vector(x)) return("No es un vector")
  qp = quantile(x,probs=ordenpercentil)
  x0 = x[x >qp]
  resultado = list(media = mean(x0),de = sd(x0))
  resultado
}
```

Y la probamos.

```
MediaDesviacionPercentilErrorLista(x)
## $media
## [1] 13.31447
##
## $de
## [1] 0.8567743
```

Por ello podemos guardar lo que nos devuelve y luego acceder a cada elemento de la lista.

```
x.des = MediaDesviacionPercentilErrorLista(x)
```

Y tendremos que la media es

```
x.des$media
## [1] 13.31447
```

y la desviación estándar sería

```
x.des$de
## [1] 0.8567743
```

**Ej. 9** — Dado un vector  $x$  programar una función que nos devuelva en una lista la media y la varianza del vector. Los elementos de la lista se han de llamar `media` y `varianza`.

**Ej. 10** — Programa una función que tenga como argumentos un vector  $x$  y un valor numérico  $\alpha$  entre 0 y 1. Eliminar el  $\alpha$  por uno de los más pequeños y la misma cantidad de los más grandes. De lo que queda ha de devolver, en forma de lista, la media y varianza con los nombres `media_ajustada` y `varianza_ajustada`.

## 2.8 Sobre Bioconductor

En este curso tan (o más) importante que el propio R es **Bioconductor**. Básicamente es una colección de paquetes de R para Bioinformática. Se instala con

```
source("http://www.bioconductor.org/biocLite.R")
biocLite()
```

Una vez instalado la parte básica de Bioconductor podemos instalar algún paquete adicional (como *ALL*).

```
source("http://www.bioconductor.org/biocLite.R")
biocLite("ALL")
```

**Sobre la elección de repositorios.** Podemos elegir el repositorio desde el cual instalamos los paquetes de Bioconductor con

```
chooseBioCmirror()
```

También podemos elegir interactivamente el repositorio con

```
setRepositories()
```

\* **Ej. 11** — Instalar los paquetes [72, ggrepify], [48, faraway], [111, multtest] y [34, edgeR].

## 2.9 Paquetes para tami

Los paquetes que utilizamos en este manual , tanto de **CRAN** como de **Bioconductor**, los podemos instalar con el script <http://www.uv.es/ayala/docencia/tami/AllTami.R>. En particular contemplamos varias opciones.

- Si pretendemos una instalación de los paquetes básicos hacemos

```
TIPO = BASICO; source("http://www.uv.es/docencia/tami/AllTami.R")
```

- Una instalación completa <sup>52</sup> se tiene con

```
TIPO = TODO; source("http://www.uv.es/docencia/tami/AllTami.R")
```

<sup>52</sup> Tarda un tiempo y se recomienda una buena conexión a Internet.

## Capítulo 3

# 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 vía microarrays o RNASeq o alguna otra técnica. En otras ocasiones hablaremos de exones o bien, cuando sea pertinente, de isoformas del gen obtenidas con empalmes alternativos.<sup>53</sup> Por tanto es de gran importancia<sup>54</sup> 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, innecesario. ¿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.<sup>55</sup>

<sup>53</sup> Alternative splicing.

<sup>54</sup> Estamos al principio del texto.

<sup>55</sup> El navegador para leer la

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. § 3.2

**OrgDb** Centrados en el organismo.

**TxDb** Paquetes relativos a transcriptomas de un organismo.

**BSgenome** Paquetes con genomas.

<sup>56</sup>

En lo que sigue seguimos el flujo de trabajo [annotation](#) de Bioconductor.

Bases de datos e identificadores de genes.

### 3.1 AnnotationDbi

Las bases de datos de tipo ChipDb, OrgDb y TxDb heredan todos los métodos de la clase `AnnotationDb` que está definida en [108, `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 los 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 [25, `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] "hgu133aACCTNUM"           "hgu133aALIAS2PROBE"
##  [9] "hgu133aCHR"                "hgu133aCHRLLENGTHS"
## [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] "hgu133aUNIGENE"              "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: Human Genome U133 Set
## | MANUFACTURERURL: http://www.affymetrix.com/support/technical/byproduct.affx?product=HGU133A
## | EGSSOURCEDATE: 2015-Sep27
## | EGSSOURCENAME: Entrez Gene
## | EGSSOURCEURL: ftp://ftp.ncbi.nlm.nih.gov/gene/DATA
## | CENTRALID: ENTREZID
## | TAXID: 9606
## | GOSOURCENAME: Gene Ontology
## | GOSOURCEURL: ftp://ftp.geneontology.org/pub/go/godatabase/archive/latest-lite/
## | GOSOURCEDATE: 20150919
## | GOEGSSOURCEDATE: 2015-Sep27
## | GOEGSSOURCENAME: Entrez Gene
## | GOEGSSOURCEURL: 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: ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19
## | GPSOURCEDATE: 2010-Mar22
## | ENSOURCEDATE: 2015-Jul16
## | ENSOURCENAME: Ensembl
## | ENSOURCEURL: ftp://ftp.ensembl.org/pub/current_fasta
## | UPSOURCENAME: Uniprot
## | UPSOURCEURL: http://www.uniprot.org/
## | UPSOURCEDATE: Thu Oct 1 23:31:58 2015

```

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] "ACCCNUM"          "ALIAS"           "ENSEMBL"
## [4] "ENSEMLBLPROT"     "ENSEMLTRANS"      "ENTREZID"
## [7] "ENZYME"            "EVIDENCE"         "EVIDENCEALL"
## [10] "GENENAME"          "GO"               "GOALL"
## [13] "IPI"               "MAP"              "OMIM"
## [16] "ONTOLOGY"          "ONTOLOGYALL"     "PATH"
## [19] "PFAM"              "PMID"             "PROBEID"
## [22] "PROSITE"           "REFSEQ"           "SYMBOL"
## [25] "UCSCKG"            "UNIGENE"          "UNIPROT"

```

Pero: ¿qué información es? Lo obtenemos con<sup>57</sup>

```
help("ENTREZID")
```

<sup>57</sup> 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] "ACCCNUM"      "ALIAS"       "ENSEMBL"
## [4] "ENSEMBLPROT"  "ENSEMBLTRANS" "ENTREZID"
## [7] "ENZYME"        "EVIDENCE"    "EVIDENCEALL"
## [10] "GENENAME"     "GO"          "GOALL"
## [13] "IPI"          "MAP"         "OMIM"
## [16] "ONTOLOGY"     "ONTOLOGYALL" "PATH"
## [19] "PFAM"         "PMID"        "PROBEID"
## [22] "PROSITE"      "REFSEQ"      "SYMBOL"
## [25] "UCSCKG"       "UNIGENE"    "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] "ENSG00000256069" "ENSG00000171428"
## [5] "ENSG00000156006" "ENSG00000196136"
```

Supongamos que tenemos algunos identificadores Affy (AffyID) y pretendemos conocer sus identificadores ENTREZID. Empezamos eligiendo cinco identificadores Affy al azar.<sup>58</sup>

<sup>58</sup> De ahí el uso de la función base::sample.

```
(ids = sample(keys(hgu133a.db,keytype="PROBEID"),5))

## [1] "202288_at"    "219673_at"    "218510_x_at"
## [4] "209088_s_at"  "207778_at"
```

Vamos a obtener sus identificadores Entrez, los de ENSEMBL y su SYMBOL.

```
AnnotationDbi::select(hgu133a.db,keys=ids,
                        columns=c("ENTREZID","ENSEMBL","SYMBOL"),
                        keytype="PROBEID")

##           PROBEID ENTREZID          ENSEMBL SYMBOL
## 1 202288_at      2475 ENSG00000198793   MTOR
## 2 219673_at      254394 ENSG00000111877   MCM9
## 3 218510_x_at     54463 ENSG00000154153 RETREG1
## 4 209088_s_at     29855 ENSG00000118900   UBN1
## 5 207778_at       5969 <NA>            REG1CP
```

## 3.2 ChipDb

Si trabajamos con microarrays ([§ 4](#)) nuestras características serán las sondas.<sup>59</sup> Estas sondas tendrán un identificador que el fabricante del chip le ha asignado. Esto no es informativo para nosotros. Hemos

<sup>59</sup> Asociadas a genes.

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 [25, hgu133a.db]. Lo cargamos.

```
pacman::p_load(hgu133a.db)
```

¿Qué sondas<sup>60</sup> tienen correspondencia en Entrez?

<sup>60</sup> Probes.

```
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**<sup>61</sup> y su identificador Entrez<sup>62</sup>.

<sup>61</sup> AffyID.

<sup>62</sup> ENTREZID.

```
mappedProbesList[1]
```

```
## $`1053_at`  
## [1] "5982"
```

O el correspondiente a la posición 4567.

```
mappedProbesList[4567]
```

```
## $`205255_x_at`  
## [1] "6932"
```

Si hacemos

```
ls("package:hgu133a.db")
```

```
## [1] "hgu133a"           "hgu133a_dbconn"  
## [3] "hgu133a_dbfile"   "hgu133a_dbInfo"  
## [5] "hgu133a_dbschema" "hgu133a.db"  
## [7] "hgu133aACCCNUM"   "hgu133aALIAS2PROBE"  
## [9] "hgu133aCHR"        "hgu133aCHRLLENGTHS"  
## [11] "hgu133aCHRLLOC"   "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] "hgu133aUNIGENE"    "hgu133aUNIPROT"
```

podemos ver todas las correspondencias que nos ofrece el paquete. Consideraremos 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 [27, 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] "ACCCNUM"      "ALIAS"       "ENSEMBL"
## [4] "ENSEMBLPROT"  "ENSEMBLTRANS" "ENTREZID"
## [7] "ENZYME"        "EVIDENCE"    "EVIDENCEALL"
## [10] "GENENAME"      "GO"          "GOALL"
## [13] "IPI"           "MAP"         "OMIM"
## [16] "ONTOLOGY"      "ONTOLOGYALL" "PATH"
## [19] "PFAM"          "PMID"        "PROBEID"
## [22] "PROSITE"       "REFSEQ"     "SYMBOL"
## [25] "UCSCKG"        "UNIGENE"    "UNIPROT"
```

```
keytypes(hgu95av2.db)
```

```
## [1] "ACCCNUM"      "ALIAS"       "ENSEMBL"
## [4] "ENSEMBLPROT"  "ENSEMBLTRANS" "ENTREZID"
## [7] "ENZYME"        "EVIDENCE"    "EVIDENCEALL"
## [10] "GENENAME"      "GO"          "GOALL"
## [13] "IPI"           "MAP"         "OMIM"
## [16] "ONTOLOGY"      "ONTOLOGYALL" "PATH"
## [19] "PFAM"          "PMID"        "PROBEID"
## [22] "PROSITE"       "REFSEQ"     "SYMBOL"
## [25] "UCSCKG"        "UNIGENE"    "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")
```

### 3.3 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] "ACCCNUM"      "ALIAS"       "ENSEMBL"
## [4] "ENSEMBLPROT"  "ENSEMBLTRANS" "ENTREZID"
## [7] "ENZYME"        "EVIDENCE"    "EVIDENCEALL"
## [10] "GENENAME"      "GO"          "GOALL"
## [13] "IPI"           "MAP"         "OMIM"
## [16] "ONTOLOGY"      "ONTOLOGYALL" "PATH"
## [19] "PFAM"          "PMID"        "PROSITE"
## [22] "REFSEQ"        "SYMBOL"     "UCSCKG"
## [25] "UNIGENE"       "UNIPROT"
```

O bien con

```
keytypes(org.Hs.eg.db)
```

```
## [1] "ACCCNUM"      "ALIAS"       "ENSEMBL"
## [4] "ENSEMBLPROT"  "ENSEMBLTRANS" "ENTREZID"
## [7] "ENZYME"        "EVIDENCE"    "EVIDENCEALL"
## [10] "GENENAME"      "GO"          "GOALL"
## [13] "IPI"           "MAP"         "OMIM"
## [16] "ONTOLOGY"      "ONTOLOGYALL" "PATH"
## [19] "PFAM"          "PMID"        "PROSITE"
## [22] "REFSEQ"        "SYMBOL"     "UCSCKG"
## [25] "UNIGENE"       "UNIPROT"
```

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] "ENSG00000256069" "ENSG00000171428"
## [5] "ENSG00000156006" "ENSG00000196136"
```

Y en **Gene Ontology**.

```
head(keys(org.Hs.eg.db, keytype="GO"))

## [1] "GO:0002576" "GO:0003674" "GO:0005576"
## [4] "GO:0005615" "GO:0008150" "GO:0031012"
```

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 identificadores 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 ENSG000000000003.

```
(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 [24, 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")
```

## 3.4 TxDb

Los paquetes TxdB están centrados en el genoma. Vamos a trabajar con la mosca de la drosophila melanogaster.<sup>63</sup> Cargamos la base de datos.

```
pacman::p_load("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.<sup>64</sup>

<sup>63</sup> [https://en.wikipedia.org/wiki/Drosophila\\_melanogaster](https://en.wikipedia.org/wiki/Drosophila_melanogaster).

<sup>64</sup> Todos los paquetes de Bioc conductor 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 [30, Genomic-Features].

```
pacman::p_load(GenomicFeatures)
```

<sup>65</sup> En <http://ucscbrowser.genenetwork.org/cgi-bin/hgGateway?hgSID=732&clade=insect&org=0&db=0> podemos en-

contrar una explicación detallada.

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
## 4 18060346
## 5 12655767
## 6 12653845
## 7 12655300
## 8 12655474
```

Nos devuelve el identificador del gen (**GENEID**), el nombre del transcritito, la hebra o cadena y el punto de inicio y de finalización. Notemos que el primer gen solo tiene un transcriptito mientras que el segundo gen tiene tres transcriptitos.

Los objetos TxDb nos permiten obtener las anotaciones como **GRanges**.<sup>66</sup> ?? Empezamos con los transcriptitos.

```
(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 |
##                 <Rle>      <IRanges>  <Rle> |
## [1]    chr2L      [ 7529,  9484]      + |
## [2]    chr2L      [ 7529,  9484]      + |
## [3]    chr2L      [ 7529,  9484]      + |
## [4]    chr2L      [8011405, 8026898]      + |
##           tx_id      tx_name
##           <integer> <character>
## [1]          1 FBtr0300689
```

```

## [2]      2 FBtr0300690
## [3]      3 FBtr0330654
## [4]    1000 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 |
##                 <Rle>      <IRanges>  <Rle> |
## [1]   chr2L [277930, 278323]      + |
##           exon_id
##           <integer>
## [1]       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

```

<sup>67</sup> En la salida que sigue nos hace una tabla de frecuencias.

Podemos ver la hebra en la que están<sup>67</sup>

```

strand(txdb.cds)

## factor-Rle of length 11003 with 2 runs
##  Lengths: 5489 5514
##  Values : + -
## Levels(3): + - *

```

<sup>68</sup> ??.

A partir de un objeto de clase **TxDb** podemos obtener un **GRangesList**<sup>68</sup> en la cual separamos por alguna característica. Por ejemplo, los objetos **GRanges** para los distintos genes.

```

transcriptsBy(txdb, by="gene")

## GRangesList object of length 2986:
## $FBgn0000018
## GRanges object with 1 range and 2 metadata columns:
##   seqnames      ranges strand |
##   <Rle>      <IRanges> <Rle> |
##   [1] chr2L [10973443, 10975273]     - |
##       tx_id    tx_name
##       <integer> <character>
##   [1] 4279 FBtr0080168
##
## $FBgn0000052
## GRanges object with 3 ranges and 2 metadata columns:
##   seqnames      ranges strand | tx_id
##   [1] chr2L [6041178, 6045593]     - | 3537
##   [2] chr2L [6041178, 6045970]     - | 3538
##   [3] chr2L [6041178, 6045970]     - | 3539
##       tx_name
##   [1] FBtr0079221

```

```

## [2] FBtr0079219
## [3] FBtr0079220
##
## $FBgn0000053
## GRanges object with 2 ranges and 2 metadata columns:
##      seqnames      ranges strand | tx_id
##      [1] chr2L [7014861, 7023940] - | 3704
##      [2] chr2L [7018085, 7023940] - | 3705
##      tx_name
##      [1] FBtr0079431
##      [2] FBtr0100353
##
## ...
## <2983 more elements>
## -----
## seqinfo: 1 sequence from dm3 genome

```

Podemos agrupar los exones por gen.

```

exonsBy(txdb, by="gene")

## GRangesList object of length 2986:
## $FBgn0000018
## GRanges object with 2 ranges and 2 metadata columns:
##      seqnames      ranges strand |
##      <Rle>      <IRanges> <Rle> |
##      [1] chr2L [10973443, 10974210] - |
##      [2] chr2L [10974268, 10975273] - |
##      exon_id exon_name
##      <integer> <character>
##      [1] 11021      <NA>
##      [2] 11022      <NA>
##
## $FBgn0000052
## GRanges object with 6 ranges and 2 metadata columns:
##      seqnames      ranges strand |
##      [1] chr2L [6041178, 6042011] - |
##      [2] chr2L [6042075, 6044351] - |
##      [3] chr2L [6044428, 6045526] - |
##      [4] chr2L [6044428, 6045593] - |
##      [5] chr2L [6045894, 6045970] - |
##      [6] chr2L [6045921, 6045970] - |
##      exon_id exon_name
##      [1] 8949      <NA>
##      [2] 8950      <NA>
##      [3] 8951      <NA>
##      [4] 8952      <NA>
##      [5] 8955      <NA>
##      [6] 8956      <NA>
##
## $FBgn0000053
## GRanges object with 8 ranges and 2 metadata columns:
##      seqnames      ranges strand |
##      [1] chr2L [7014861, 7016374] - |

```

```

## [2] chr2L [7016437, 7017486] - |
## [3] chr2L [7017545, 7017966] - |
## [4] chr2L [7018085, 7018374] - |
## [5] chr2L [7018149, 7018374] - |
## [6] chr2L [7018431, 7019080] - |
## [7] chr2L [7019135, 7019382] - |
## [8] chr2L [7023529, 7023940] - |

## exon_id exon_name
## [1] 9389      <NA>
## [2] 9390      <NA>
## [3] 9391      <NA>
## [4] 9392      <NA>
## [5] 9393      <NA>
## [6] 9394      <NA>
## [7] 9395      <NA>
## [8] 9396      <NA>

##
## ...
## <2983 more elements>
## -----
## seqinfo: 1 sequence from dm3 genome

```

O bien podemos tener los **CDS** agrupados por transcripto.

```

cdsBy(txdb, by="tx")

## GRangesList object of length 4951:
## $1
## GRanges object with 2 ranges and 3 metadata columns:
##   seqnames      ranges strand |  cds_id
##   <Rle>        <IRanges> <Rle> | <integer>
##   [1] chr2L [7680, 8116]    + |      1
##   [2] chr2L [8193, 8610]    + |      3
##   cds_name exon_rank
##   <character> <integer>
##   [1]      <NA>       1
##   [2]      <NA>       2
##
## $2
## GRanges object with 3 ranges and 3 metadata columns:
##   seqnames      ranges strand |  cds_id
##   [1] chr2L [7680, 8116]    + |      1
##   [2] chr2L [8193, 8589]    + |      2
##   [3] chr2L [8668, 9276]    + |      5
##   cds_name exon_rank
##   [1]      <NA>       1
##   [2]      <NA>       2
##   [3]      <NA>       3
##
## $3
## GRanges object with 2 ranges and 3 metadata columns:
##   seqnames      ranges strand |  cds_id
##   [1] chr2L [7680, 8116]    + |      1
##   [2] chr2L [8229, 8610]    + |      4

```

```

##      cds_name exon_rank
## [1]     <NA>        1
## [2]     <NA>        2
##
## ...
## <4948 more elements>
## -----
## seqinfo: 1 sequence from dm3 genome

```

O los intrones agrupados por transcríto.

```

intronsByTranscript(txdb)

## GRangesList object of length 5384:
## $1
## GRanges object with 1 range and 0 metadata columns:
##   seqnames      ranges strand
##   <Rle>    <IRanges>  <Rle>
## [1] chr2L [8117, 8192]      +
##
## $2
## GRanges object with 2 ranges and 0 metadata columns:
##   seqnames      ranges strand
##   [1] chr2L [8117, 8192]      +
##   [2] chr2L [8590, 8667]      +
##
## $3
## GRanges object with 1 range and 0 metadata columns:
##   seqnames      ranges strand
##   [1] chr2L [8117, 8228]      +
##
## ...
## <5381 more elements>
## -----
## seqinfo: 1 sequence from dm3 genome

```

También podemos tener las regiones UTR 5' y 3' agrupadas por transcríto.

```

fiveUTRsByTranscript(txdb)

## GRangesList object of length 4733:
## $1
## GRanges object with 1 range and 3 metadata columns:
##   seqnames      ranges strand | exon_id
##   <Rle>    <IRanges>  <Rle> | <integer>
## [1] chr2L [7529, 7679]      + |       1
##   exon_name exon_rank
##   <character> <integer>
## [1]     <NA>        1
##
## $2
## GRanges object with 1 range and 3 metadata columns:
##   seqnames      ranges strand | exon_id

```

```

## [1] chr2L [7529, 7679] + | 1
## exon_name exon_rank
## [1] <NA> 1
##
## $3
## GRanges object with 1 range and 3 metadata columns:
##   seqnames      ranges strand | exon_id
##   [1] chr2L [7529, 7679] + | 1
##   exon_name exon_rank
##   [1] <NA> 1
##
## ...
## <4730 more elements>
## -----
## seqinfo: 1 sequence from dm3 genome

```

```

threeUTRsByTranscript(txdb)

## GRangesList object of length 4721:
## $1
## GRanges object with 1 range and 3 metadata columns:
##   seqnames      ranges strand | exon_id
##   <Rle>    <IRanges> <Rle> | <integer>
##   [1] chr2L [8611, 9484] + | 3
##   exon_name exon_rank
##   <character> <integer>
##   [1] <NA> 2
##
## $2
## GRanges object with 1 range and 3 metadata columns:
##   seqnames      ranges strand | exon_id
##   [1] chr2L [9277, 9484] + | 5
##   exon_name exon_rank
##   [1] <NA> 3
##
## $3
## GRanges object with 1 range and 3 metadata columns:
##   seqnames      ranges strand | exon_id
##   [1] chr2L [8611, 9484] + | 4
##   exon_name exon_rank
##   [1] <NA> 2
##
## ...
## <4718 more elements>
## -----
## seqinfo: 1 sequence from dm3 genome

```

## 3.5 BSgenome

Estos paquetes contienen datos de secuencias para organismos secuenciados.

### B<sub>S</sub>genome.Dmelanogaster.UCSC.dm3

```
pacman::p_load(BSgenome.Dmelanogaster.UCSC.dm3)
tx2seqs = extractTranscriptSeqs(BSgenome.Dmelanogaster.UCSC.dm3, TxDb.Dmelanogaster.UCSC
```

La secuencia correspondiente al primer gen sería

```
tx2seqs[[1]]
##   1880-letter "DNAString" instance
## seq: CTACTCGCATGTAGAGATTTC...TTCAATAAAATTCCCCAAGT
```

Si queremos conocer la secuencia entre las posiciones 1000 y 1020 entonces

```
tx2seqs[[1]][1000:1020]
##   21-letter "DNAString" instance
## seq: ATATTGATGTCTTCGTACCC
```

También podemos trasladar 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 traslada. Para obtener obtener las que realmente se trasladan se puede hacer

```
cds2seqs = extractTranscriptSeqs(BSgenome.Dmelanogaster.UCSC.dm3,
                                 cdsBy(txdb, by="tx"))
translate(cds2seqs)
```

### B<sub>S</sub>genome.Hsapiens.UCSC.hg19

Estos paquetes contienen datos de secuencias para organismos secuenciados.

```
pacman::p_load(BSgenome.Hsapiens.UCSC.hg19)
```

Hsapiens

```
## Human genome:
## # organism: Homo sapiens (Human)
## # provider: UCSC
## # provider version: hg19
## # release date: Feb. 2009
## # release name: Genome Reference Consortium GRCh37
## # 93 sequences:
## #     chr1                  chr2
```

```

## #   chr3           chr4
## #   chr5           chr6
## #   chr7           chr8
## #   chr9           chr10
## # ...
## #   chrUn_g1000241  chrUn_g1000242
## #   chrUn_g1000243  chrUn_g1000244
## #   chrUn_g1000245  chrUn_g1000246
## #   chrUn_g1000247  chrUn_g1000248
## #   chrUn_g1000249
## # (use 'seqnames()' to see all the sequence names,
## # use the '$' or '[' operator to access a given
## # sequence)

```

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])

## A DNAStringSet instance of length 2
##      width seq      names
## [1] 249250621 NNNNNNN...NNNNNNN chr1
## [2] 243199373 NNNNNNN...NNNNNNN chr2

```

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

## GRanges object with 2 ranges and 0 metadata columns:
##      seqnames      ranges strand
##      <Rle>      <IRanges>  <Rle>
## [1]   chr1 [100000, 100023]    +
## [2]   chr4 [300000, 300037]    -
## -----
##      seqinfo: 2 sequences from an unspecified genome; no seqlengths

res <- getSeq(Hsapiens, rngs)
res

## A DNAStringSet instance of length 2
##      width seq
## [1]    24 CACTAAGCACACAGAGAATAATGT
## [2]    38 GCTGGTCCCTTACTTCCAGTAGAAAAGACGTGTTCAAGG

```

## 3.6 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.<sup>69</sup>

<sup>69</sup> 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_g1000237 [ 1, 2686]   -
## [82957] chrUn_g1000241 [20433, 36875] -
## [82958] chrUn_g1000243 [11501, 11530] +
## [82959] chrUn_g1000243 [13608, 13637] +
## [82960] chrUn_g1000247 [ 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
```

### 3.7 biomaRt

El paquete [42, biomaRt] es un interfaz para poder acceder a una serie de bases de datos que implementan BioMart.<sup>70</sup>

<sup>70</sup> <http://www.biomart.org>.

```
pacman::p_load(biomaRt)
```

Podemos ver la lista de `mart`'s que tenemos (mostramos los primeros).

```
listMarts(host="www.ensembl.org")
```

```
##                               biomart      version
## 1 ENSEMBL_MART_ENSEMBL    Ensembl Genes 92
## 2   ENSEMBL_MART_MOUSE     Mouse strains 92
## 3     ENSEMBL_MART_SNP    Ensembl Variation 92
## 4 ENSEMBL_MART_FUNCGEN Ensembl Regulation 92
```

Elegimos utilizar `ensembl`.

```
(ensembl = useMart("ENSEMBL_MART_ENSEMBL",host="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           mfuro_gene_ensembl
## 2           jjaculus_gene_ensembl
## 3 mfascicularis_gene_ensembl
## 4     psinensis_gene_ensembl
## 5     xmaculatus_gene_ensembl
## 6       pmarinus_gene_ensembl
##                                         description
## 1                               Ferret genes (MusPutFur1.0)
## 2             Lesser Egyptian jerboa genes (JacJac1.0)
## 3 Crab-eating macaque genes (Macaca_fascicularis_5.0)
## 4           Chinese softshell turtle genes (PelSin_1.0)
## 5                 Platyfish genes (Xipmac4.4.2)
## 6                   Lamprey genes (Pmarinus_7.0)
##                               version
## 1           MusPutFur1.0
## 2             JacJac1.0
## 3 Macaca_fascicularis_5.0
## 4           PelSin_1.0
## 5             Xipmac4.4.2
## 6           Pmarinus_7.0
```

Elegimos la correspondiente a ser humano.

```
(ensembl = useMart("ENSEMBL_MART_ENSEMBL", dataset="hsapiens_gene_ensembl",
host="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))
```

## 3.8 KEGGREST

Con [134, KEGGREST] podemos acceder a la base de datos KEGG. En concreto permite el acceso a [KEGG REST API](#).

## 3.9 Tareas habituales con anotaciones

¿Cuáles son las tareas que tenemos que realizar habitualmente? En esta sección vemos posibles soluciones que las resuelvan.

### 3.9.1 Cambiar identificadores de un ExpressionSet

Consideremos el ExpressionSet tamidata::gse1397.<sup>71</sup> 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.

<sup>71</sup> § 5.10.

```
data(gse1397, package="tamidata")
```

¿Qué paquete de anotación necesitamos?

```
Biobase::annotation(gse1397)
## [1] "hgu133a"
```

72 En <https://www.bioconductor.org/packages/3.3/data/annotation/> Lo cargamos.<sup>72</sup>

tenemos el listado de los paquetes de anotación de los que dispone Bioconductor.

Como hemos visto en § 3.1 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] "ACCCNUM"      "ALIAS"        "ENSEMBL"
## [4] "ENSEMLBLPROT" "ENSEMLTRANS"   "ENTREZID"
## [7] "ENZYME"       "EVIDENCE"     "EVIDENCEALL"
## [10] "GENENAME"     "GO"          "GOALL"
## [13] "IPI"          "MAP"          "OMIM"
## [16] "ONTOLOGY"     "ONTOLOGYALL" "PATH"
## [19] "PFAM"         "PMID"         "PROBEID"
## [22] "PROSITE"      "REFSEQ"       "SYMBOL"
## [25] "UCSCKG"       "UNIGENE"     "UNIPROT"
```

```
keytypes(hgu133a.db)
```

```
## [1] "ACCCNUM"      "ALIAS"        "ENSEMBL"
## [4] "ENSEMLBLPROT" "ENSEMLTRANS"   "ENTREZID"
## [7] "ENZYME"       "EVIDENCE"     "EVIDENCEALL"
## [10] "GENENAME"     "GO"          "GOALL"
## [13] "IPI"          "MAP"          "OMIM"
## [16] "ONTOLOGY"     "ONTOLOGYALL" "PATH"
## [19] "PFAM"         "PMID"         "PROBEID"
## [22] "PROSITE"      "REFSEQ"       "SYMBOL"
## [25] "UCSCKG"       "UNIGENE"     "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. ¿Quiénes 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
```

¿Qué 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 devuelto 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 1007_s_at 100616237
```

```
## 3    1053_at      5982
## 4    117_at       3310
## 5    121_at       7849
## 6  1255_g_at     2978
```

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" "1007_s_at" "1053_at"
## [4] "117_at"    "121_at"   "1255_g_at"
## [7] "1294_at"

head(indices)

## [1] 1 3 4 5 6 7
```

Elige la primera correspondencia e ignora las demás 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
```

## 3.10 Ejercicios

\* **Ej. 12** — Determinar los códigos **Entrez**, **Gene Ontology** y **Ensembl** para los genes BRCA1 y BRCA2 implicados en el cáncer de mama.

\* **Ej. 13** — Se pide encontrar el gen BRCA1 en el ratón (*Mus musculus*). Utilizad el paquete de anotación [29, org.Mm.eg.db].

## **Parte II**

### **Datos**



# Capítulo 4

## Microarrays

### 4.1 Introducción

En este tema tratamos sobre datos de expresión obtenidos utilizando microarrays. 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.

Los datos multtest::golub que hemos usado ya han sido preprocesados. 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 la referencia de mayor interés es [60]. En particular tienen un interés especial los tres primeros capítulos.

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.

### 4.2 Affymetrix GeneChip

Es conveniente consultar [152, páginas 93 y siguientes]. Vamos a leer datos obtenidos mediante un escáner Affymetrix GeneChip. Estos arrays de oligonucleótidos tienen un pequeño número de sondas para un mismo gen. Hablaremos de conjunto de sondas refiriéndonos a todas las sondas que en un mismo chip están asociadas con un mismo gen. Cada sonda son pequeñas celdas con oligonucleótidos específicos de un gen. Estos oligonucleótidos tienen 25 pares de bases. Cada oligonucleótido es una parte (aproximadamente) específica de un gen. En principio solo el mRNA de este gen lo reconocerá como propio. Sin embargo, también puede ocurrir que el RNA de otros genes encuentre partes comunes con la sonda y se adjunte. Tendríamos una hibridación no específica. Con objeto de considerar el problema Affymetrix modifica la secuencia original. Concretamente el par de bases en la posición 13. De este modo una hibridación con esta sonda se considera un acoplamiento erróneo.

Las imágenes digitales con la información original son los ficheros DAT. Una vez procesados por el software que proporciona el fabricante obtenemos un resumen por celda que los guarda en los ficheros CEL. Este fichero es el resultado del escaneo y la segmentación de la imagen obtenida. Todavía no hemos resumido estas intensidades de los píxeles en un valor por gen. En estos ficheros tenemos la media y desviación estándar de los niveles de gris así como la localización de la sonda dentro del array. Se necesita también conocer la correspondencia entre sondas y nombres de los genes. Esta información aparece en el fichero CDF.<sup>1</sup>

La información que se maneja se puede considerar a tres niveles:

**Imagen** El primer nivel es de la imagen digital (ficheros DAT en Affymetrix GeneChip). Hay que utilizar técnicas de procesado de imagen digital para obtener a partir de esta imagen otra imagen donde cada pixel tiene como nivel de gris la expresión de la sonda en esa celda.

**Datos a nivel de sonda** Tenemos un imagen digital donde un pixel contiene una cuantificación de la cantidad de hibridación.

**Datos de expresión** Ya tenemos corregido el fondo y normalizadas las muestras. Tenemos una matriz de expresión de modo que en la fila tenemos el gen y en la columna la muestra. A partir de aquí se trabaja para analizar la posible actividad diferenciada de un gen o un conjunto de genes.

### 4.3 Obteniendo los datos

En esta sección vamos a utilizar los datos con número de acceso [GEO GSE21779](#).<sup>73</sup>

### 4.4 Sonda y conjunto de sondas

En estos microarrays tenemos distintas sondas correspondientes al mismo gen. ¿Qué chip o plataforma se ha utilizado se ha utilizado?

Podemos verlo con Biobase::annotation.

```
pacman::p_load(Bioconductor, affy)
annotation(gse21779raw)

## [1] "hg13plus2"
```

El número de sondas y de muestras lo podemos conocer con

```
dim(exprs(gse21779raw))
```

```
## [1] 1354896      18
```

Nos fijamos en el primer array cuyas expresiones aparecen en la primera columna. Podemos ver las intensidades del primer microarray en forma de una imagen a niveles de gris. Cuando más blanco es el punto mayor es la intensidad observada en la sonda y por lo tanto mayor la

---

<sup>1</sup> Cuando leamos datos veremos que si no tenemos cargado el fichero CDF necesario el programa los baja sin que se lo pidamos.

expresión del gen al que está asociada. Cada pixel corresponde con una sonda distinta. La imagen la tenemos en la figura 4.1.

```
png(paste0(dirTamiFigures, "microarray4"))
image(gse21779raw[,1])
dev.off()

## pdf
## 2
```

Podemos conocer las identificadores que Affymetrix le ha asignado a estas sondas con

```
probeNames(gse21779raw)
```

Nos fijamos en la sonda que aparece en la posición 400.

```
ID = probeNames(gse21779raw)[400]
```

El número de sondas que componen el conjunto asociado a este identificador es

```
sum(probeNames(gse21779raw) == ID)

## [1] 11
```

Podemos ver los cardinales de todos los conjuntos de sondas que lleva el chip.

```
counts = table(probeNames(gse21779raw))
table(counts)

## counts
##     8      9     10     11     13     14     15     16 
##     5      1     6 54130      4      4      2   482 
##    20     69 
##    40      1
```

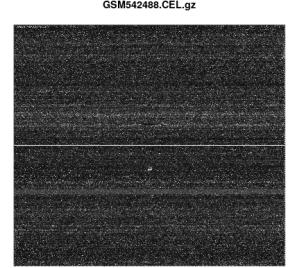


Figura 4.1: Imagen a niveles de gris correspondiente al primer array de los datos gse21779.

## 4.5 Variabilidad intra y entre arrays

Dentro de todo el vector con los nombres de las sondas buscamos las que ocupa la primera tanto para PM como para MM (de ahí el argumento `both`).

```
indexProbes(gse21779raw, "both", ID) [[1]]

## [1] 1088751 883561 1054203 366753 178855
## [6] 68793 490689 62456 1123802 939573
## [11] 1349651 1089915 884725 1055367 367917
## [16] 180019 69957 491853 63620 1124966
## [21] 940737 1350815
```

¿Cuáles corresponden a PM?

```
(posiciones = indexProbes(gse21779raw, "pm", ID) [[1]])

## [1] 1088751 883561 1054203 366753 178855
## [6] 68793 490689 62456 1123802 939573
## [11] 1349651
```

Tenemos 11 correspondientes a PM. Y las demás corresponden a MM:

```
indexProbes(gse21779raw, "mm", ID) [[1]]

## [1] 1089915 884725 1055367 367917 180019
## [6] 69957 491853 63620 1124966 940737
## [11] 1350815
```

<sup>74</sup> Probe set.

En figura 4.2(a) vemos dónde está localizado el conjunto de sondas<sup>74</sup> dentro de la imagen. En cada localización tenemos un valor de PM y otro valor MM.

Vamos a representar los niveles de expresión PM correspondientes a un mismo conjunto de sondas de un array dado (figura 4.2(b)).

```
library(ggplot2)
pmID = probes(gse21779raw[,1], "pm", ID)
df1 = data.frame(sondas = 1:11, intensidad = pmID[, "GSM542488.CEL.gz"])
ggplot(df1, aes(x=sondas, y=intensidad)) + geom_line()
```

En abscisas tenemos el número de la sonda (de un mismo conjunto) y en ordenadas su expresión. Si no hubiera ningún tipo de ruido lo que se debe observar sería una línea horizontal, unos mismos niveles de expresión. Pero no es así.

El dibujo 4.2(b) se refiere a la variabilidad **intra array** (utilizando las sondas de un mismo conjunto) para un mismo gen. Vamos superponer (una línea por muestra) las expresiones del mismo gen pero para todas las muestras. De este modo ilustramos la variabilidad entre muestras. En la figura 4.2(c) mostramos este dibujo.<sup>75</sup>

<sup>75</sup> Observad el uso de la función reshape::melt para obtener el data.frame.

```
pm0 = probes(gse21779raw, "pm", ID)
pm0 = data.frame(pm0, sondas=1:11)
pm1 = reshape::melt(pm0, id="sondas")
ggplot2::ggplot(data=pm1, aes(x=sondas, y=value, colour=variable)) +
  geom_line() + ylab("Intensidad")
```

```
png(paste(dirTamiFigures, "gse217791arrays.png", sep=""))
pm0 = probes(gse21779raw, "pm", ID)
pm0 = data.frame(pm0, sondas=1:11)
pm1 = reshape::melt(pm0, id="sondas")
ggplot2::ggplot(data=pm1, aes(x=sondas, y=value, colour=variable)) +
  geom_line() + ylab("Intensidad")
dev.off()
```

En la figura 4.2(c) cada línea corresponde a un array y los puntos que representamos en cada línea es un conjunto de sondas. Este conjunto de sondas es el mismo que elegimos en todos los arrays.

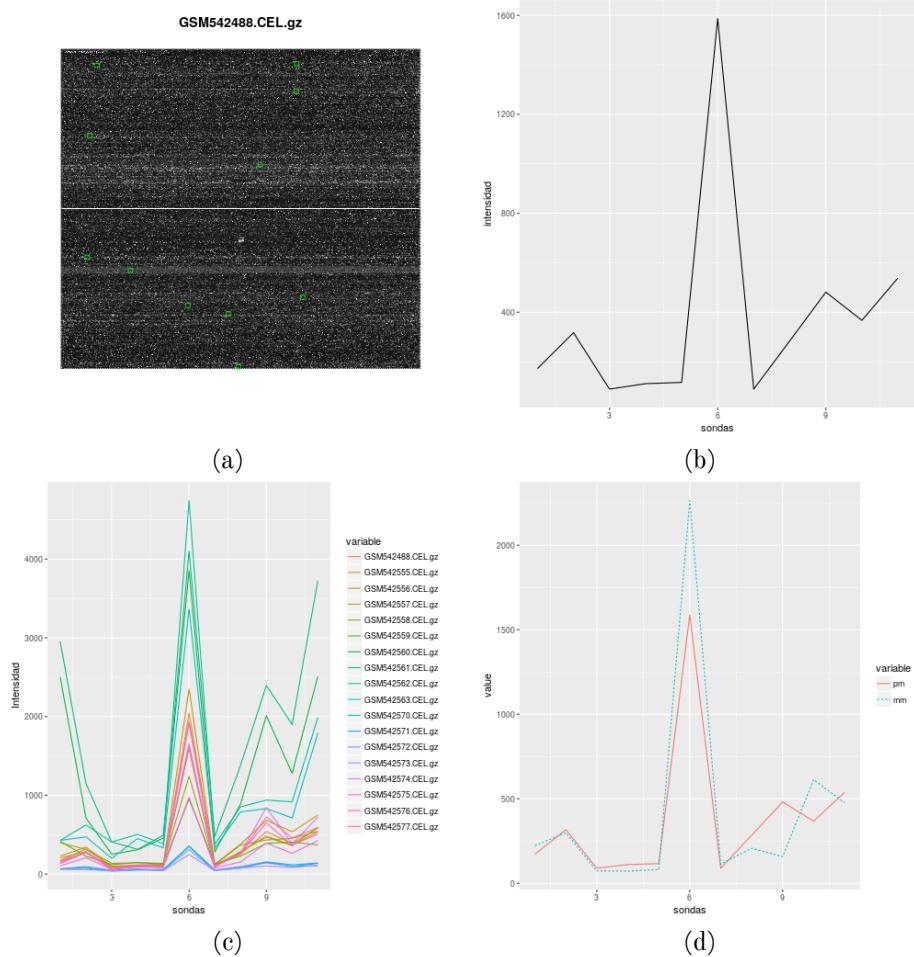


Figura 4.2: (a) Image CEL correspondiente al primer array o chip. Cada pixel corresponde con una sonda distinta. Los cuadrados verdes nos localizan un conjunto de sondas asociadas al mismo gen. (b) Los valores (o intensidades) PM para un grupo de sondas en el primer array de nuestras muestras. (c) Los valores (o intensidades) PM para un mismo grupo de sondas en los distintos arrays. Cada línea corresponde con un array distinto. En abscisas tenemos la misma sonda en cada uno de los arrays. (d) En la primera muestra de gse21779 mostramos los valores PM y, en las mismas localizaciones, los valores MM. No siempre PM es mayor que MM como sería de esperar.

¿Qué significa este dibujo? ¿Cómo lo podemos interpretar? Si las distintas sondas asociadas al mismo gen nos dieran una misma expresión cada una de las líneas debía ser horizontal. No es así como hemos visto para la primera y para todas las demás. Tenemos ahora una segunda fuente de variación. Estas muestras se han tomado bajo tres condiciones distintas. Si no hubiera diferencia entre condiciones las líneas debieran de estar muy superpuestas.

De acuerdo al diseño de este chip cabe esperar que en una misma localización el valor PM sea mayor que el valor de MM. Esto es lo esperable ya que PM es señal, estamos observando la hibridación específica (aquella para la que la sonda ha sido diseñada). El valor de MM lo que mide es ruido óptico e hibridación no específica, en resumen, ruido en un sentido amplio. La señal debe dominar al ruido, el valor de PM debiera de ser mayor que el valor de MM en cada celda. Esto sobre el papel. Pero no en la realidad. Veamos (figura 4.2(d)) para un mismo array y para un conjunto de genes dados los valores de PM y los valores MM.

```
library(reshape)
pmm = data.frame(sondas=1:11,pm = probes(gse21779raw[,1],"pm",ID),
                 mm = probes(gse21779raw[,1],"mm",ID))
df2 = reshape::melt(pmm,id="sondas")
levels(df2[,"variable"]) = c("pm","mm")
ggplot(df2,aes(x=sondas,y=value,colour=variable,linetype=variable))
+geom_line()
```

Podemos hacer una valoración global para todos los arrays que tenemos. ¿Qué proporción de sondas son tales que el valor PM es menor que el correspondiente valor MM?

```
table(probes(gse21779raw,"pm") >= probes(gse21779raw,"mm"))

##      FALSE      TRUE
## 4460757 6415887
```

Vemos que tenemos un 41 % en este caso particular. No es poco.

## 4.6 Control de calidad

Vamos a estudiar distintos procedimientos que intentan evaluar los distintos microarrays que componen nuestro experimento. El objetivo es decidir si es necesario descartar total o parcialmente alguna de las muestras.

### 4.6.1 Dibujo de la media-diferencia de Tukey o dibujo de Bland-Altman

El dibujo media-diferencia de Tukey es más habitualmente conocido como el dibujo Bland-Altman estos autores lo popularizaron utilizándolo en una revista médica.<sup>76</sup>

<sup>76</sup> El trabajo completo (con correcciones respecto de la publicación original) es de Bland-Altman y corresponde con la referencia [16].

Supongamos que tenemos pares de medidas  $(x_i, y_i)$  con  $i = 1, \dots, n$  y pretendemos ver el grado de coincidencia de las mismas. O, dicho de otro modo, si valores mayores corresponden con diferencias mayores entre las cantidades observadas. Es de suponer que dos procedimientos

que coincidan tendrán un coeficiente de correlación alto. Sin embargo, este valor viene condicionado por la variabilidad de las medidas. El dibujo media-diferencia de Tukey o dibujo de Bland-Altman es una representación gráfica que sirve para valorar el grado de coincidencia de los pares de medidas. Supongamos que tenemos los pares  $(x_i, y_i)$  con  $i = 1, \dots, n$  entonces consideramos los puntos que tienen coordenadas

$$\left( \frac{x_i + y_i}{2}, (x_i - y_i) \right).$$

Es decir, en abscisas tenemos la media de las dos medidas  $\frac{x_i + y_i}{2}$  y en ordenadas tenemos su diferencia,  $x_i - y_i$ .

En ocasiones, este dibujo se completa con dos líneas horizontales. Si  $d_i = x_i - y_i$  entonces podemos completar la representación gráfica añadiendo dos líneas horizontales correspondientes con los límites del intervalo de confianza para la diferencia media entre los dos procedimientos esto es las líneas tendrían como ordenadas  $\bar{d} \pm t_{n-1, 1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}}$  donde  $S$  es la desviación estándar de las diferencias  $d_i$  y  $t_{n-1, 1-\frac{\alpha}{2}}$  es el percentil  $1 - \frac{\alpha}{2}$  de una t de Student con  $n - 1$  grados de libertad.

#### 4.6.2 MA plots

Estos dibujos sirven para determinar si alguno de los microarrays debe de ser descartado del estudio y si necesitamos homogeneizar los valores para las distintas muestras. Son una aplicación del dibujo media-diferencia propuesto por Tukey.<sup>77</sup>

Trabajamos, en lugar de con las intensidades originales observadas, con su logaritmo en base 2. Supongamos que  $u_i$  y  $v_i$  denotan las intensidades originales en la sonda  $i$ -ésima. Consideramos  $x_i = \log_2 u_i$  e  $y_i = \log_2 v_i$  y estos son los valores a comparar. En este contexto se habla de dibujos MA ya que

$$m_i = x_i - y_i = \log_2 \frac{u_i}{v_i},$$

y

$$a_i = \frac{1}{2}(x_i + y_i) = \frac{1}{2} \log_2 u_i v_i = \log_2 \sqrt{u_i v_i}.$$

Una primera opción es comparar todos los pares de microarrays utilizando el dibujo media-diferencia de Tukey.

Una segunda opción consiste en considerar una especie de microarray típico y comparar los demás con este. Por ejemplo, una opción que implementa el paquete [75, affy] en la función affy::MAplot consiste en calcular para cada sonda la mediana a lo largo de todos los microarrays que componen nuestro experimento. Por tanto, la mediana de las intensidades será  $x_i$  y el valor  $y_i$  corresponde a cada uno de los microarrays de nuestro experimento.

**Ejemplo 4.1 (MA plot)** Vamos a ilustrar el uso de los dibujos MA con los datos **gse21779**. En la figura 4.3 podemos ver una comparación entre los dos primeros microarrays del experimento.

```
MAplot(gse21779raw[,1:2], pairs=TRUE, plot.method="smoothScatter")
```

<sup>77</sup> Es conveniente consultar [http://en.wikipedia.org/wiki/MA\\_plot](http://en.wikipedia.org/wiki/MA_plot).

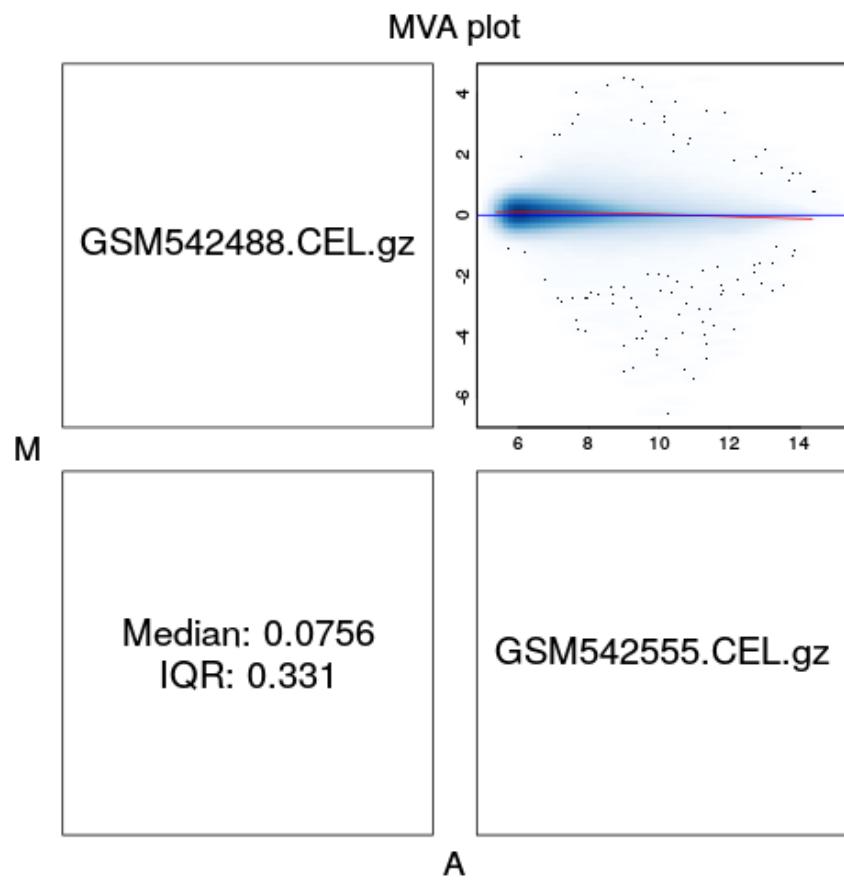


Figura 4.3: MAplot comparando los dos primeros microarrays.

### 4.6.3 Densidades y diagramas de cajas

En los dos dibujos anteriores hemos ilustrado la variabilidad intra y entre arrays para un mismo gen. Tomemos un punto de vista más global. En particular, vamos a mostrar el comportamiento de los niveles de expresión para todas las sondas en un array, por ejemplo, el primero de ellos. Estamos evaluando la posibilidad de que alguno de ellos tenga un nivel de ruido inaceptable y debiera de ser eliminado del estudio.

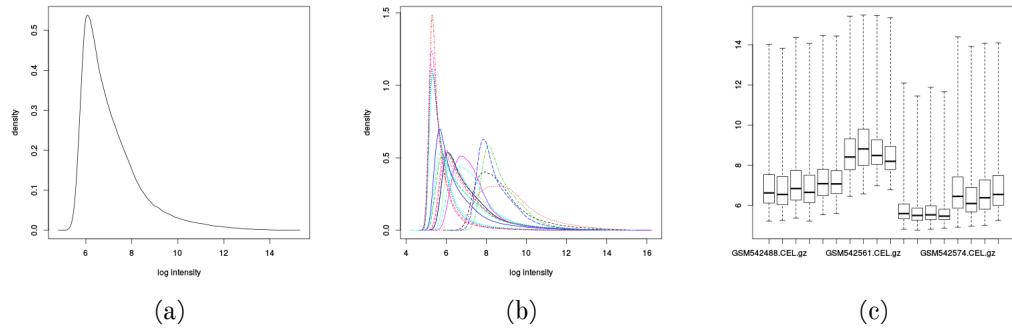


Figura 4.4: a) Estimador kernel de la densidad de las expresiones a nivel de sonda en el primer microarray. b) Estimadores kernel de la densidad de las intensidades para los distintos microarrays. Vemos que hay diferencias claras. c) Diagramas de cajas de las expresiones a nivel de sonda para los datos gse21779.

Empezamos mostrando en la figura 4.4(a) la densidad de los niveles de expresión en el primer array. Lo hacemos utilizando un estimador kernel de la densidad.

```
affy:::hist(gse21779raw[,1])
```

Aunque la función que utilizamos es `hist` (de histograma), no es histograma.<sup>2</sup> Es un estimador kernel de la densidad. Con objeto de ver (y comparar) la variabilidad en los distintos arrays podemos estimar los estimadores kernel de las densidades de las expresiones en cada uno de los arrays que componen nuestra muestra. Lo tenemos en la figura 4.4(b).

```
affy:::hist(gse21779raw)
```

Llama la atención la gran diferencia entre estos estimadores de la densidad para distintas muestras. En cualquier caso en todos ellos vemos una clara asimetría a la derecha. Obviamente corresponde a los genes con un alto nivel de expresión.

De un modo análogo podemos para evaluar la variabilidad las expresiones en cada arrays y entre arrays podemos utilizar diagramas de cajas. Se muestra en la figura 4.4(c).

```
affy:::boxplot(gse21779raw)
```

En este caso los genes con una alta (anormalmente grande) expresión corresponden con el bigote superior en los diagramas de caja que acabamos de representar.

<sup>2</sup>Una mala elección de nombre.

#### 4.6.4 Utilizando arrayQualityMetrics

<sup>78</sup> El informe correspondiente lo tenemos en [http://www.uv.es/ayala/docencia/tami/reports/Report\\_for\\_gse21779/index.html](http://www.uv.es/ayala/docencia/tami/reports/Report_for_gse21779/index.html).

```
library(arrayQualityMetrics)
arrayQualityMetrics(expressionset = gse21779raw,
outdir = "Report_for_gse21779raw", ## Fija directorio de salida
force = TRUE, ## Sobreescribe
do.logtransform = TRUE) ## Transforma a log2
```

#### 4.6.5 Corrigiendo errores técnicos

Si algo queda claro en esta sección es la necesidad de tratar los datos antes de poder compararlos y obtener alguna conclusión biológica. Es decir, hemos de trabajar los valores observados a nivel de sonda antes de obtener un resumen que es la expresión del gen.

De hecho podemos ver los mínimos de PM en las distintas muestras.

```
apply(probes(gse21779raw, "pm"), 2, min)

## GSM542488.CEL.gz GSM542555.CEL.gz
##            32            31
## GSM542556.CEL.gz GSM542557.CEL.gz
##            34            33
## GSM542558.CEL.gz GSM542559.CEL.gz
##            21            39
## GSM542560.CEL.gz GSM542561.CEL.gz
##            67            60
## GSM542562.CEL.gz GSM542563.CEL.gz
##            74            82
## GSM542570.CEL.gz GSM542571.CEL.gz
##            26            27
## GSM542572.CEL.gz GSM542573.CEL.gz
##            26            27
## GSM542574.CEL.gz GSM542575.CEL.gz
##            28            29
## GSM542576.CEL.gz GSM542577.CEL.gz
##            28            32
```

Y vemos como efectivamente no son nulos. La *corrección de fondo* pretende que aproximadamente sean nulos los valores más pequeños. Esto correspondería con aquellos genes que no tienen actividad.

También hemos visto que la distribución de las intensidades es claramente muy diferente entre arrays. Lo mostraban los estimadores kernel y los diagramas de cajas. El proceso de hacer comparables los datos entre arrays es lo que se conoce como *normalización*.

Una vez hemos corregido fondo y normalizadas las muestras, seguimos teniendo para cada gen distintas sondas en cada array. Toca resumir estos valores en un solo valor que indique la expresión del gen asociado al conjunto de sondas.

Estas tres operaciones pueden realizarse de un modo secuencial o bien hay métodos que las realizan de un modo combinado. Hay publica-

dos muchos métodos. Lo que vamos a utilizar en este capítulo está en el paquete [75, *affy*].<sup>79</sup>

En estas notas vamos a estudiar con detalle dos métodos. El primero es un método que corrige cada chip, es el método MAS5.0 que es el que proporciona el software de Affymetrix. Muchos trabajan con los datos que directamente les da este software y por tanto es interesante conocerlo. Lo tratamos en § 4.7. El segundo es el método RMA. Este procedimiento trabaja simultáneamente con todos los chips del experimento. Es pues un procedimiento multiarray o multichip. Este método lo estudiamos en § 4.8.

El objetivo ahora es considerar procedimientos que corrigen las variabilidades entre los microarrays intentando conseguir un valor de expresión asociado a cada gen sin esos ruidos de carácter técnico.

<sup>79</sup> Es especialmente interesante consultar la viñeta `builtInMethods.pdf` que aparece en la ayuda del paquete *affy*. Como material complementario es de interés [60, capítulo 2].

## 4.7 Método MAS5

Veremos cómo pasar de los datos a nivel de sonda a los valores de expresión del gen. Mucho de lo que luego analizamos depende de cómo se realiza este paso. Lo que se comenta es específico para Affymetrix GeneChip. En cada localización tendremos sondas que son oligonucleótidos específicos de un gen. De hecho son pares de sonda: una con la secuencia deseada y otra en la que se modifica la base en la posición 13. La intensidad (o expresión) observada con el oligonucleótido correcto le llamamos PM (perfect matching) y el observado sobre el modificado le llamamos MM (miss matching). Se pensaba que si a PM le restábamos MM corregiríamos el ruido de fondo. Esto no es cierto, una cierta proporción de localizaciones muestran un valor MM mayor que el valor PM, lo que en principio va en contra de lo que lógicamente debería producirse.

Los datos y la información de un Affymetrix GeneChip se almacenan en distintos ficheros. En el fichero CEL tenemos la intensidad media de los pixels de la sonda, la desviación estándar e información de dónde se sitúa la sonda dentro del array. Cada array lleva asociado un fichero CDF. Para poder interpretar el significado biológico de las sondas se necesita un mapa indicando a qué gen corresponde cada gen. Esta información está en el fichero CDF. Este fichero depende del chip utilizado y no varía para distintas muestras.

En esta sección explicamos el método MAS5.0 y nos basamos en el manual [http://media.affymetrix.com/support/technical/whitepapers/sadd\\_whitepaper.pdf](http://media.affymetrix.com/support/technical/whitepapers/sadd_whitepaper.pdf).<sup>3</sup>

Empezamos a trabajar a partir del fichero .CEL que ha sido generado por el software Microarray Suite.

### 4.7.1 Corrección de fondo

Se pretende determinar unos valores para restarlos a las intensidades originales en cada celda.

**Determinar valores por zonas** 1. Se divide el array en K zonas rectangulares:  $Z_k$  con  $k = 1, \dots, K$ .

2. En cada  $Z_k$  se ordenan las intensidades observadas en las distintas celdas y se considera el 2% con intensidades menores. Se toma como medida de fondo la media muestral de

<sup>3</sup>Como casi todos los manuales técnicos no es prodigo de claridad precisamente.

las intensidades previamente consideradas. Denotamos este valor como  $b(Z_k)$ .

3. Del mismo modo que en punto anterior, se considera en cada  $Z_k$  ese 2% de celdas con intensidades menores y calcular su desviación estándar que denotamos  $s(Z_k)$ .

**Suavizando el ajuste** Sea la celda localizada en  $(x, y)$  y vamos a denotar por  $d_k(x, y)$  la distancia desde  $(x, y)$  al centro de la región  $Z_k$ . Consideramos la siguiente cantidad

$$w_k(x, y) = \frac{1}{d_k^2(x, y) + s_0}$$

donde  $s_0$  nos evita que el cociente se pueda anular y por defecto toma el valor  $s_0 = 100$ . Para la celda localizada en  $(x, y)$  consideramos

$$b(x, y) = \frac{1}{\sum_{k=1}^K w_k(x, y)} \sum_{k=1}^K w_k(x, y) b(Z_k)$$

**Corrección del ruido** Ahora vamos a bajar las intensidades restándole una medida de ruido de fondo local. El problema que se puede producir es que nos encontramos con valores negativos. Este valor de ruido en la celda localizada en  $(x, y)$  se calcula como

$$n(x, y) = \frac{1}{\sum_{k=1}^K w_k(x, y)} \sum_{k=1}^K w_k(x, y) s(Z_k).$$

La idea ahora es quitar a las intensidades originales el fondo ( $b(x, y)$ ) que hemos calculado pero asegurándonos que no nos surja valores negativos. La intensidad original se denota como  $I(x, y)$  en la localización  $(x, y)$ . El valor ajustado de la intensidad viene dado por

$$A(x, y) = \max\{I(x, y) - b(x, y), f_0 n(x, y)\}$$

El parámetro  $f_0$  se suele fijar en  $f_0 = 0.5$ .<sup>4</sup>

#### 4.7.2 Cálculo del valor de expresión

Por el procedimiento descrito anteriormente podemos ajustar los valores de PM y de MM. En lo que sigue se supone que hemos ajustado estos valores.

Lo esperable y correcto es que en una celda tengamos un valor de PM mayor que el correspondiente valor MM. Esto no es así en muchas ocasiones como hemos visto. Se define un valor IM (Ideal Mismatch)<sup>5</sup> que intenta corregir este problema.

Denotamos por  $(PM_{ij}, MM_{ij})$  el valor de PM y de MM de la  $j$ -ésima sonda perteneciente al  $i$ -ésimo conjunto de sondas (formado por todas las sondas asociadas a un mismo gen).

Se calcula un valor que llaman background específico para cada conjunto de sondas.

$$B_i = T_{bi}\{\log_2(PM_{ij}) - \log_2(MM_{ij}) : j = 1, \dots, n_i\},$$

---

<sup>4</sup>En el manual hay una igualdad incomprendible que dice  $I(x, y) \max\{I(x, y), 0.5\}$ . En fin, ya veremos qué es exactamente lo que quieren hacer. He puesto lo que sensatamente parece que quieren hacer.

<sup>5</sup>Un apaño se decía antes.

donde  $T_{bi}$  denota el algoritmo bipesado en un solo paso ([§ 31.1](#)). Básicamente un promedio robusto de las diferencias indicadas que no son más que logaritmos de sus cocientes. ¿Qué estima este valor  $B_i$  para el conjunto de sondas i? Si el valor de

$$IM_{ij} = \begin{cases} MM_{ij} & \text{si } MM_{ij} < PM_{ij} \\ \frac{PM_{ij}}{\frac{2^{B_i}}{PM_{ij}}} & \text{si } MM_{ij} \geq PM_{ij} \text{ y } B_i > \alpha \\ \frac{\alpha}{2^{1+(\alpha-B_i)/\beta}} & \text{si } MM_{ij} \geq PM_{ij} \text{ y } B_i \leq \alpha \end{cases}$$

Por defecto se toma  $\alpha = 0.03$  y  $\beta = 10$ .

Ahora calculamos

$$V_{ij} = \max\{PM_{ij} - IM_{ij}, d\}$$

siendo el valor por defecto para d,  $d = 2^{-20}$ .

Transformamos este valor con

$$\mathbb{V}_{ij} = \log_2(V_{ij})$$

con  $j = 1, \dots, n_i$ . Y volvemos a aplicar un estimador biponderado en un solo paso ([§ 31.1](#)).

$$SignalLogValue_i = T_{bi}(\mathbb{V}_{i1}, \dots, \mathbb{V}_{in_i}).$$

Fijamos una señal objetivo, Sc, que por defecto se toma  $Sc = 500$ . Se calcula el siguiente factor de escala para el grupo de sondas i.

$$sf_i = \frac{Sc}{MediaAjustada\{2^{SignalLogValue_i}, 0.02, 0.98\}}$$

En la anterior expresión MediaAjustada indica la media ajustada (media de los valores quitando una cierta proporción de los más grandes y de los más pequeños) en donde se quita el 2% de los más grandes y el 2% de los más pequeños.

El valor final para el conjunto de sondas i es

$$ReportedValue(i) = sf_i \times 2^{SignalLogValue_i}$$

Se incorpora una posibilidad de normalización que no indicamos.

El método **plier** no lo tratamos en estas notas. Está implementado en el paquete [[74](#), plier].

## 4.8 Robust multichip average (RMA)

Una referencia con muchos más detalles de los que vemos en esta sección es [[18](#), pág. 41-59] En este método trabajamos con todos los arrays simultáneamente<sup>80</sup> a diferencia del método anterior donde se procesaba cada array independientemente. El procedimiento solamente utiliza los valores PM y no utiliza para nada los valores MM.

<sup>80</sup> Y de ahí su nombre **Robust Multiarray Average**.

### 4.8.1 Corrección de fondo

Se empieza calculando una corrección de fondo.<sup>81</sup> Asumimos (sugrido por la distribución empírica observada en las intensidades de las sondas) que los valores PM pueden considerarse como la suma de una

<sup>81</sup> Que esencialmente no se ha publicado. Aparece una descripción más o menos vaga en [[76](#)].

variable  $Y$  con distribución normal con media  $\mu$  y varianza  $\sigma^2$  (que modeliza el ruido) y de una variable  $X$  con distribución exponencial con media  $\alpha$  (que modelizaría la señal). Por tanto el valor observado aleatorio  $S$  sería

$$S = X + Y$$

con  $X \sim Exp(\alpha)$  e  $Y \sim N(\mu, \sigma^2)$ . La normal se trunca en cero para evitar valores negativos. Si  $S$  denota la intensidad aleatoria en la sonda entonces, para un valor observado  $S = s$ , se tiene

$$E(X|S=s) = a + b \frac{\phi(\frac{a}{b})}{\Phi(\frac{a}{b})}$$

siendo  $a = s - \mu - \sigma^2 \alpha$  y  $b = \sigma$  mientras que  $\phi$  y  $\Phi$  denotan las funciones de densidad y de distribución de una normal estándar (con media 0 y varianza 1). Los parámetros del modelo  $(\alpha, \mu, \sigma^2)$  se estiman mediante un procedimiento no paramétrico. Utilizando las intensidades observadas en las sondas PM se estima la moda de la distribución. Aquellos valores por encima de la moda son utilizados para estimar  $\alpha$  y los puntos por debajo de la moda se utilizan para estimar los parámetros  $\mu$  y  $\sigma^2$ .

#### 4.8.2 Normalización de cuantiles

En este paso se pretende conseguir que las distribuciones empíricas de las expresiones a nivel de sonda de los distintos arrays sean la misma.<sup>6</sup>

<sup>82</sup> En § 21.3.6 se recuerda el concepto de distribución empírica.

<sup>83</sup> En § 21.3.6 se recuerda el concepto de distribución empírica.

<sup>82</sup> Esencialmente lo que buscamos es que las expresiones que se observan en los distintos microarrays sean las mismas y con las mismas frecuencias.

La normalización que se aplica en este método es una *normalización de cuantiles*.<sup>83</sup>

[18].

Veamos qué es con un ejemplo detallado y sencillo.

- Nuestro experimento tiene dos conjuntos de sondas y cada una con tres sondas. Además suponemos que tenemos simplemente tres chips. En la siguiente tabla recogemos las expresiones.

Conjunto de sondas	Sonda	Chip 1	Chip 2	Chip 3
1	1	7	9	19
1	2	3	5	14
1	3	2	6	11
2	1	4	8	8
2	2	10	11	16
2	3	12	10	15

- Determinamos los valores mayores en cada chip y calculamos el promedio.

```
(12 + 11 + 19) / 3
```

```
## [1] 14
```

Sustituimos los valores originales por estos promedios.

---

<sup>6</sup> Recordemos que en este método solamente utilizamos las expresiones de las sondas PM.

Conjunto de sondas	Sonda	Chip 1	Chip 2	Chip 3
1	1	7	9	<b>14</b>
1	2	3	5	14
1	3	2	6	11
2	1	4	8	8
2	2	10	<b>14</b>	16
2	3	<b>14</b>	10	15

3. Consideramos los segundos valores mayores en cada chip y los promediamos.

$$(10 + 10 + 16) / 3$$

```
## [1] 12
```

Sustituimos los valores originales por los promedios.

Conjunto de sondas	Sonda	Chip 1	Chip 2	Chip 3
1	1	7	9	14
1	2	3	5	14
1	3	2	6	11
2	1	4	8	8
2	2	<b>12</b>	14	<b>12</b>
2	3	14	<b>12</b>	15

4. Consideramos los terceros valores mayores en cada chip.

$$(7 + 9 + 15)/3$$

```
## [1] 10.33333
```

Sustituimos los valores originales por los promedios.

Conjunto de sondas	Sonda	Chip 1	Chip 2	Chip 3
1	1	<b>10.33</b>	<b>10.33</b>	14
1	2	3	5	14
1	3	2	6	11
2	1	4	8	8
2	2	12	14	12
2	3	14	12	<b>10.33</b>

5. Y así sucesivamente hasta llegar al sexto valor que ya es el más pequeño en cada chip. Los datos finales son los de la siguiente tabla.

Conjunto de sondas	Sonda	Chip 1	Chip 2	Chip 3
1	1	10.33	10.33	14
1	2	6.66	5	8.66
1	3	5	6.66	6.66
2	1	8.66	8.66	5
2	2	12	14	12
2	3	14	12	10.33

Una vez ilustrado el método es sencillo de explicar.

1. Tomamos n vectores de longitud N y construimos la matriz X  $N \times n$  que los tiene por vectores columna.

2. Ordenamos cada columna de  $X$  separadamente y tenemos  $X_s$ .
3. Calculamos la media por filas de la matriz  $X_s$  y creamos  $X'_s$  una matriz con la misma dimensión que  $X$ , tal que en la fila  $j$  tenemos la media de la fila de  $X_s$  repetida  $n$  veces.
4. Obtenemos  $X_t$  en donde cada columna de  $X'_s$  recupera el orden original.

¿Qué estamos haciendo desde el punto de vista probabilístico? Hemos considerado los cuantiles muestrales en cada columna. Hemos considerado la media muestral de estos cuantiles muestrales. Supongamos que  $F$  es la función de distribución de estas medias de cuantiles muestrales. Tomamos  $G$  la función de distribución muestral o empírica de las expresiones de cada uno de los microarrays. Si la expresión original la denotamos por  $y_i$  para la  $i$ -ésima sonda entonces la nueva expresión de dicha sonda viene dada por  $y'_i = F^{-1}(G(y_i))$ . Cuando  $F$  corresponde a la uniforme en el intervalo  $[0, 1]$  esto recibe el nombre de transformación integral de la probabilidad.

### 4.8.3 Resumen

Ya hemos realizado la corrección de fondo ([§ 4.8.1](#)). A los valores obtenidos les hemos aplicado una normalización de cuantiles ([§ 4.8.2](#)). Ahora nos queda obtener el resumen de las distintas sondas dentro de cada conjunto (de sondas) y para cada microarray. Se aplica el método median polish de Tukey.<sup>84</sup> Previamente necesitamos algo de notación. En cada array tendremos  $N$  sondas y suponemos que tenemos  $n$  arrays. Por tanto la matriz de expresión a nivel de sonda será:  $x = [x_{ij}]_{i=1,\dots,N; j=1,\dots,n}$ . La sonda  $i$ -ésima en el array  $j$ -ésimo tiene una expresión o intensidad  $x_{ij}$ . Las sondas las suponemos agrupadas en grupos correspondientes a un mismo gen. Denotamos el grupo  $k$ -ésimo de sondas con  $S_k$ . Por tanto,  $S_k \subset \{1, \dots, N\}$ . Por ejemplo, el conjunto  $S_k = \{i_1, \dots, i_{|S_k|}\}$ , es decir, corresponde con las filas  $\{i_1, \dots, i_{|S_k|}\}$  de la matrix  $x$  original. Por simplificar la notación tomaremos  $y_{rj} = x_{i_r, j}$  y por tanto

$$y = [y_{ij}]_{i=1,\dots,|S_k|, j=1,\dots,n} = [x_{ij}]_{i \in S_k, j=1,\dots,n}.$$

En resumen, la matriz  $y$  simplemente corresponde con las filas de la matrix  $x$  correspondiente al grupo de sondas  $S_k$  y todas las columnas.

En lo que sigue nos referimos al grupo  $k$ -ésimo de sondas y lo que se asume es para cada grupo. Asumimos el siguiente modelo:

$$\log_2(y_{ij}) = \mu + \theta_j + \alpha_i + \epsilon_{ij} \quad (4.1)$$

con las siguientes restricciones

1. la mediana de  $\{\theta_j : j = 1, \dots, n\}$  es nula,
2. la mediana de  $\{\alpha_i : i = 1, \dots, |S_k|\}$  es nula
3. la mediana de  $\{\epsilon_{ij} : j = 1, \dots, n\}$  sea nula para cada  $i$ ,
4. la mediana de  $\{\epsilon_{ij} : i = 1, \dots, |S_k|\}$  sea nula para cada  $j$ .

El método median polish es un procedimiento para estimar los parámetros anteriores que aparecen en las ecuaciones 4.1. ¿Cómo? Consideramos, para cada grupo de sondas (por ejemplo, el  $k$ -ésimo  $S_k$ ), la

<sup>84</sup> § 31.2.

matriz

$$\begin{array}{ccc|c} \delta_{1,1} & \cdots & \delta_{1,n} & a_1 \\ \vdots & \ddots & \vdots & \vdots \\ \delta_{|S_k|,1} & \cdots & \delta_{|S_k|,n} & a_{|S_k|} \\ \hline b_1 & \cdots & b_n & m \end{array}$$

En esta matriz las sondas corresponden a las filas y las columnas a los arrays. Además añadimos una columna y una fila (las últimas separadas por una línea continua) en la que aparecen los efectos de fila y columna. Aplicamos el siguiente procedimiento iterativo.

1. Fijamos  $\delta_{ij} = \log_2(y_{ij})$ ,  $a_i = b_i = 0 \forall i, j$ .
2. Calculamos la mediana para cada fila a lo largo de las columnas ignorando la última columna (separada por la línea).
3. Restamos a cada elemento de la fila la mediana correspondiente. Esta mediana se suma a la última columna (formada por  $a_1, \dots, a_{|S_k|}, m$ ).
4. Calculamos la mediana para cada columna de los valores correspondientes a las distintas filas sin considerar la última fila.
5. Restamos la mediana calculada en el paso anterior a cada elemento de la columna exceptuando la última fila. A la última fila sumamos las medianas calculadas.
6. El proceso descrito en los cuatro pasos anteriores continua iterando sucesivamente entre filas y columnas hasta que los cambios que se producen son nulos o muy pequeños.
7. Una vez finalizado el proceso iterativo tendremos  $\hat{\mu} = m$ ,  $\hat{\theta}_j = b_j$  y  $\hat{a}_i = a_i$ . El valor  $\delta_{ij}$  será el estimador de  $\epsilon_{ij}$ :  $\hat{\epsilon}_{nij} = \delta_{ij}$ .

Una vez estimados estos parámetros los estimadores (en escala logarítmica en base 2) de la expresión del grupo de sondas  $k$  en el array  $j$  viene dada por

$$\hat{\mu} + \hat{\theta}_j.$$

Es un método que no nos proporciona estimaciones de error de los estimadores. Es muy rápido de implementar. Al utilizar medianas es menos sensibles a observaciones extremas. Sin embargo, el resultado puede ser distinto según empecemos por filas o columnas.

## 4.9 MAS5 y RMA con affy

En el paquete [75, affy] tenemos implementados los dos procedimientos comentados en las dos secciones previas. Usamos las función affy::rma y affy::mas5.

```
gse21779_rma = affy::rma(gse21779raw)
gse21779_mas5 = affy::mas5(gse21779raw)
```

```
save(gse21779_rma, file=paste(dirTamiData, "gse21779_rma.rda", sep=""))
save(gse21779_mas5, file=paste(dirTamiData, "gse21779_mas5.rda", sep=""))
```

```
load(paste(dirTamiData, "gse21779_rma.rda", sep=""))
load(paste(dirTamiData, "gse21779_mas5.rda", sep=""))
```

En las figuras 4.5(a) y 4.5(b) tenemos los estimadores kernel de densidad de los niveles de expresión para los distintos chips una vez hemos aplicado el método RMA y MAS5. En las figuras 4.5(c) y 4.5(d) mostramos los diagramas de cajas correspondientes a los mismos datos preprocesados.

```
library(geneplotter)
geneplotter::multidensity(exprs(gse21779_rma))
geneplotter::multidensity(exprs(gse21779_mas5))
graphics::boxplot(exprs(gse21779_rma))
graphics::boxplot(exprs(gse21779_mas5))
```

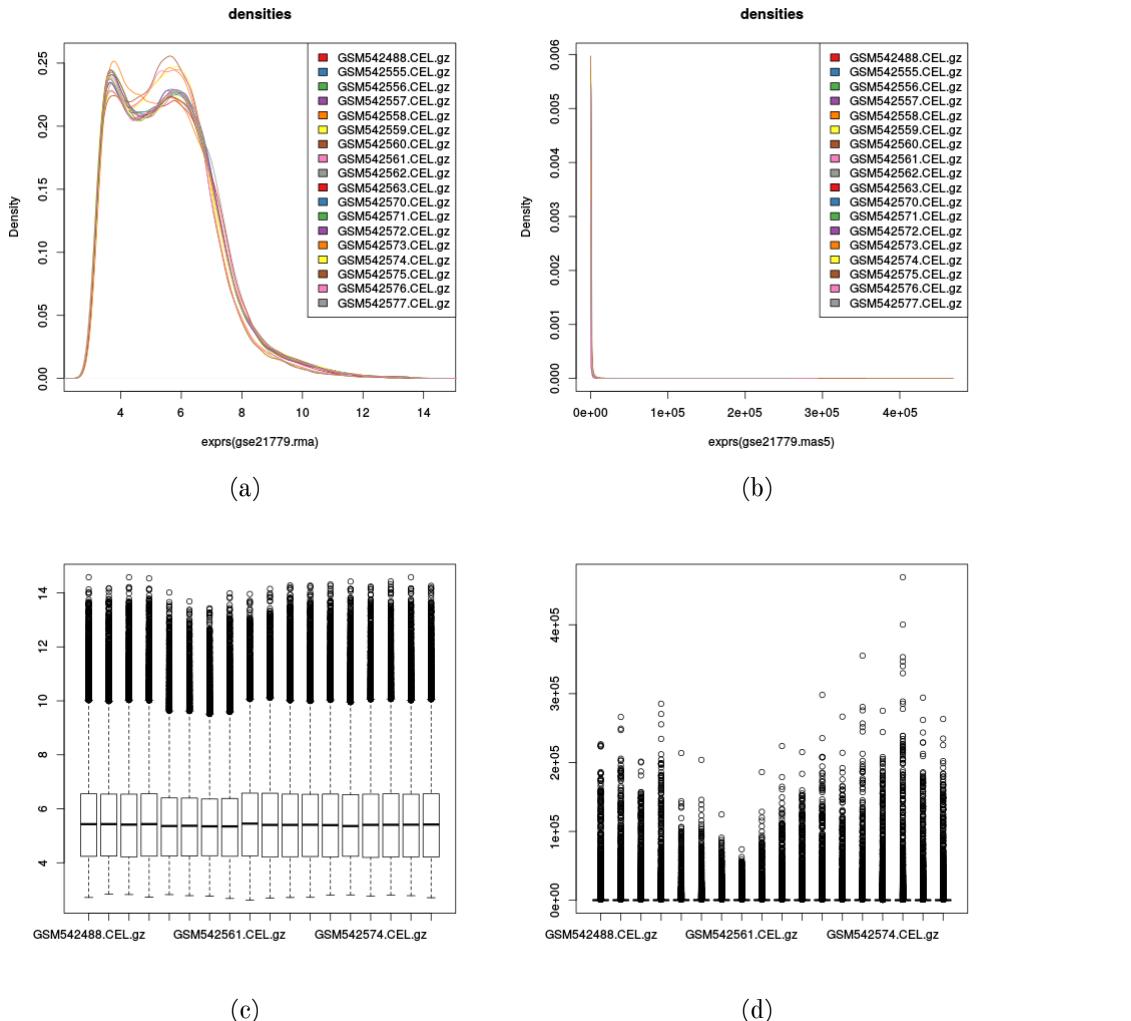


Figura 4.5: a) Estimadores kernel de densidad para los chips una vez hemos aplicado el procedimiento RMA a los datos gse21779. b) Lo mismo con el método MAS5. c) Boxplot para los niveles de expresión de los datos gse21779 con método RMA. d) Lo mismo que el punto (c) pero con el método MAS5.

## 4.10 Otros métodos posibles

Hemos visto dos métodos. Sin embargo, podemos considerar métodos distintos con el paquete [75, affy] ¿Qué métodos proporciona el paquete affy? Empezamos con los métodos de corrección de fondo. Las opciones son

```
bgcorrect.methods()

## [1] "bg.correct"      "mas"        "none"
## [4] "rma"
```

Las opciones que nos muestra corresponden con:

**none** En la opción *none* no hacemos nada. Dejamos los valores originales.

**mas** Utiliza el procedimiento que hemos visto en MAS5.0.

**RMA** Utiliza el procedimiento visto en algoritmo RMA.

Nos ofrece los siguientes métodos de normalización.

```
normalize.methods(gse21779raw)

## [1] "constant"          "contrasts"
## [3] "invariantset"      "loess"
## [5] "methods"           "qspline"
## [7] "quantiles"         "quantiles.robust"
## [9] "quantiles.probeset" "scaling"
```

En particular, para los chip de Affymetrix, tenemos distintas opciones para corregir los valores PM.

```
pmcorrect.methods()

## [1] "mas"       "methods"   "pmonly"
## [4] "subtractmm"
```

Los métodos para resumir los valores de los conjuntos de sondas en un único valor de expresión del gen los podemos ver con el siguiente código.

```
express.summary.stat.methods()

## [1] "avgdiff"     "liwong"      "mas"
## [4] "medianpolish" "playerout"
```

La función *expresso* permite realizar los distintos pasos del preprocess. De hecho, la función **mas5** que hemos utilizado anteriormente no es más que una llamada a esta función.

```
mas5

## function (object, normalize = TRUE, sc = 500, analysis = "absolute",
##           ...)
## {
```

```
##     res <- expresso(object, bgcorrect.method = "mas", pmcorrect.method = "mas",
##                       normalize = FALSE, summary.method = "mas", ...)
##     if (normalize)
##         res <- affy.scalevalue.exprSet(res, sc = sc, analysis = analysis)
##     return(res)
## }
```

Una explicación (bastante) detallada de cada una de las opciones la podemos encontrar en la vignette *builtinMethods.pdf* del paquete [75, affy].

## 4.11 Cómo hacer un preprocessado sencillo a partir de datos de expresión

En la sección anterior hemos tratado cómo preprocessar datos partiendo de datos a nivel de sonda muy orientado a Affymetrix. Cuando conseguimos datos de expresión (por ejemplo de GEO) en ocasiones no tenemos los datos a nivel de sonda (los .CEL en el caso de Affymetrix GeneChip) sino que tenemos unos datos de expresión con un preprocessado que no controlamos exactamente. El problema es que estos datos todavía necesitan un procesado para homogeneizarlos. ¿Cómo hacerlo de un modo sencillo? Una opción rápida la tenemos con la función limma::normalizeBetweenArrays.

**Ejemplo 4.2 (GSE1397)** Veamos un ejemplo con los datos GSE1397. En § 5.10 hemos visto cómo bajarlos de GEO y cómo construir el ExpressionSet.

```
load(paste(dirTamiData, "gse1397raw.rda", sep=""))
```

Son datos Affymetrix correspondientes al chip

```
annotation(gse1397raw)
## [1] "GPL96"
```

Supongamos que queremos aplicar una normalización de los cuantiles.<sup>85</sup> Podemos hacer

```
library(limma)
exprs1 = normalizeBetweenArrays(exprs(gse1397raw), method = "quantile")
```

O bien podemos hacer que coincidan las medianas de los distintos arrays con

```
exprs1 = normalizeBetweenArrays(exprs(gse1397raw), method = "scale")
```

En la ayuda de limma::normalizeBetweenArrays podemos consultar otras opciones.

## 4.12 Lecturas complementarias

El texto [131] es una excelente referencia sobre lo tratado en este tema. Mucho del material ha sido extraido de este libro.

<sup>85</sup> Ver § 4.8.2 y [19].

# Capítulo 5

## Datos de microarrays

### 5.1 Introducción

Comentaremos los bancos de datos que utilizamos en el resto del curso. Son datos de expresión de genes obtenidos utilizando microarrays. Algunos van incorporados en los paquetes de R/Bioconductor como los datos golub (en [111]) o los datos ALL (en [87]). Se utilizan como ejemplos en muchos paquetes y en artículos. Es interesante analizarlos y conocerlos. 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. Veremos también cómo bajar datos de repositorios públicos.

### 5.2 sample.ExpressionSet

Los datos Biobase::sample.ExpressionSet. El experimento tiene 26 muestras y 500 genes. Sobre las muestras conocemos tres variables (o covariables): sex, type (caso y control) y score. La última covariable no es categórica, es una covariable continua. Estos datos de expresión están almacenados en un objeto de clase ExpressionSet. Para poder utilizar esta estructura de datos hemos de cargar el paquete [59, Biobase].

```
library(Biobase)
```

Cargamos los datos.

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

Podemos ver ayuda sobre los mismos con

```
help(sample.ExpressionSet)
```

Las variables fenotípicas o metadatos que nos describen las muestras tienen las siguientes etiquetas.

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

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

Una variable dada, por ejemplo `type`, la tendremos con

```
sample.ExpressionSet$type

## [1] Control Case     Control Case     Case
## [6] Control Case     Case     Case     Control
## [11] Case     Control Case     Case     Case
## [16] Control Case     Control Case     Case
## [21] Control Control Control Control Case
## [26] Case
## Levels: Case Control
```

Y todos los datos fenotípicos se obtienen con

```
phenoData(sample.ExpressionSet)

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

La matriz de expresión se obtiene con (no la mostramos)

```
exprs(sample.ExpressionSet)
```

Para tener información sobre un objeto de clase `ExpressionSet` es recomendable consultar [69, capítulo 2]. Esto nos va a permitir trabajar con los datos de expresión y los metadatos que nos dan una descripción completa del experimento. En la clase `ExpressionSet` tendremos:

**assayData** Los datos de expresión de los distintos microarrays que componen toda la experimentación.

**Metadatos** En particular tendremos `phenoData` que nos dan información sobre las muestras utilizadas. Las características del chip vienen dadas en `featureData`. Las anotaciones de los genes las tendremos en `annotation`.

En la viñeta [59, Biobase] tenemos una buena descripción de la creación y manipulación de un `ExpressionSet`. Por ejemplo podemos saber la anotación de estos datos con

```
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
```

### 5.3 Datos golub

Una buena explicación de estos datos aparece en el manual de uso del paquete [111, multtest]. Estos datos fueron analizados por primera vez en [62]. En estas notas se utilizan en dos versiones. La primera es la incluida en el paquete [111]. La segunda corresponde a los datos incluidos en el paquete [63]. En el manual hablaremos de multtest::golub o bien golubEsets::golub\_Train.<sup>86</sup>

Hay muestras correspondientes a dos grupos o condiciones: 27 muestras (las primeras 27 columnas) correspondientes a pacientes con leucemia linfoblástica aguda (ALL, la clase 0) y leucemia mieloide aguda (AML, clase 1). Para medir los niveles de expresión se utilizaron chips de alta densidad de oligonucleótidos Affymetrix para 6817 genes. Los datos los podemos obtener a partir del paquete [111] con

```
data(golub, package= "multtest")
```

Según se indica en el manual de [111, multtest] se realizó una selección previa de genes de carácter no específico (no se utiliza por tanto información de las dos condiciones que pretendemos diferenciar). Esta selección no específica consistió en los siguientes pasos. Se trabajó con los datos normalizados que se pueden encontrar en <http://www.broadinstitute.org>. Allí podemos encontrar una detallada explicación del protocolo experimental utilizado. Una vez obtenidos los datos normalizados se realizó la siguiente selección previa.

1. Una umbralización de los niveles de expresión. Valores por debajo de 100 se les asignó el valor 100. Los valores superiores a 16000 se les asignó el valor 16000.
2. Se calculó para cada gen el valor máximo y el valor mínimo de sus niveles de expresión para todas las muestras analizadas. Si denotamos por  $m_1$  el mínimo y por  $m_2$  el máximo entonces se eliminaron los genes que verifican

$$\frac{m_2}{m_1} \leq 5$$

o bien que

$$m_2 - m_1 \leq 500$$

3. Se tomó el logaritmo en base 10 para los niveles de expresión de los genes que no fueron eliminados.

<sup>86</sup> Son unos datos antiguos pero que aparecen en numerosas publicaciones y viñetas de paquetes de R.

Finalmente se estandarizaron los niveles de expresión dentro de cada array. Es decir, se calculó para cada array la media y la desviación estándar de todos los niveles observados en este array. A los datos originales se les restó la media y se dividió por la desviación estándar. Estos son los datos finales con los que trabajamos.<sup>1</sup> Una vez hemos cargado los datos golub tenemos la siguiente información:

**golub** Es una matriz  $3051 \times 38$  de niveles de expresión donde las 3051 filas corresponden a los genes y las 38 columnas corresponden a las muestras. Podemos ver su primera fila:

```
golub[1,]

## [1] -1.45769 -1.39420 -1.42779 -1.40715 -1.42668
## [6] -1.21719 -1.37386 -1.36832 -1.47649 -1.21583
## [11] -1.28137 -1.03209 -1.36149 -1.39979  0.17628
## [16] -1.40095 -1.56783 -1.20466 -1.24482 -1.60767
## [21] -1.06221 -1.12665 -1.20963 -1.48332 -1.25268
## [26] -1.27619 -1.23051 -1.43337 -1.08902 -1.29865
## [31] -1.26183 -1.44434  1.10147 -1.34158 -1.22961
## [36] -0.75919  0.84905 -0.66465
```

**golub.gnames** Es una matriz  $3051 \times 3$  con identificadores de los genes. Veamos la información de la primera fila correspondiente al primer gen.

```
golub.gnames[1,]

## [1] "36"
## [2] "AFFX-HUMISGF3A/M97935_MA_at (endogenous control)"
## [3] "AFFX-HUMISGF3A/M97935_MA_at"
```

**golub.cl** Es un vector de longitud 38 con la clasificación del tipo de leucemia.

```
golub.cl

## [1] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [24] 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

Con frecuencia, y con objeto de mejorar la presentación de los resultados ejecutaremos la siguiente línea de código.

```
golub.cl = factor(golub.cl, levels= 0:1, labels=c("ALL", "AML"))
```

Si volvemos a ver el contenido de golub.cl veremos el cambio producido.

---

<sup>1</sup> Es importante darse cuenta de la enorme cantidad de transformaciones de la información original que estamos haciendo y que, por lo tanto, afecta de un modo fundamental las conclusiones que obtengamos en cualquier tratamiento estadístico posterior. Y lo peor: no sabemos de un modo claro cómo afectan estos resultados.

```
golub.cl

## [1] ALL ALL
## [12] ALL ALL
## [23] ALL ALL ALL ALL ALL AML AML AML AML AML
## [34] AML AML AML AML AML
## Levels: ALL AML
```

87

En el paquete [63] tenemos los datos originales. Tanto las muestras que se utilizaron para entrenar el procedimiento como las que se utilizaron para evaluarlo.

## 5.4 Datos ALL

Para una versión ampliada de esta sección podemos consultar [69, 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 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 [75, affy] con la función affy::rma y están almacenados en forma de un Biobase::ExpressionSet. Empezamos cargando los datos.

```
library(Bioconductor)
library(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.

<sup>87</sup> El script de R con todo el preprocesado descrito podemos encontrarlo en <http://svitsrv25.epfl.ch/R-doc/library/multtest/doc/golub.R> y en la documentación del paquete [111].

## 5.5 Construyendo un ExpressionSet

Supongamos (y no es mucho suponer) que tenemos los datos de expresión en un fichero, en otro fichero tenemos la descripción de las distintas muestras que hemos analizado, en otro fichero (o en la red) tenemos información sobre los genes (o sondas) que tenemos en las filas de la matriz de expresión. Tenemos información sobre las distintas muestras que componen toda la experimentación, bajo qué condiciones se consiguieron. Lo que podemos llamar la descripción fenotípica de las muestras. Y finalmente sabemos cosas sobre toda la experimentación realizada: autores, publicación principal que se derivó, objetivo de la experimentación. Y, como es habitual, lo tenemos todo por separado, en distintos ficheros. La clase Biobase::ExpressionSet tiene como objetivo tenerlo todo contenido en un mismo objeto. En esta sección vemos cómo construir este ExpressionSet a partir de la información dispersa. En ocasiones esto nos puede pasar si bajamos la información de alguna publicación en la que nos aparece como información suplementaria.

Es probable que en nuestro trabajo no tengamos que realizar todo el proceso que indicamos a continuación pero con mucha probabilidad al menos una parte sí.

### ExpressionSet

Es una estructura de datos que nos permite mantener en un solo objeto toda la información que relativa a un experimento con microarrays.

Para poder utilizar esta clase necesitamos el paquete [59, Biobase]. Lo cargamos (la clase y los métodos que luego vamos a utilizar).

```
pacman::p_load(Biobase)
```

### ¿Qué necesitamos?

Supongamos que tenemos la información por separado y queremos construir un ExpressionSet. ¿Qué datos nos describen un experimento genómico de alto rendimiento? Podemos distinguir la siguiente información:

1. Los datos del ensayo (los niveles de expresión).
2. Datos sobre las muestras que podemos llamar de un modo genérico *metadatos fenotípicos*.
3. Anotaciones sobre las sondas u otros metadatos.
4. Una descripción del experimento.

Vamos a construir un ExpressionSet construyendo cada una de las componentes a partir de unos datos.<sup>2</sup>

---

<sup>2</sup>Los datos que estamos usando han sido amablemente proporcionados por Joan Climent y están ligeramente modificados respecto de los publicados originalmente en [104].

## Datos del ensayo

Son los valores de expresión posiblemente preprocesados. Es una matriz con N filas y con n columnas. El número N de filas corresponde con el número de características (features) del chip y n con el número de muestras. Leemos los datos (posiblemente con `read.table` o `read.csv`. Nos han pasado los datos.<sup>88</sup> Normalmente las funciones `utils::read.table` o `utils::read.csv` nos suelen resolver el problema.

<sup>88</sup> Como Dios quiere y nosotros hemos de averiguar.

```
pacman::p_load("tamidata")
finput = system.file("extdata", "n06_expr.txt", package="tamidata")
exprs0 = read.table(file = finput, header = TRUE, sep="\t")
```

Cuando usamos estas funciones nos devuelve un `data.frame` como podemos comprobar.

```
class(exprs0)
## [1] "data.frame"
```

En lo que sigue necesitamos que sea una matriz. Lo hacemos (por cierto el `-1` nos sirve para quitar la primera columna de la matriz donde teníamos los nombres de las sondas).

```
exprs0 = as.matrix(exprs0[,-1])
```

Y podemos comprobar que ya lo tenemos.

```
class(exprs0)
## [1] "matrix"
```

¿Cuántas filas (features) y columnas (samples) tenemos?

```
dim(exprs0)
## [1] 22215      54
```

Los nombres de las columnas son

```
colnames(exprs0)
##  [1] "X600MPE"      "AU565"        "BT474"
##  [4] "BT483"        "CAMA1"        "DU4475"
##  [7] "HCC202"        "HCC1428"      "HCC1007"
## [10] "HCC2185"       "LY2"          "MCF7"
## [13] "MDAMB415"      "MDAMB175"     "MDAMB361"
## [16] "MDAMB435"      "MDAMB134"     "SUM185PE"
## [19] "SUM225CWN"     "SUM44PE"      "SKBR3"
## [22] "T47D"          "UACC812"      "ZR75B"
## [25] "ZR751"          "ZR7530"       "SUM52PE"
## [28] "BT20"           "HCC1937"      "HCC70"
## [31] "HCC1187"        "HCC1008"      "HCC1599"
## [34] "HCC3153"        "HCC1569"      "HCC2157"
## [37] "HCC1954"        "HCC38"         "HCC1500"
```

```
## [40] "HCC1143"      "MCF12A"       "MCF10A"
## [43] "MDAMB468"     "SUM149PT"     "SUM190PT"
## [46] "BT549"        "HS578T"       "HBL100"
## [49] "MDAMB231"     "MDAMB435_2"   "MDAMB157"
## [52] "MDAMB436"     "SUM159PT"     "SUM1315"
```

Y podemos ver

```
head(exprs0,n=1)

##      X600MPE    AU565    BT474    BT483    CAMA1
## [1,] 10.11474 10.42933 9.445266 10.16027 10.0863
##      DU4475   HCC202   HCC1428   HCC1007   HCC2185
## [1,] 8.79988 10.32634 9.279211 9.620473 9.620686
##      LY2      MCF7 MDAMB415 MDAMB175 MDAMB361
## [1,] 9.471435 9.87178 10.07334 9.736757 10.20135
##      MDAMB435 MDAMB134 SUM185PE SUM225CWN
## [1,] 9.438713 9.355205 10.18208 10.35122
##      SUM44PE  SKBR3    T47D UACC812    ZR75B
## [1,] 10.28434 9.58074 9.968553 10.68766 9.624339
##      ZR751    ZR7530   SUM52PE    BT20   HCC1937
## [1,] 9.872964 9.843985 9.584688 9.52413 9.881727
##      HCC70    HCC1187   HCC1008   HCC1599   HCC3153
## [1,] 10.53008 10.18941 10.14184 9.933858 9.920917
##      HCC1569   HCC2157   HCC1954   HCC38    HCC1500
## [1,] 9.009182 9.30533 8.945598 9.207611 9.857016
##      HCC1143   MCF12A   MCF10A MDAMB468 SUM149PT
## [1,] 10.35362 9.458165 8.582511 10.12283 9.849195
##      SUM190PT   BT549   HS578T   HBL100 MDAMB231
## [1,] 9.497212 8.225951 7.960181 8.137387 7.625879
##      MDAMB435_2 MDAMB157 MDAMB436 SUM159PT
## [1,] 8.038661 8.320586 7.414636 7.962508
##      SUM1315
## [1,] 7.512639
```

Vamos a etiquetar las filas con los identificadores de las sondas.

```
finput = system.file("extdata","n06_gene_attributes.txt",package="tamidata")
rn = read.table(file = finput,header = TRUE,sep="\t")
rownames(exprs0) = rn[,1]
```

## Datos fenotípicos

Es la información que tenemos de las distintas muestras. Será una matriz de datos (un data.frame) con tantas filas como muestras, n, y con tantas columnas como variables tenemos. Estas variables que nos describen las muestras podemos llamarlas covariables (una denominación muy estadística).

En nuestro caso los datos fenotípicos los tenemos en el fichero `sample_attributes.txt`. Los leemos.

```
finput = system.file("extdata","n06_sample_attributes.txt",
 package="tamidata")
pd0 = read.table(finput,sep = "\t",header=TRUE)
```

Podemos ver el número de muestras (que ha de coincidir con el número de columnas de la matriz de expresión) y de covariables.

```
dim(pd0)
## [1] 54 14
```

Es **necesario** que los nombres de las filas de pd0 coincidan con los nombres de las columnas de exprs0 ya que nos están dando covariables relativas a estas muestras.

```
rownames(pd0) = colnames(exprs0)
```

Esto es fundamental. Si no coinciden los nombres de las columnas de la matriz de expresión con los nombres de las filas de la matriz de datos de las covariables no podremos construir el ExpressionSet.

Y ahora es conveniente un pequeño resumen descriptivo de las covariables.

```
summary(pd0)

##      IDENTIFIER    sample_no      Type
## 600MPE : 1   Min.   : 1.00   B:18
## AU565  : 1   1st Qu.:14.25   L:27
## BT20   : 1   Median  :27.50   M: 9
## BT474  : 1   Mean    :27.50
## BT483  : 1   3rd Qu.:40.75
## BT549  : 1   Max.    :54.00
## (Other):48
##      in_CGH        ALL      CDH1_meth
##  Min.   :0.000   Min.   :1   Del   : 1
##  1st Qu.:1.000   1st Qu.:1   full  : 1
##  Median :1.000   Median :1   no    :14
##  Mean   :0.963   Mean   :1   partial: 4
##  3rd Qu.:1.000   3rd Qu.:1   NA's   :34
##  Max.   :1.000   Max.   :1
##
##      SFRP1        HER2      KIT      known.p53
##  Min.   :2.975   NO   :27   NO   :45   NO   :24
##  1st Qu.:3.160   YES  :11   YES  : 8   YES  :29
##  Median :3.293   NA's:16   NA's: 1   NA's: 1
##  Mean   :3.701
##  3rd Qu.:4.004
##  Max.   :7.780
##  NA's   :1
##      BRCA        order
##  loss     :16   Min.   : 1.00
##  loss_mutant: 3  1st Qu.:14.00
##  mutant   : 1   Median  :27.00
##  WT       : 9   Mean    :27.23
##  NA's     :25   3rd Qu.:40.00
##                  Max.   :54.00
##                  NA's   :1
##      adherence    per3
##  adherent :12   Min.   :4.000
```

```
##  not adherent: 5   1st Qu.:4.000
##  partial       : 6   Median  :5.000
##  NA's          :31   Mean    :5.452
##                           3rd Qu.:5.000
##                           Max.   :9.000
##                           NA's   :23
```

Podemos ver que tenemos covariables categóricas (en R se les llama de clase **factor**) y covariables numéricas. Con las covariables categóricas tenemos las frecuencias mientras que con las numéricas nos da mínimo, primer cuartil (o percentil de orden 25 %), mediana y media, tercer cuartil (o percentil de orden 75 %) y máximo.

Podemos ver los tipos (o clases) de cada columna del siguiente modo.

```
sapply(pd0, class)

## IDENTIFIER  sample_no      Type     in_CGH
## "factor"    "integer"    "factor"  "integer"
##      ALL    CDH1_meth    SFRP1      HER2
## "integer"   "factor"    "numeric"  "factor"
##      KIT    known.p53    BRCA      order
## "factor"   "factor"    "factor"   "integer"
## adherence  per3        "integer"
## "factor"   "integer"
```

Podemos tener información adicional sobre las covariables que nos describen las muestras. Por ejemplo, un nombre de variable como `known.p53` puede no significar demasiado. Pero si una etiqueta como “Presencia de p53”<sup>3</sup> ¿Cómo podemos añadir esta información?

```
metadatos = data.frame(labelDescription = c("Identificador",
"Número de muestra", "Tipo", "En CGH", "ALL", "Método CDH1", "SFRP1",
"HER2", "KIT", "Presencia de p53", "BRCA", "Orden", "Adherencia",
"per3"), row.names=colnames(pd0))
```

Notemos que la columna `labelDescription` debe de estar presente. Utilizamos la clase `Biobase::AnnotatedDataFrame` que nos permite tener los valores de las covariables y los metadatos que acabamos de introducir. Lo podemos hacer del siguiente modo.

```
datosfenotipo = new("AnnotatedDataFrame", data = pd0,
varMetadata = metadatos)
```

Una vez tenemos este `AnnotatedDataFrame` podemos ver los nombres de las muestras

```
sampleNames(datosfenotipo)
```

```
## [1] "X600MPE"      "AU565"       "BT474"
## [4] "BT483"        "CAMA1"       "DU4475"
## [7] "HCC202"        "HCC1428"     "HCC1007"
## [10] "HCC2185"       "LY2"         "MCF7"
```

---

<sup>3</sup>No es un ejemplo muy lucidor.

```
## [13] "MDAMB415"    "MDAMB175"    "MDAMB361"
## [16] "MDAMB435"    "MDAMB134"    "SUM185PE"
## [19] "SUM225CWN"   "SUM44PE"     "SKBR3"
## [22] "T47D"        "UACC812"    "ZR75B"
## [25] "ZR751"       "ZR7530"     "SUM52PE"
## [28] "BT20"         "HCC1937"    "HCC70"
## [31] "HCC1187"     "HCC1008"    "HCC1599"
## [34] "HCC3153"     "HCC1569"    "HCC2157"
## [37] "HCC1954"     "HCC38"      "HCC1500"
## [40] "HCC1143"     "MCF12A"     "MCF10A"
## [43] "MDAMB468"    "SUM149PT"   "SUM190PT"
## [46] "BT549"        "HS578T"     "HBL100"
## [49] "MDAMB231"    "MDAMB435_2" "MDAMB157"
## [52] "MDAMB436"    "SUM159PT"   "SUM1315"
```

O también ver los valores de las covariables (mostramos los primeros solamente)

```
head(pData(datosfenotipo))

##           IDENTIFIER sample_no Type in_CGH ALL
## X600MPE      600MPE          1   L    1   1
## AU565        AU565          2   L    1   1
## BT474        BT474          3   L    1   1
## BT483        BT483          4   L    1   1
## CAMA1        CAMA1          5   L    1   1
## DU4475       DU4475         6   L    1   1
##          CDH1_meth SFRP1 HER2 KIT known.p53 BRCA
## X600MPE      no 3.40987 <NA> NO    NO <NA>
## AU565        <NA> 3.16902 <NA> NO    YES <NA>
## BT474        no 3.16737 YES  NO    NO loss
## BT483        no 2.97456 NO   NO    NO WT
## CAMA1        no 3.16856 <NA> NO    NO WT
## DU4475       no 3.06074 <NA> NO    NO WT
##           order adherence per3
## X600MPE      1      <NA>    9
## AU565        2      <NA>    4
## BT474        3      <NA>    NA
## BT483        4      <NA>    NA
## CAMA1        5      <NA>    NA
## DU4475       6      <NA>    NA
```

## Anotaciones y datos de las características

Sin duda este es el punto fundamental, la descripción de información sobre las filas de la matriz de expresión. Estas filas (features) corresponden con sondas. De hecho, distintas sondas corresponden con un mismo gen. Esta información depende del chip que utilizamos. Por ello distintos experimentos pueden corresponder a un mismo chip o instrumento. Es conveniente consultar <http://www.bioconductor.org/packages/2.11/bioc/html/annotate.html>. La información de cada chip aparece en un paquete de metadatos del mismo. En particular, nos proporcionan el nombre del gen, el símbolo y la localización en el cromosoma. Otros paquetes proporcionan información de GO o KEGG.

## Descripción del experimento

Los experimentos los hace la gente y hay que incluir esta información en los datos.

```
datosexperimento = new('MIAME',name='Neve et alt.',lab='Varios',
  contact ='rmneve@lbl.gov',
  title = 'A collection of breast cancer cell lines for the study of
  functionally distinct cancer subtypes',abstract = 'An example ExpressionSet',
  url = '10.1016/j.ccr.2006.10.008',
  other = list(notes = 'Creado a partir de ficheros de texto'))
```

El formato MIAME (Minimum information about a microarray experiment) puede consultarse en <http://fged.org/projects/miame/>.

## Y montamos las piezas

Ejemplo de ExpressionSet a partir de los distintos elementos que

<sup>89</sup> Esto es un poco como el hemos construido.<sup>89</sup>

Mecano.

```
neve06 = new("ExpressionSet",exprs=exprs0,phenoData = datosfenotipo,
  experimentData = datosexperimento,annotation = "hgu95av2")
```

Guardamos este ExpressionSet para uso posterior.

```
save(neve06,file="neve06.rda")
```

Todo lo hecho en esta sección se puede hacer de un modo más simple con EnrichmentBrowser::read.eset.

## 5.6 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 ExpressionSet que leemos.

```
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))

##                      type
## GSM153907.CEL.gz    wt
## GSM153908.CEL.gz  edc3D
## GSM153909.CEL.gz    wt
## GSM153910.CEL.gz  edc3D
## GSM153911.CEL.gz    wt
## GSM153912.CEL.gz  edc3D
```

Los datos han sido normalizados utilizando el método RMA. En la figura 5.2(a) mostramos las densidades estimadas.

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

En la figura 5.2(b) tenemos los diagramas de cajas.

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

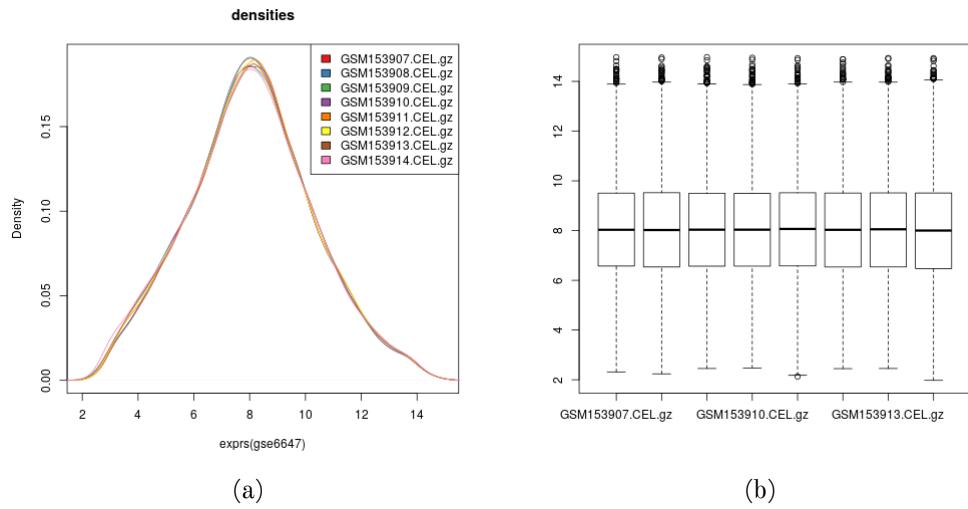


Figura 5.1: Datos gse6647: estimadores de densidad y diagramas de cajas.

Vamos a reproducir los dibujos de la figura 5.2 utilizando el paquete [150, ggplot2].<sup>90</sup> El código es el que sigue y los tenemos en la figura ??.

<sup>90</sup> Sobre el paquete [147, reshape] es interesante consultar [148].

```
library(reshape); library(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) ## Estimadores densidad
ggplot(df1, aes(x=variable, y = value)) + geom_boxplot() +
  coord_flip()
```

## 5.7 De cómo utilizar un ExpressionSet

En esta sección pretendemos mostrar cómo utilizar funciones que manejan ExpressionSet. Vamos a utilizar los datos ALL que aparecen en el paquete [87, ALL]. Cargamos el paquete.

```
library(ALL)
```

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

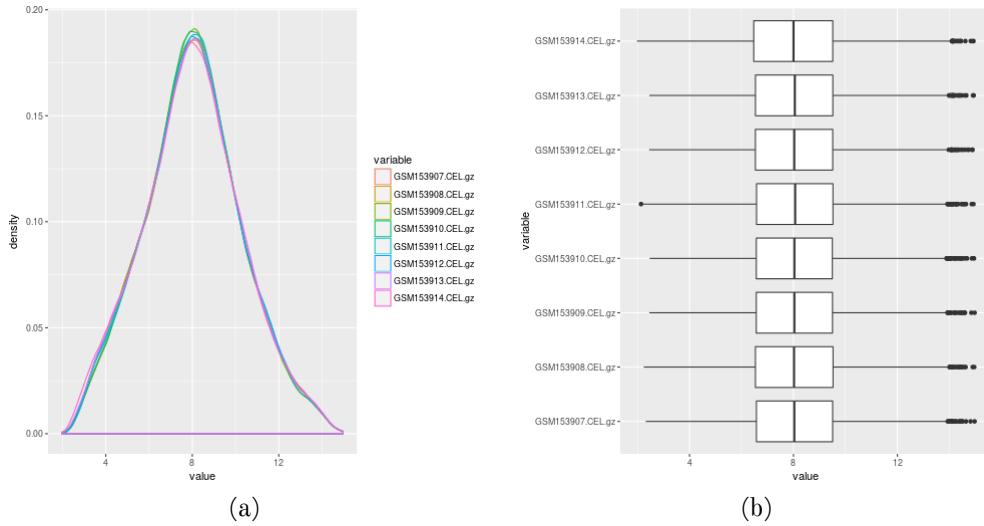


Figura 5.2: Datos gse6647: estimadores de densidad y diagramas de cajas utilizando [150, ggplot2].

```
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 co-variables que nos describen las distintas muestras. Solamente los nombres los podemos ver con

```
names(pData(ALL))

## [1] "cod"           "diagnosis"
## [3] "sex"           "age"
## [5] "BT"            "remission"
## [7] "CR"            "date.cr"
## [9] "t(4;11)"       "t(9;22)"
## [11] "cyto.normal"   "citog"
## [13] "mol.biol"      "fusion protein"
## [15] "mdr"           "kinet"
## [17] "ccr"           "relapse"
## [19] "transplant"    "f.u"
## [21] "date last seen"
```

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

```
ALL$BT

## [1] B2 B2 B4 B1 B2 B1 B1 B2 B2 B3 B3 B3 B2 B3
## [16] B B2 B3 B2 B3 B2 B2 B1 B1 B2 B1 B2 B1 B2
## [31] B B B2 B2 B2 B1 B2 B2 B2 B2 B4 B4 B2 B2
## [46] B2 B4 B2 B1 B2 B2 B3 B4 B3 B3 B4 B3 B3 B1
## [61] B1 B1 B1 B3 B3 B3 B3 B3 B3 B1 B3 B1 B4
## [76] B2 B2 B1 B3 B4 B4 B2 B2 B3 B4 B4 B4 B1 B2 B2
## [91] B2 B1 B2 B B T T3 T2 T2 T3 T2 T T4 T2 T3
## [106] T3 T T2 T3 T2 T2 T1 T4 T T2 T3 T2 T2 T2
## [121] T2 T3 T3 T3 T2 T3 T2 T
## Levels: B B1 B2 B3 B4 T T1 T2 T3 T4
```

Los datos de expresión 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.biol` toma el valor NEG. Determinamos las columnas.

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

Y nos quedamos con esa muestras.

```
ALL[, selcol]

## ExpressionSet (storageMode: lockedEnvironment)
## assayData: 12625 features, 74 samples
##   element names: exprs
## protocolData: none
## phenoData
##   sampleNames: 01010 04007 ... LAL4 (74
##     total)
##   varLabels: cod diagnosis ... date last
##     seen (21 total)
```

```
##   varMetadata: labelDescription
##   featureData: none
##   experimentData: use 'experimentData(object)'
##   pubMedIds: 14684422 16243790
##   Annotation: hgu95av2
```

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

## 5.8 NCBI GEO

GEO es el 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 proteínas. 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. Estas notas son una versión muy extendida de estos análisis.

## 5.9 GSE21779

Hemos elegido los datos que en dicha base de datos tienen el identificador GSE21779. Podemos buscarlos utilizando la página web <http://www.ncbi.nlm.nih.gov/geo/> o bien podemos acceder directamente a esta dirección <http://www.ncbi.nlm.nih.gov/sites/GDSbrowser?acc=GDS4128>. Bajamos los ficheros CEL (están en formato comprimido gzip).

Sabiendo el identificador de los datos y que tienen los datos de expresión originales los podemos bajar del siguiente modo. Primero cargamos el paquete [38, GEOquery].

```
library(GEOquery)
```

Y sabiendo el identificador podemos bajar los datos.

```
gcel = getGEOSuppFiles("GSE21779")
```

Podemos ver que los coloca en un subdirectorio al que denomina **GSE21779**.<sup>91</sup> Cambiamos al directorio donde tenemos los datos.

```
setwd("GSE21779/")
```

Descomprimimos.

<sup>91</sup> A gusto del consumidor.

Possiblemente para bajar un simple banco de datos lo mejor es utilizar el navegador.

```
system("tar xvf GSE21779_RAW.tar")
```

Cargamos el paquete [75, affy].

```
library(affy)
```

Y leemos todos los datos.

```
gse21779 = ReadAffy()
```

Guardamos estos datos de expresión en un fichero externo.

```
save(gse21779,file = "gse21779.rda")
```

```
save(gse21779,file=paste(dirTamiData,"gse21779.rda",sep=""))
```

Otra opción es bajar los mismos datos de ArrayExpress. Su código en esta base de datos es E-GEOID-21779. Lo hacemos con el siguiente código.

```
library(ArrayExpress)
geod21779 = ArrayExpress("E-GEOID-21779")
```

La ventaja es que directamente nos devuelve un AffyBatch. Normalizamos los datos mediante el método RMA.<sup>92</sup>

<sup>92</sup> § 4.8.

```
geod21779.rma = rma(geod21779)
```

```
save(geod21779.rma,file=paste(dirTamiData,"geod21779.rma.rda",sep=""))
```

## 5.10 GSE1397

<sup>93</sup> El síndrome de Down es una enfermedad causada por la aparición de un copia extra total o parcial del cromosoma 21. Analizaremos si los genes de este cromosoma muestra una sobre expresión. Además se verá que solamente es específica de estos genes y no de otros que aparecen en otros cromosomas. Los datos se pueden obtener de GEO y el identificador del experimento es GSE1397. Los utilizaremos en distintos temas pero de un modo muy importante en ???. Los resultados que se obtienen son realmente bonitos e interesantes. Los datos a nivel de sonda (raw data) no están disponibles. Nos hemos de traer unos datos preprocesados. En el experimento se utilizaron arrays Affymetrix GeneChip U133A. Los obtenemos con

```
pacman::p_load("Biobase","GEOquery")
gse1397raw = getGEO("GSE1397")[[1]]
```

La anotación de estos datos es GPL96 que (podemos comprobarlo entrando en GEO) corresponde con Affymetrix Human Genome U133 chip hgu133a. El paquete que contiene sus datos de anotación es [25, hgu133a.db]. Cambiamos la anotación.

<sup>93</sup> El uso de estos datos y parte del análisis aparecen en un documento que se puede encontrar en [esta dirección](#). También he utilizado parte del análisis que propone [Rodrigo Santamaría](#).

```
annotation(gse1397raw) = "hgu133a"
```

Los datos fenotípicos de las muestras los podemos ver con (no mostramos)

```
pData(gse1397raw)
```

Podemos ver las etiquetas de las covariables asociadas a las muestras con

```
varLabels(gse1397raw)
```

Fijémonos en la variable `type` que nos indica la alteración. En concreto puede ser trisomía 21 (síndrome de Down), trisomía 13 o euploide (ploidía normal) y el tejido puede ser cerebro, cerebelo, astrocito y corazón. Los valores de esta covariable la podemos ver con

```
head(pData(gse1397raw) [, "title"])
```

La matriz de expresión la tenemos (y no la mostramos) con

```
exprs(gse1397raw)
```

Vamos a seleccionar aquellas muestras que corresponden a cerebro o cerebelo y que son o bien TS21 o euploides.

```
(ts21=c(grep("T.*21.*cerebrum", as.character(pData(gse1397raw) [, "title"])),
       grep("TS21.*cerebellum", as.character(pData(gse1397raw) [, "title"]))),
  eu=c(grep("Euploid.*cerebrum", as.character(pData(gse1397raw) [, "title"])),
       grep("Euploid.*[Cc]erebellum", as.character(pData(gse1397raw) [, "title"]))))
```

Efectivamente tenemos las columnas con las muestras que buscábamos

```
pData(gse1397raw) [eu, "title"]
```

```
pData(gse1397raw) [ts21, "title"]
```

Y ahora seleccionamos las muestras.

```
gse1397raw = gse1397raw[, c(eu, ts21)]
```

Modificamos los datos fenotípicos.

```
tissue = factor(rep(c(1,2,1,2), c(4,3,4,3)), levels=1:2,
                 labels=c("Cerebrum", "Cerebellum"))
type = factor(c(rep(1,7), rep(2,7)), levels=1:2, labels=c("Euploid", "TS21"))
GROUP = as.numeric(type) - 1
pData(gse1397raw) = data.frame(tissue, type)
```

Guardamos los datos.

```
save(gse1397raw,file="gse1397raw.rda")
```

<sup>94</sup> Vamos a mostrar los estimadores kernel de la densidad en su escala original. Primero cargamos el paquete [17, affyPLM] que, entre otras cosas, nos permite obtener estos estimadores con facilidad.

```
library(affyPLM)
```

No tenemos los ficheros .CEL que necesitamos para preprocesar los datos. Por ello recurrimos a la función `normalizeBetweenArrays()` del paquete [129, limma].

```
library(limma)
gse1397 = gse1397raw
exprs(gse1397) = normalizeBetweenArrays(exprs(gse1397raw))
```

En la figura ?? tenemos el estimador kernel para las distintas muestras.

```
df = data.frame(gene = featureNames(gse1397),exprs(gse1397))
df1 = melt(df,id=c("gene"))
png(paste(dirTamiFigures,"gse1397normden.png",sep=""))
ggplot(df1,aes(x=value,colour=variable,group=variable)) +
  geom_density(kernel = "epanechnikov",fill=NA) + xlim(0,1000)
dev.off()
```

95

Los autores introdujeron información adicional sobre las filas (genes) que podemos ver (no lo mostramos) con

```
fData(gse1397raw)
```

Es un `data.frame` que contiene las siguientes variables sobre los genes.

```
names(fData(gse1397raw))

## [1] "ID"
## [2] "GB_ACC"
## [3] "SPOT_ID"
## [4] "Species Scientific Name"
## [5] "Annotation Date"
## [6] "Sequence Type"
## [7] "Sequence Source"
## [8] "Target Description"
## [9] "Representative Public ID"
## [10] "Gene Title"
## [11] "Gene Symbol"
## [12] "ENTREZ_GENE_ID"
## [13] "RefSeq Transcript ID"
## [14] "Gene Ontology Biological Process"
## [15] "Gene Ontology Cellular Component"
## [16] "Gene Ontology Molecular Function"
```

<sup>94</sup> El análisis que acabamos de ver desde bajar los datos hasta obtener el ExpressionSet lo podemos encontrar en <http://www.uv.es/ayala/docencia/tami/Rmd/gse1397.html>.

95 [gse1397](#).

## 5.11 Datos GSE20986

<sup>4</sup>

Empezamos obteniendo los datos. Tenemos los datos a nivel de sonda de modo que podemos realizar nuestro propio preprocesado. Una explicación detallada del protocolo experimental se puede ver en **GEO** consultando la información de una cualquiera de las muestras, por ejemplo, en <http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSM524662>. Podemos ver en esta página que el preprocesado de los datos para pasar de los datos a nivel de sonda a datos de expresión se realizó utilizando el método GC-RMA (en concreto utilizaron la función `germa::germa` del paquete [156, `germa`].

<sup>96</sup> El código que le dan en **GEO** a este chip.

La plataforma **GEO**<sup>96</sup> es **GPL570** que es su código para **Affymetrix Human Genome U133 Plus 2.0 Array**. Empezamos cargando paquetes necesarios.

```
pacman::p_load("Biobase", "GEOquery")
```

Nos bajamos los ficheros CEL.

```
gcel = getGEOSuppFiles("GSE20986")
```

Nos ha creado el subdirectorio **GSE20986** por lo que hemos de cambiar el directorio de trabajo al nuevo directorio.

```
setwd("GSE20986/")
```

Además nos ha bajado un fichero tar del cual hemos de extraer los distintos ficheros CEL. En Linux lo haremos con<sup>5</sup>

```
system("tar xvf GSE20986_RAW.tar")
```

Leemos los datos.

```
gse20986raw = affy::ReadAffy()
```

Añadimos variables fenotípicas.

```
tissue = factor(c(1,2,2,1,2,1,3,3,3,4,4,4), levels = 1:4,
  labels=c("iris", "retina", "choroides", "huvec"))
pd = data.frame(tissue)
rownames(pd) = colnames(gse20986raw)
pData(gse20986raw) = pd
```

Guardamos los datos.

```
save(gse20986raw, file="gse20986raw.rda")
```

En la referencia original el preprocesado de los datos se realizó con el método GC-RMA.<sup>97</sup>. Vamos a reproducirlo.

---

<sup>4</sup> El análisis de estos datos lo sugirió un estudio que se puede encontrar en <http://bioinformatics.knowledgeblog.org/2011/06/20/analysing-microarray-data-in-bioconductor/>. En lo que sigue utilizaremos parte del análisis que se propone allí. La referencia original del estudio es [23].

<sup>5</sup> En Windows podemos usar el Winrar.

<sup>97</sup> [131, capítulo 3]

```
pacman::p_load("gcrma", "hgu133plus2probe")
gse20986 = gcrma::gcrma(gse20986raw)
```

Y guardamos el nuevo ExpressionSet.<sup>6</sup>

```
save(gse20986, file="gse20986.rda")
```

En este trabajo se comparan muestras obtenidas a partir de células endoteliales microvasculares de ojos humanos con células endoteliales obtenidas de venas umbilicales humanas (HUVEC). Del primer tipo de células hay tres subtipos dependiendo de dónde se extraen. En concreto se trajeron del iris, la retina y la coroides. Las células HUVEC son más fáciles de obtener y por lo tanto de estudiar. Sin embargo, no es claro que lo que se estudie con ellas sea extrapolable con los resultados que se obtendrían con los otros tipos de células. En resumen, el objeto fundamental es la comparación de este grupo con los otros tres grupos.

Todo el proceso para su obtención a partir de los datos originales <sup>98</sup> que hemos visto en esta sección lo podemos reproducir lo tenemos en el script <http://www.uv.es/ayala/docencia/tami/md/gse20986.html>.

<sup>98</sup> Y todo lo que se hace con estos datos en el manual.

## 5.12 GSE34764

Consideremos los datos de GEO con identificador [GSE34764](#). Necesitamos el paquete [38, GEOquery]. Lo cargamos.

```
library(GEOquery)
```

Obtenemos los ficheros CEL.

```
getGEOSuppFiles("GSE34764")
```

Cambiamos el directorio de trabajo.

```
setwd("GSE30129/")
```

Para leer los datos vamos a utilizar el paquete [31, oligo].

```
library(oligo)
celFiles = list.celfiles()
gse34774 = read.celfiles(celFiles)
save(gse34774, file="gse34774.rda")
```

## 5.13 ArrayExpress

En § 5.5 hemos visto cómo construir un ExpressionSet cuando tenemos los datos de expresión, los fenotípicos y los de anotación. Si consultamos la referencia original [104] vemos que los autores han subido toda la información del experimento a [ArrayExpress](#). Realmente gran

---

<sup>6</sup>Los datos que hemos bajado con la función getGEO ya estaban preprocesados utilizando las mismas funciones que usamos aquí. Lo he hecho para mostrar cómo se hace o bien porque podemos querer utilizar algún otro procedimiento de preprocesado.

parte del trabajo que hicimos en esa sección la podíamos ahorrar bajando los datos ya preparados de esta base de datos. Lo podemos hacer con el paquete [81, ArrayExpress] del siguiente modo.

```
library(ArrayExpress)
rawset = ArrayExpress("E-TABM-157")
```

Podemos comprobar que el objeto **rawset** en donde guarda los datos es un **AffyBatch** a la cual podemos aplicar la corrección de fondo, la normalización y el resumen en § 4.

## 5.14 Varios

Esta sección es una especie de cajón de sastre que contiene cosas que no he sabido colocar elegantemente en el resto del tema.

### 5.14.1 Guardando la matriz de expresión

Supongamos que tenemos un ExpressionSet y queremos guardar la matriz en un fichero externo para leerlo con alguna otra aplicación. Podemos utilizar la función Biobase::write.exprs.

```
data(gse1397, package="tamidat")
```

Guardamos en un fichero texto la matriz de expresión.

```
write.exprs(gse1397, file="gse1397.txt")
```

<sup>99</sup> La versión libre de Excel que tenemos en LibreOffice. Y luego podemos abrir este fichero con Calc<sup>99</sup> o con nuestra aplicación preferida.

## 5.15 Ejercicios

**Ej. 14** — Se pide construir un ExpressionSet. Este ExpressionSet ha de tener la siguiente matriz de expresión.

```
##      [,1]   [,2]   [,3]   [,4]   [,5]
## [1,] 22.01 21.14 24.22 23.06 23.06
## [2,] 24.46 21.59 23.73 22.31 23.73
## [3,] 23.26 24.41 22.07 23.26 22.65
## [4,] 23.09 22.50 23.82 22.00 19.78
## [5,] 20.43 20.78 22.75 22.84 21.51
## [6,] 24.77 23.58 24.00 23.68 23.69
## [7,] 21.99 22.43 22.65 24.45 24.36
## [8,] 23.47 24.02 23.38 23.15 23.83
## [9,] 24.06 24.11 21.36 21.86 23.34
## [10,] 24.44 22.62 23.35 22.20 20.60
## [11,] 24.58 21.44 24.66 23.41 24.38
## [12,] 22.77 25.95 23.20 21.70 24.72
## [13,] 23.66 23.93 24.28 23.25 22.42
## [14,] 20.49 23.83 21.10 21.44 25.75
## [15,] 22.97 20.95 22.54 22.56 19.35
## [16,] 24.07 24.33 24.04 22.23 21.75
## [17,] 22.60 22.62 23.67 24.31 23.30
## [18,] 21.76 22.58 21.27 23.40 23.08
## [19,] 23.52 24.25 21.79 22.62 21.22
## [20,] 23.25 22.39 21.87 22.79 24.96
```

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] "m1"   "m2"   "m3"   "m4"   "m5"   "m6"   "m7"
## [8] "m8"   "m9"   "m10"  "m11"  "m12"  "m13"  "m14"
## [15] "m15"  "m16"  "m17"  "m18"  "m19"  "m20"
```

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
```

**Ej. 15 —**<sup>7</sup> 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 [[38](#), GEOquery] y la función `getGEOSuppFiles`.
- 2.Leer los datos utilizando la función `read.celfiles` del paquete [[31](#), oligo].
- 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.

---

<sup>7</sup>Este problema es muy similar a lo que hacemos en § [5.12](#).

6. Obtener los MAplots.
7. Aplicar el método RMA.
8. Aplicar el método MAS5. ¿Es posible hacerlo?
9. Repetir los dibujos de los apartados 5 y 6 para los datos procesados utilizando el método RMA.

\* **Ej. 16** — 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 6

## Datos de RNA-seq

### 6.1 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. [107] da una buena visión general. En [39, RnaSeqTutorial] tenemos una presentación general de cómo trabajar con estos datos en el contexto de R/Bioconductor. Un texto general que trata lo relativo al análisis de este tipo de datos es [83]. Una referencia más breve pero muy interesante es [36] 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 § 4 y § 5 comentaremos este tipo de información de expresión génica.<sup>100</sup>

<sup>100</sup> En el momento actual uno de los que más interés desperta.

### 6.2 Flujo de trabajo con RNA-seq

Un estudio de este tipo implica la realización de los siguientes pasos ([107]):

1. Diseño experimental.
2. Protocolos de extracción del RNA.
3. Preparación de las librerías. Se convierte el RNA en cDNA y se añaden los adaptadores para la secuenciación.
4. Se secuencian las lecturas cDNA utilizando una plataforma de secuenciación.
5. Alineamiento de las lecturas secuenciadas a un genoma de referencia.
6. Resumen del número de lecturas alineadas a una región.
7. Normalización de las muestras para eliminar diferencias técnicas en la preparación.
8. Estudio estadístico de la expresión diferencial incluyendo en lo posible un modelo.

### 9. Interpretación de los resultados desde el punto de vista biológico.

En este manual estamos interesados fundamentalmente en el análisis de expresión diferencial. Obviamente el punto ?? es básico. Lo realiza el experimentador. En la medida en que vamos a trabajar con datos de experimentos ya realizados hablaremos de cuestiones de diseño aunque no de un modo central. La extracción del RNA, preparación de librerías y la secuenciación son cuestiones no tratadas aquí.

Sí que comentaremos cómo alinear las lecturas cuando disponemos de un genoma de referencia y de los pasos que siguen en el análisis. Sin embargo, este manual se quiere centrar en problemas estadísticos en el contexto de la Bioinformática. Por ello, toda la parte de alineamiento y conteo de las lecturas lo veremos de un modo sencillo insistiendo en cómo hacerlo y no en qué se hace. Nuestro trabajo empieza a partir del momento en que tenemos los conteos y nos interesamos por estudiar si estos conteos son significativamente distintos entre distintas condiciones biológicas.<sup>101</sup>

<sup>101</sup> En [8] tenemos el análisis detallado con todos los pasos desde las lecturas originales hasta el análisis de expresión diferencial final. El interés del trabajo es que podemos reproducir un análisis completo. Además contiene enlaces para aquellas herramientas que se utilizan fuera de R/Bioconductor como `bowtie2`, `samtools` y `tophat2`. En <http://www.bioconductor.org/help/workflows/rnaseqGene/> tenemos todo un flujo en donde se analiza un experimento. El paquete [89, airway] tiene los datos para poder trabajar.

En el diseño hemos de diferenciar entre **réplicas técnicas** en la que utilizamos una misma muestra biológica, un mismo procesado y los mismos protocolos de secuenciación. Es de esperar que las diferencias de lo que observamos en estas réplicas técnicas sean menores a las obtenidas con **réplicas biológicas** en las que se utilizan distintas muestras biológicas. En los experimentos RNA-seq solemos tener réplicas biológicas y raramente réplicas técnicas.

¿Cuántas muestras? Se trabaja con muestras extremadamente pequeñas: 2 o 3 por tratamiento. Obviamente **no** podemos esperar ver diferencias claras entre tratamientos.

Un diseño experimental siempre ha de ser simple. Este es un comentario aplicable a cualquier tipo de dato pero es particularmente importante con dato ómico. Sin duda, en general y en particular en este contexto, los diseños experimentales han de mantenerse lo más simples que se pueda.

¿Qué tipos de tratamiento hemos de realizar? Es necesaria una normalización previa de las muestras que permita “controlar” las variaciones técnicas.

1. Por ejemplo, las muestras proporcionadas por el secuenciador tienen distintas cantidades de DNA y esto da lugar a diferencias en el número total de lecturas secuenciadas y alineadas independientemente de la diferente expresión diferencial de un gen. El tratamiento debiera afectar solamente a una fracción de los genes evaluados. Si la diferencia se observa en “todos” los genes lo que vemos es un efecto del protocolo de trabajo.
2. También hay diferencias entre genes que no se deben a diferencias de expresión. En los protocolos estandar un gen largo es secuenciado con más frecuencia que un gen corto.

En las secciones que siguen se comentan los datos que utilizamos en los restantes capítulos.

La clase que en R/Bioconductor debe de utilizarse para trabajar con datos RNA-seq es `GenomicRanges::SummarizedExperiment`.

Responder las siguientes preguntas es el objeto de este tema.

1. ¿Dónde podemos obtener datos de secuenciación? De otro modo: ¿qué repositorios podemos utilizar?

2. ¿Cómo podemos realizar búsquedas en los metadatos con objeto de obtener experimentos que tenga que ver con lo que me interesa?
3. ¿Cómo bajarlos?
4. ¿Cómo contar las lecturas alineadas sobre intervalos o uniones de intervalos sobre el genoma?

## 6.3 Repositorios

Los repositorios donde podemos encontrar este tipo de datos son NCBI en los Estados Unidos, EMBL en Europa y DDBJ en Japón.<sup>102</sup> En NCBI GEO también hay datos de secuencias y podemos acceder a ellos utilizando [38, GEOquery]. En concreto las direcciones son

**NCBI SRA** <http://www.ncbi.nlm.nih.gov/sra> Posiblemente es la mayor de las bases de datos. No almacena lecturas alineadas. En NCBI **GEO** sí que tenemos algunos bancos de datos con lecturas alineadas sobre un genoma de referencia.

**EBI ENA** <http://www.ebi.ac.uk/ena>

**DDBJ** <http://www.ddbj.nig.ac.jp>

En lo que sigue utilizamos fundamentalmente la primera base de datos y, a veces, la segunda. Cada una de estas bases de datos puede ser consultada en línea y bajar los datos desde la propia página. También usaremos en lo que sigue herramientas para hacerlo desde R, por ejemplo, [**R-SRAdb**].

En las secciones § 6.4 y § 6.5. comentamos distintos formatos para almacenar lecturas cortas. También comentamos paquetes R/Bioconductor para trabajar con estos formatos.

<sup>102</sup> Son los básicos como para datos de microarrays tenemos NCBI GEO y EBI ArrayExpress.

## 6.4 Formato FASTA

Es el formato basado en texto para representar secuencias bien de nucleótidos bien de péptidos. 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 tiene en la primera posición 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.

## 6.5 Formato FASTQ

Es uno de los formatos más populares para 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 línea 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. Las cuatro siguientes líneas corresponden a un ejemplo.

<sup>103</sup> El procedimiento que utiliza aparece en [46] y cómo estima las probabilidades de error cuando asigna cada base aparece en [45].

<sup>104</sup> No es realmente una probabilidad. Es una cuantificación de la calidad de la asignación y no más.

<sup>105</sup> Sin duda, un método curioso de mostrar valores numéricos utilizando texto.

¿Cómo se cuantifica la confianza o precisión? Se utiliza el programa Phred.<sup>103</sup> Este programa lo que hace es asignar los picos de fluorescencia a una de las cuatro bases (o *base call*). En [45] tenemos la explicación del método de asignación. Si  $P$  denota la probabilidad<sup>104</sup> para una base dada de ser mal asignada o clasificada entonces el valor con el que se trabaja es

$$Q = -10 \log_{10} P. \quad (6.1)$$

Esto significa que una probabilidad  $P$  muy pequeña de clasificación incorrecta se traduce en un valor grande de  $Q$ . Una vez calculado  $Q$  entonces se asigna (usualmente) el carácter ASCII que ocupa la posición  $33 + Q$ .<sup>105</sup> Una descripción detallada sobre FASTQ la tenemos en [35].

## Utilizando ShortRead

Para trabajar con el formato **FASTQ** es útil el paquete [99, ShortRead].

```
library(ShortRead)
```

<sup>106</sup> En ?? vemos cómo podemos bajar estos datos.

Vamos a ver sus funcionalidades utilizando el `SRR1293399_1.fastq`.<sup>106</sup> En esta sección suponemos fijado el directorio de trabajo a donde tenemos los datos.

<sup>107</sup> Veremos que tarda lo suyo en leer los datos.

Leemos con `ShortRead::readFastq`.<sup>107</sup>

Es un fichero con muchas lecturas y pa-

azar. Y lo hacemos con `ShortRead::FastqSampler`. Volvemos a la muestra aleatoria.

Abono fujo es un ejemplo de clase

**class**(fg0)

```
## [1] "ShortReadQ"  
## attr(,"package")  
## [1] "ShortRead"
```

En particular podemos ver la información básica con

fg0

```
## class: ShortReadQ  
## length: 10000 reads; width: 50 cycles
```

<sup>108</sup> Subsetting

Obviamente tiene solamente 10000 lecturas. Podemos seleccionar<sup>108</sup> del modo habitual. Por ejemplo, la primera sería

```
fq0[1]
## class: ShortReadQ
## length: 1 reads; width: 50 cycles
```

O de la tercera a la décima con

```
fq0[3:10]
## class: ShortReadQ
## length: 8 reads; width: 50 cycles
```

Con `ShortRead::sread` podemos tener la lectura. Por ejemplo, la que ocupa la posición 1000 sería

```
sread(fq0[1000])
## A DNAStringSet instance of length 1
##   width seq
## [1] 50 AGAGAAATCGGTGTCAGTC...GTATGCCGTCTTGCTT
```

Podemos ver las primeras y últimas con

```
sread(fq0)
## A DNAStringSet instance of length 10000
##   width seq
## [1] 50 TAAACACTGTCTCCCTG...CCTGGTGTCACTCACT
## [2] 50 ACAGCTTGCTGAGAC...TGTCAGTCATTCCAGC
## [3] 50 GTGTCAGTCACTTCCAG...GGTCGTATGCCGTCTT
## [4] 50 TGCAGCCCCGGGTTCC...CTGAGCGTGTCACTCA
## [5] 50 TCTTGGGTTCCGGGG...TGAAACGTGTCACTCA
## ...
## [9996] 50 ATAGTCTGTGGGAGTCA...CACTTCCAGCGGTCTG
## [9997] 50 CAAAGTGCCTACAGTGC...CCAGCGGTCTATGCC
## [9998] 50 GTGTCAGTCACTTCCAG...CTGCTTGAAAAAAA
## [9999] 50 TGGCGCTGCAGGATGAA...GCGCCCGATGCCGACG
## [10000] 50 CAACTAGCCCTGAAAAT...CAGGCCCATACCCGTG
```

Podemos ver la longitud de las lecturas con

```
width(fq0)
```

Con `ShortRead::detail` tenemos una información detallada. Las medidas de calidad de la lectura 1000 las obtenemos con

```
quality(fq0[1000])
## class: SFastqQuality
## quality:
##   A BStringSet instance of length 1
##   width seq
## [1] 50 ffffffffffffefff...fffeffff`ddbdd
```

Podemos saber el valor a que corresponde la codificación de la calidad con

```
encoding(quality(fq0))

## ; < = > ? @ A B C D E F G H I J
## -5 -4 -3 -2 -1 0 1 2 3 4 5 6 7 8 9 10
## K L M N O P Q R S T U V W X Y Z
## 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26
## [ \ ] ^ _ ` a b c d e f g h i
## 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41
```

```
f1s = dir(paste(dirTamiData, "SRP042140/fastq", sep=""), "*fastq$", full=TRUE)
```

**Evaluando la calidad del experimento.** Vamos a realizar un control de calidad del experimento con ShortRead::qa. Lo hacemos para la primera muestra.<sup>109</sup>

<sup>109</sup> Obviamente debemos repetir el análisis para cada una de las muestras.

```
SRP042140_1.qa = qa(f1s[1], type="fastq")
```

<sup>110</sup> A los usuarios de Windows es probable que el código que sigue no les funcione. ¿Les extraña?

Generamos un informe y lo visualizamos.<sup>110</sup>

```
browseURL(report(SRP042140_1.qa))
```

Es un informe muy interesante. Notemos que estos datos ya están previamente filtrados<sup>111</sup> y por ello algunas de las características evaluadas no tienen sentido.

Podemos ver la información contenida en SRP042140.qa con

```
show(SRP042140_1.qa)

## class: FastqQA(10)
## QA elements (access with qa[["elt"]]):
##   readCounts: data.frame(1 3)
##   baseCalls: data.frame(1 5)
##   readQualityScore: data.frame(512 4)
##   baseQuality: data.frame(94 3)
##   alignQuality: data.frame(1 3)
##   frequentSequences: data.frame(50 4)
##   sequenceDistribution: data.frame(357 4)
##   perCycle: list(2)
##     baseCall: data.frame(249 4)
##     quality: data.frame(1800 5)
##   perTile: list(2)
##     readCounts: data.frame(0 4)
##     medianReadQualityScore: data.frame(0 4)
##   adapterContamination: data.frame(1 1)
```

Por ejemplo podemos ver el número de lecturas.

```
SRP042140_1.qa[["readCounts"]]

##                      read filter aligned
## SRR1293399_1.fastq 29508968      NA      NA
```

O también podemos ver las frecuencias de cada una de las bases.

```
SRP042140_1.qa[["baseCalls"]]

##                      A          C          G
## SRR1293399_1.fastq 10600637 12762822 12873094
##                      T          N
## SRR1293399_1.fastq 13753938 9509
```

O las lecturas más frecuentes.

```
head(SRP042140_1.qa[["frequentSequences"]])

##                                     sequence
## 1 GTGTCAGTCACCTCCAGCGGTCGTATGCCGTCTTGCTTGGAAAAAAA
## 2 TAGCTTATCAGACTGATGTTGACGTGTCAGTCACCTCCAGCGGTCGTATG
## 3 TGTAAACATCCCCGACTGGAAGCGTGTCAAGTCACCTCCAGCGGTCGTATG
## 4 AACTGCCCTCAAAGTCCCGCTGTGTCAAGTCACCTCCAGCGGTCGTATGC
## 5 TGTCAGTCACCTCCAGCGGTCGTATGCCGTCTTGCTTGGAAAAAAA
## 6 TGTAAACATCCCCGACTGGAAGCTGTGTCAAGTCACCTCCAGCGGTCGTAT
##   count type           lane
## 1 94635 read SRR1293399_1.fastq
## 2 55085 read SRR1293399_1.fastq
## 3 37265 read SRR1293399_1.fastq
## 4 28687 read SRR1293399_1.fastq
## 5 18234 read SRR1293399_1.fastq
## 6 13152 read SRR1293399_1.fastq
```

Podemos ver la frecuencia de cada base en cada posición de la lectura. Por ejemplo, para las dos primeras posiciones.

```
head(SRP042140_1.qa[["perCycle"]])$baseCall,n=10

##   Cycle Base Count           lane
## 1     1   A 221622 SRR1293399_1.fastq
## 2     1   C  70578 SRR1293399_1.fastq
## 3     1   G 202760 SRR1293399_1.fastq
## 4     1   T  504717 SRR1293399_1.fastq
## 15    1   N   323 SRR1293399_1.fastq
## 19    2   A 350254 SRR1293399_1.fastq
## 20    2   C  97623 SRR1293399_1.fastq
## 21    2   G 280410 SRR1293399_1.fastq
## 22    2   T 271675 SRR1293399_1.fastq
## 33    2   N    38 SRR1293399_1.fastq
```

```
head(SRP042140_1.qa[["perCycle"]])$quality,n=10

##   Cycle Quality Score Count           lane
## 35     1        B     2   678 SRR1293399_1.fastq
```

```
## 44      1      K      11     80 SRR1293399_1.fastq
## 45      1      L      12    3411 SRR1293399_1.fastq
## 46      1      M      13     625 SRR1293399_1.fastq
## 51      1      R      18     137 SRR1293399_1.fastq
## 53      1      T      20     6936 SRR1293399_1.fastq
## 54      1      U      21     907 SRR1293399_1.fastq
## 55      1      V      22    1140 SRR1293399_1.fastq
## 56      1      W      23      88 SRR1293399_1.fastq
## 57      1      X      24    757 SRR1293399_1.fastq
```

## 6.6 De SRA a fastq

<sup>112</sup> En Debian/Ubuntu hemos de instalar el paquete sra-toolkit. Consultar

<https://github.com/ncbi/sra-tools/wiki/Downloads>.

<sup>113</sup> Previamente hay que instalar *parallel* con *apt-get install parallel*.

Lo hacemos con SRA Toolkit.<sup>112</sup> Utilizamos la función *fastq-dump*.

```
fastq-dump -I --split-files nombre_fichero.sra
```

Y lo repetimos para cada fichero SRA.

Una opción que nos lo hace para todos es<sup>113</sup>

```
cat files | parallel -j 7 fastq-dump --split-files {} .sra
```

donde *files* es un fichero que tiene en cada línea el nombre del fichero sra sin la extensión. En nuestro caso el fichero *files* es

```
SRR1293399
SRR1293400
SRR1293401
SRR1293402
SRR1293403
SRR1293404
SRR1293405
SRR1293406
SRR1293407
```

## 6.7 Control de calidad de un experimento RNA-seq con EDASeq

El control de calidad se puede realizar en dos pasos. En el primero se trata de evaluar si tenemos librerías (muestras) con bajas profundidades, o con problemas de calidad o con frecuencias de nucleóticos no frecuentes. En un segundo paso, a nivel de gen, buscamos etiquetas equivocadas asociadas a las muestras, problemas con las distribuciones de probabilidad supuesta (sobredispersión es el gran problema) y sesgos en los contenidos GC.

**EDA al nivel de lectura.** En particular, tenemos que considerar los siguientes aspectos:

**Número de lecturas alineadas y no alineadas.** Si hay un total de pocas lecturas esto se atribuye a la mala calidad del RNA. Si hay una tasa de alineamiento bajo es que las lecturas tienen poca calidad, normalmente tienen secuencias poco complejas de difícil alineamiento. También podemos tener un mal genoma de referencia o simplemente mal etiquetadas las muestras.

## 6.8 STAR y samtools

Veamos cómo alinear y contar utilizando conjuntamente STAR y samtools.

1. Instalamos **STAR**.
2. Copiamos la versión Linux en el directorio donde tenemos los ficheros **SRA**.
3. Creamos dos subdirectorios con los nombres **aligned** y **fastq**.
4. Nos bajamos (con `wget`) de [http://labshare.cshl.edu/shares/gingeraslab/www-data/dobin/STAR/STARgenomes/ENSEMBL/homo\\_sapiens](http://labshare.cshl.edu/shares/gingeraslab/www-data/dobin/STAR/STARgenomes/ENSEMBL/homo_sapiens) la última versión.<sup>1</sup>
5. Descomprimimos el fichero anterior y extraemos los ficheros del archivo tar en el subdirectorio `ENSEMBL.homo\_sapiens.release-75`.
6. Supongamos que:
  - (a) El ejecutable **STAR** lo tenemos en nuestro directorio de trabajo.
  - (b) `ENSEMBL.homo\_sapiens.release-75` es un subdirectorio de nuestro directorio de trabajo.
  - (c) Los ficheros **FASTQ** los tenemos en el subdirectorio **fastq**.
  - (d) Pretendemos guardar el fichero **SAM** en el subdirectorio **aligned**.

Entonces el código (utilizando STAR) para generar el fichero **SAM** a partir del **FASTQ** sería el siguiente

```
./STAR --genomeDir ENSEMBL.homo_sapiens.release-75
      ↪ --readFilesIn
fastq/SRR1293399_1.fastq --runThreadN 12 --
      ↪ outFileNamePrefix aligned/SRR1293399_1.
```

Podemos comprobar que el fichero generado es `SRR1293399_1.Aligned.out.sam`. Y el código anterior lo hemos de repetir para cada uno de nuestros ficheros **FASTQ**.

### De **SAM** a **BAM** utilizando **Samtools**.

Utilizamos **Samtools** para transformar los ficheros **SAM** en ficheros **BAM**.

```
samtools sort view -bS aligned/SRR1293399_1.Aligned.out.
      ↪ sam -o aligned/SRR1293399_1.bam
```

### Contando las lecturas con **Rsamtools**.

1. Suponemos que los ficheros **bam** generados previamente están en el subdirectorio **aligned** de nuestro directorio de trabajo.
2. Creamos el fichero **bamfiles.txt** que tenga en cada línea el nombre de uno de los ficheros **BAM**. Por ejemplo, las tres primeras líneas de este fichero podrían ser

<sup>1</sup> En el momento de escribir `ENSEMBL.homo_sapiens.GRCh38.release-79`.

```
SRR1293399_1.bam
SRR1293400_1.bam
SRR1293401_1.bam
```

3. Creamos la lista de ficheros bam.

```
library(Rsamtools)
dirActualData = paste(dirTamiData, "SRP042140_sra/", sep="")
sampleTable = read.table(paste(dirActualData, "bamfiles.txt", sep=""))
fls = paste(dirActualData, "aligned/", sampleTable[,1], sep="")
bamLst = BamFileList(fls, index=character(), yieldSize=100000, obeyQname=TRUE)
```

4. Contamos las lecturas.

- (a) Cargamos [30, GenomicFeatures].

```
library(GenomicFeatures)
```

- (b) Leemos el fichero GTF/GFF que hemos bajado previamente.

```
gtffile = paste(dirTamiData,
"ENSEMBL.homo_sapiens.release-75/Homo_sapiens.GRCh37.75.gtf", sep="")
```

- (c) txdb = makeTxDbFromGFF(gtffile, format="gtf")
genes = exonsBy(txdb, by="gene")
library(GenomicAlignments)
SRP042140.counts = summarizeOverlaps(features = genes, read=bamLst,
mode="Union",
singleEnd=TRUE, ## No son lecturas apareadas
ignore.strand=TRUE,
fragments=FALSE)
save(SRP042140.counts,
file=paste(dirTamiData, "SRP042140\_sra/", "SRP042140.counts.rda"), sep="")

## 6.9 Bowtie2

<sup>114</sup> <http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml>

<sup>115</sup> gapped

<sup>116</sup> Paired-end

114

Es una herramienta software para alinear lecturas cortas sobre secuencias de referencia largas. Soporta distintos modos de alineamiento: con huecos<sup>115</sup>, locales y con lecturas apareadas<sup>116</sup>. La salida del programa la realiza en formato SAM.

El programa bowtie2 utiliza un fichero de índices de clase Bowtie 2 y ficheros con lecturas secuenciadas y produce un fichero en formato SAM.

¿Qué es alinear? Tomamos una lectura o secuencia corta y pretendemos encontrar en una secuencia larga donde es más similar, en qué punto hay una mayor similitud respecto de la secuencia larga o de referencia. ¿Cuál es el resultado que obtenemos después de alinear? La secuencia corta es colocada sobre una parte de la secuencia de referencia indicando en qué puntos se produce una correspondencia e indicando

<sup>117</sup> Gaps.

los huecos<sup>117</sup> que se han tenido que introducir en una u otra de las secuencias para conseguir la mejor correspondencia posible entre dichas secuencias.

```
Read:      GACTGGCGATCTGACTTCG
           |||||  |||||||||  ||
Reference: GACTG--CGATCTCGACATCG
```

Mediante la línea vertical mostramos en qué puntos hay una correspondencia correcta mientras que con un guión en una u otra secuencia indicamos en qué posiciones hemos debido de incorporar un hueco para poder hacer corresponder ambas secuencias.

Notemos que no estamos seguros de que efectivamente la secuencia corta se haya originado en este punto. Es una hipótesis. La mejor que se puede conjutar pero no es seguro ni mucho menos. En muchas ocasiones no se puede encontrar una correspondencia.

Dos tipos de alineamientos son considerados: **alineamiento de extremo a extremo** o alineamiento global<sup>118</sup> y alineamiento local.<sup>119</sup> En la primera opción toda la secuencia corta es alineada. En el segundo tipo permitimos que uno o los dos extremos contengan bases no alineadas. En **Bowtie2** la opción local se indica `--local` para el alineamiento local mientras que la opción global se asume por defecto.

Hemos de evaluar la calidad del alineamiento. Para ello definimos una puntuación asociada a cada posible correspondencia. Lo que podemos llamar una función objetivo. La idea es que cuanto mayor sea la función objetivo mejor es el alineamiento. Realmente penalizamos cada error en la correspondencia. Tendremos una penalización por correspondencia errónea, una penalización por cada hueco. En el caso de alineamiento local sumaremos un valor por cada correspondencia. Finalmente el valor de la función objetivo nos cuantifica la calidad final de la correspondencia conseguida. En concreto se consideran los siguientes valores que se suman (bajando o subiendo) para calcular el valor de la función objetivo.

- ma** Sumamos un valor positivo por cada correspondencia (para alineamiento local).
- mp** Penalización (restamos) por una correspondencia errónea.
- np** Penalización por tener un valor N (esto es una base no especificada) en la secuencia corta o en la larga de referencia.
- rdg** Penalización por hueco en secuencia corta.
- rfg** Penalización por hueco en secuencia de referencia.

Obviamente mediante algún procedimiento encontraremos siempre una correspondencia. No necesariamente esta correspondencia la hemos de considerar suficientemente buena. ¿Cuándo la consideraremos aceptable? Lo que se hace es fija un umbral por debajo del cual el alineamiento no se acepta. Este umbral lógicamente ha de depender de la longitud de la lectura corta. El valor de este umbral que por defecto (aunque es configurable) tiene definido **Bowtie2** es, para alineamiento global,

$$-0.6 - 0.6 \cdot L, \quad (6.2)$$

siendo  $L$  la longitud de la lectura. Para alineamiento local el valor es

$$20 + 8 \cdot \ln(L). \quad (6.3)$$

<sup>118</sup> End-to-end alignment.

<sup>119</sup> Local alignment.

Lo podemos configurar con la opción `--score-min`.

¿Podemos alinear de un modo único? Por supuesto que no. Pensamos en zonas de la secuencia en donde se repiten mucho los elementos. Si tenemos secuencias cortas entonces podremos alinear (perfectamente) de distintos modos. Tendremos distintos alineamientos igualmente buenos. Se plantea el problema de elegir entre estos alineamientos. Esto se puede hacer utilizando las medidas de calidad de la correspondencia.

<sup>120</sup> Paired-end o mate-pair.  
No es lo mismo.

¿Qué son lecturas apareadas?<sup>120</sup> Son pares de lecturas cortas o secuencias cortas de las cuales conocemos a priori su posición relativa y la distancia que las separa en la lectura de DNA. Cuando tenemos lecturas apareadas tendremos dos ficheros que incluyen el primer y el segundo elemento del par, uno en cada fichero. En concreto la misma línea en cada fichero indica cada una de las componentes del par de lecturas. En **Bowtie2** se utilizan los argumentos `-1` y `-2` para indicar el primer y segundo fichero. Cuando se realiza el alineamiento de un par de secuencias cortas en el fichero **SAM** resultante se indica en los campos **RNEXT** y **PNEXT** el nombre y la posición del otro elemento del par. También se indica la longitud del fragmento de DNA del cual se secuenciaron los dos fragmentos. Cuando se realiza el alineamiento de un par de lecturas este se puede producir de un modo concordante o de un modo discordante. En un modo concordante los dos elementos del par se alinean verificando lo que se espera en orientación y distancia entre ellos. El alineamiento es discordante entonces tenemos para cada uno **un alineamiento único** pero no se verifican las restricciones de orientación y distancia. Por defecto, **Bowtie2** busca tanto alineamientos concordantes como alineamientos discordantes. Con la opción `--no-discordant` solamente busca concordantes.

En la opción por defecto, **Bowtie2** realiza un alineamiento mixto. En esta opción primero busca alinear el par de lecturas de un modo concordante. Si no lo logra entonces busca alinear cada una de ellas de un modo discordante. Si no queremos la segunda parte de la búsqueda podemos eliminarla con la opción `--no-mixed`.

**Bowtie2** busca alineamientos válidos para cada lectura. Cuando encuentra un alineamiento válido sigue buscando otros que sean al menos tan buenos como el que ha encontrado. Tiene unos controles para dejar de buscar. Cuando cesa la búsqueda utiliza los mejores alineamientos que ha encontrado para evaluar la calidad de la correspondencia encontrada (el campo **MAPQ** en el formato **SAM**).

En <http://bowtie-bio.sourceforge.net/bowtie2/manual.shtml> se tiene una muy buena exposición de todas las opciones que ofrece este programa y es más que recomendable su lectura.

#### Ejemplo 6.1 Vamos a realizar el alineamiento utilizando **Bowtie2**.

1. Los datos corresponden a Illumina.
2. Bajamos el fichero de índices de [http://support.illumina.com/sequencing/sequencing\\_software/igenome.html](http://support.illumina.com/sequencing/sequencing_software/igenome.html).
3. En concreto bajamos `\Homo_sapiens_Eensembl_GRCh37.tar.gz`.
4. Ejecutamos en línea de comandos.

```
bowtie2 -x ~/DOCENCIA/tami-data/Homo_sapiens/Ensembl
        ↪ /GRCh37/Sequence/Bowtie2Index/genome -U fastq
        ↪ /SRR1293399_1.fastq -S aligned/SRR1293399_1.
        ↪ sam
```

## 6.10 Utilizando tophat y samtools

Seguimos los siguientes pasos.

1. Necesitamos el genoma humano de referencia. Vamos a utilizar GRCh37.
2. Lo podemos encontrar en <http://ccb.jhu.edu/software/tophat/igenomes.shtml>. Lo bajamos.
3. Descomprimimos el fichero anterior.

```
gzip -d Homo_sapiens_Eensembl_GRCh37.tar.gz
```

4. Alineamos con

```
tophat2 -o file_tophat_out genome SRR479053_1.fastq
    ↪ SRR479053_2.fastq
samtools sort -n file_tophat_out/accepted_hits.
    ↪ bam_sorted
```

Antes de realizar el alineamiento necesitamos el genoma humano de referencia. Se utiliza GRCh37. En concreto vamos a utilizar el índice Bowtie que podemos encontrar en <http://ccb.jhu.edu/software/tophat/igenomes.shtml>.

Descomprimimos el fichero anterior.

```
gzip -d Homo_sapiens_Eensembl_GRCh37.tar.gz
```

Alineamos.

```
tophat2 -o file_tophat_out genome SRR479053_1.fastq
    ↪ SRR479053_2.fastq
samtools sort -n file_tophat_out/accepted_hits.bam_sorted
```

## 6.11 GSE37211: paquete parathyroidSE

Estos datos corresponden al experimento con número de acceso GEO GSE37211. En la viñeta del paquete [90, parathyroidSE] tenemos una descripción detallada para obtener los conteos a nivel de gen o de exon partiendo de los ficheros originales sra.<sup>121</sup> Ya los tenemos en el paquete [90, parathyroidSE].

<sup>121</sup> Primero hay que bajarlos pacientemente.

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

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

```
class(se)

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

Es un SummarizedExperiment::RangedSummarizedExperiment y será nuestra clase de referencia cuando trabajamos con datos de RNA-seq.<sup>122</sup> ¿Cuántos genes y muestras tenemos?

```
dim(se)
```

```
## [1] 63193     27
```

<sup>123</sup> Es la análoga a Bio-base::exprs para Biobase::ExpressionSet.

La matriz con los conteos nos la da GenomicRanges::assay.<sup>123</sup>

```
assay(se)
```

Veamos las tres primeras filas (genes) y las tres primeras columnas (muestras).

```
assay(se)[1:3,1:3]
```

```
##          [,1] [,2] [,3]
## ENSG000000000003 792 1064 444
## ENSG000000000005    4    1    2
## ENSG00000000419 294 282 164
```

<sup>124</sup> Corresponde con Biobase::pData.

Tenemos los metadatos de las columnas o variables fenotípicas o covariables asociadas a las muestras con GenomicRanges::colData.<sup>124</sup>

```
colData(se)
```

Es un IRanges::DataFrame.

```
class(colData(se))
```

```
## [1] "DataFrame"
## attr(,"package")
## [1] "IRanges"
```

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

```
names(colData(se))
```

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

Y acceder a los valores de la variable treatment con

```
colData(se)[, "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
```

<sup>122</sup> Es muy conveniente leer la viñeta de [101, SummarizedExperiment].

o bien con (no mostrado)

```
colData(se)$treatment
```

La información sobre las filas que, en este caso, corresponde con genes.

```
rowRanges(se)
```

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

```
rownames(se)
```

Los primeros con

```
head(rownames(se))
```

```
## [1] "ENSG00000000003" "ENSG00000000005"
## [3] "ENSG00000000419" "ENSG00000000457"
## [5] "ENSG00000000460" "ENSG00000000938"
```

125

<sup>125</sup> En esta sección hemos visto cómo manejar un la clase GenomicRanges::SummarizedExperiment y sus métodos asociados.

## 6.12 GSE64099

Bajamos los datos de [GEO](#).

```
wget ftp.ncbi.nlm.nih.gov/geo/series/GSE64nnn/GSE64099/
    ↳ suppl/GSE64099_RAW.tar
tar xvf GSE64099_RAW.tar
gzip -d *
rm GSE64099_RAW.tar
```

Se tienen las siguientes muestras:

```
GSM1564328_2086.txt  GSM1564331_241.txt  GSM1564329_2433.txt
GSM1564332_1932.txt  GSM1564327_1934.txt  GSM1564330_3225.txt
GSM1564333_1940.txt
```

Construimos un ExpressionSet.

```
library(Bioconductor)
exprs0 = CountAll
rownames(exprs0) = EntrezAll
colnames(exprs0) = c("GSM1564328", "GSM1564331", "GSM1564329", "GSM1564332",
                     "GSM1564327", "GSM1564330", "GSM1564333")
type = c(2,1,2,1,1,2,2)
type = factor(type, levels=1:2, labels=c("Wild-type", "Smchd1"))
type = data.frame(type)
rownames(type) = colnames(exprs0)
datosfenotipo = new("AnnotatedDataFrame", data = type)
sampleNames(datosfenotipo)
```

```
datosexperimento = new('MIAME',name='GSE64099',
  lab='Molecular Medicine Division, The Walter and Eliza Hall Institute
of Medical Research',
  contact ='mritchie@wehi.edu.au',
  title = ' Transcriptome profiling for genes transcriptionally
regulated by Smchd1 in lymphoma cell lines',
  url = 'http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE64099')
gse64099 = new("ExpressionSet",exprs=exprs0,phenoData = datosfenotipo,
experimentData = datosexperimento,annotation = "MusMusculus")
```

Eliminamos los genes que no tienen ninguna lectura alineada. Todos estos genes no tienen interés cuando estudiemos la posible expresión diferencial.

```
nullsum = apply(exprs(gse64099),1,sum)==0
gse64099 = gse64099[!nullsum,]
```

## 6.13 GSE63776 a partir de conteos

En ocasiones tampoco es necesario realizar todo el procesado que hemos visto. Esto es, no necesitamos partir de las librerías de lecturas cortas y realizar todo el proceso de alineamiento. Los propios autores del estudio original han puesto a nuestra disposición los conteos. En este caso lo más cómodo es utilizar un `ExpressionSet` para almacenar la información. Como ejemplo los datos GSE63776.

1. Los bajamos de <http://www.ncbi.nlm.nih.gov/geo/download/?acc=GSE63776&format=file&file=GSE63776%5FPANC1%5Fcounts%2Etxt%2Egz>.<sup>126</sup> Vamos a trabajar con los conteos.
2. Descomprimimos el fichero `GSE37211_count_table.txt.gz`.
3. Abrimos el fichero con un editor<sup>127</sup> y añadimos al principio de la primera línea la palabra `Gene`. Guardamos el fichero con la modificación.
4. Leemos los datos.

```
x = read.table(file="GSE63776_PANC1_counts.txt",header=TRUE)
```

5. la primera columna es el nombre del gen por lo que la eliminamos para quedarnos con la matriz de expresión (conteos).

```
exprs0 = as.matrix(x[,-1])
rownames(exprs0) = x[, "Gene"]
```

6. Construimos los metadatos o datos fenotípicos.

```
type = factor(rep(1:2,each=3),levels=1:2,labels=c("siControl","siTCF7L2"))
pd0 = data.frame(type)
rownames(pd0) = colnames(exprs0)
datosfenotipo = new("AnnotatedDataFrame", data = pd0)
```

<sup>126</sup> Casi mejor poner el código de acceso GSE63776 en Google y tenemos la dirección.

<sup>127</sup> Bloc de nota, gedit, emacs

```

datosexperimento = new('MIAME',name='GSE63776',
                      lab='Seth Frietze and Farnham P',
                      contact ='seth.frietze@unco.edu',
                      title = ' ',abstract = 'We have compared the genome-wide effects on the transcriptome after treatment with ICG-001 (the specific CBP inhibitor) versus C646, a compound that competes with acetyl-coA for the Lys-coA binding pocket of both CBP and p300. We found that both drugs cause large-scale changes in the transcriptome of HCT116 colon cancer cells and Panc1 pancreatic cancer cells, and reverse some tumor-specific changes in gene expression. Interestingly, although the epigenetic inhibitors affect cell cycle pathways in both the colon and pancreatic cancer cell lines, the WNT signaling pathway was affected only in the colon cancer cells. Notably, WNT target genes were similarly down-regulated after treatment of HCT116 with C646 as with ICG-001.',
                      url = 'http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE63776',
                      other = list(notes = 'Public on Jan 07, 2015'))
gse63776 = new("ExpressionSet",exprs=exprs0,phenoData = datosfenotipo,
               experimentData = datosexperimento,annotation =
               "Illumina HiSeq 2000 (Homo sapiens)")
save(gse63776,file="gse63776.rda")

```

## 6.14 Normalización de RNA-seq

En esta sección se proponen distintos métodos de normalización propuestos en la literatura. Es cierto que los datos de RNA-seq tienen menos ruido que los datos de microarrays. En un principio se consideró que no se necesitaban procedimientos de normalización sofisticados [145]. No obstante, en el protocolo de preparación intervienen muchos procedimientos que introducen variabilidad: la extracción del RNA, la transcripción reversa, la amplificación y la fragmentación.<sup>128</sup> Empezamos con muy normalizado, con algo muy normal, con algo demasiado normal que podemos ver en la figura ??.

¿Cuáles pueden ser las covariables relativas a los genes y a las muestras que pueden influir en los conteos y podemos considerar ruido técnico?<sup>129</sup>

**Profundidad de secuenciación o tamaño de la librería**<sup>130</sup> Estamos muestreando del total de moléculas disponibles. El tamaño de nuestra muestra es el total de lecturas. Este total de lecturas es distinto para las distintas muestras. En algunos procedimientos para análisis de la expresión diferencial este efecto no necesitará corrección previa pues el propio método incorpora en su modelo estoacástico la distinta profundidad.

**Composición del RNA** Mediante la técnica RNA-seq tenemos una medida de la abundancia relativa de cada gen en una muestra de RNA. Pero no tenemos una medida de la cantidad total de RNA por célula. Supongamos que tenemos una pequeña cantidad de genes que se expresan mucho en una muestra pero no en otra. En la muestra en donde se expresan mucho consumen una parte fundamental de la librería de lecturas de modo que otros genes que se expresen en esa muestra pero en mucho menor medida apenas aparecerán. De

<sup>128</sup> En resumen, que no nos libraremos de normalizar.



<sup>129</sup> El término ruido técnico es peligroso. ¿Cuándo es puramente técnico lo que eliminamos o cuando estamos eliminando señal biológica? Sinceramente el que escribe lo desconoce.

<sup>130</sup> Sequencing depth, size library.

hecho, veremos que se expresan mucho menos de lo que realmente lo hacen porque los otros han consumido una parte importante del total de lecturas. En otras muestras donde no aparecen tan expresados esto no ocurrirá. Si no se ajusta la composición del RNA los otros genes aparecen falsamente infraregulados en la muestra donde unos pocos se expresan mucho. El método que vamos a utilizar para corregir este efecto es el la media ajustada de M-valores (TMM)<sup>131</sup> entre cada par de muestras.

<sup>131</sup> Trimmed mean of M-values (TMM).

**Contenido GC** El contenido GC de un gen no cambia entre muestras.

Para un análisis de expresión diferencial no debiera influir. Algunos autores han indicado que puede que esto no sea tan cierto. La primera referencia en notar este efecto es [110].

**Longitud del gen** Genes más largos tenderán a tener más lecturas alineadas. Otra vez, entre muestras no cambia este factor y para un análisis de expresión diferencial no debiera tener influencia.

**Sobre la longitud del gen.** En [OshlackWakefield09] se comenta e ilustra el problema. ¿Qué efecto cabe esperar cuando tenemos genes con distinta longitud. En RNA-seq lo que tenemos son fragmentos de transcritos. Si el transcripto es más largo el número de lecturas que producen es mayor. Es cierto que si queremos comparar entre muestras este efecto es el mismo. La longitud es la misma en todas las muestras. Sin embargo, al tener más lecturas la potencia del test es mayor para genes más largo y por ello encontraremos más genes con expresión diferencial entre los genes más largos. Un mayor tamaño muestral se asocia a una mayor probabilidad de rechazar la hipótesis nula de igualdad de las medias para una misma diferencia entre medias. De hecho, se propone en el trabajo indicado una simple formalización de este hecho. Denotamos por  $X$  la variable aleatoria que nos da el número de lecturas alineadas sobre un gen. Olvidando otros posibles sesgos podemos asumir que el valor medio de  $X$  depende del total de transcritos  $N$  y de la longitud del gen según la siguiente relación

$$\mu_X = EX = cNL,$$

siendo  $c$  un valor constante. En resumen asumimos que la media de lecturas es proporcional al tamaño de la librería y a la longitud del gen. Si se asume que la media y la varianza son iguales<sup>132</sup> entonces

$$var(X) = \sigma_X^2 = cNL,$$

Si consideramos la variable  $X$  en dos muestras distintas tendremos  $X_i$  con  $i = 1, 2$  y tamaños de librería  $N_i$  con  $i = 1, 2$ . Si utilizamos para contrastar la igualdad de medias el estadístico

$$\frac{X_1 - X_2}{\sqrt{cN_1L + cN_2L}}.$$

La potencia del test depende del cociente entre la media y la desviación estándar de  $D = X_1 - X_2$ <sup>133</sup> que viene dado por

$$\delta = \frac{cN_1L - cN_2L}{\sqrt{cN_1L + cN_2L}} \propto \sqrt{L}.$$

Es decir,  $\delta$  es proporcional a la raíz cuadrada de la longitud del gen. Puede pensarse que un corrección simple como sería dividir el conteo

<sup>132</sup> Lo cual es cierto bajo la hipótesis de que la variable  $X$  sigue una distribución de Poisson.

<sup>133</sup> O error estándar.

por la longitud del gen corrige este efecto. Si aplicamos esta corrección no se resuelve el problema ya que entonces la diferencia sería  $D' = \frac{X_1}{L} - \frac{X_2}{L}$ . Como es bien conocido<sup>134</sup> tenemos que la media y varianza de  $\frac{X_i}{N_i}$  son <sup>134</sup> § 21

$$E\frac{X_i}{L} = \frac{EX_i}{L} \text{ y } var\left(\frac{X_i}{L}\right) = \frac{var(X_i)}{L^2}$$

Por tanto el cociente entre la media y la desviación estándar de  $D'$  no se modifica y vale

$$\delta = \frac{cN_1L - cN_2L}{\sqrt{cN_1L + cN_2L}} \propto \sqrt{L}.$$

### 6.14.1 RPKM

<sup>135</sup> El primer método de normalización se propuso en [102]. Supongamos que denotamos el conteo de interés por  $C$ . Es el número de lecturas alineadas sobre la característica de interés (exon, gen, etc.). El total de lecturas que podemos alinear es  $N$  (o tamaño de la librería). Y denotamos por  $L$  la longitud de la característica de interés en bp. Se define el **RPKM** como el siguiente cociente

$$RPKM = \frac{10^9 C}{NL}. \quad (6.4)$$

Esta cuantificación de la expresión del gen nos permite comparar genes entre sí dentro de una misma muestra o librería.

En el paquete [161, geneLenDataBase] tenemos las longitudes de los distintos transcritos para distintos organismos. En el siguiente código leemos los datos correspondientes a humanos (versión 19).

```
pacman::p_load("geneLenDataBase")
data(hg19.ensGene.LENGTH)
head(hg19.ensGene.LENGTH)

##           Gene Transcript Length
## 1 ENSG00000118473 ENST00000237247   4024
## 2 ENSG00000118473 ENST00000371039   4102
## 3 ENSG00000118473 ENST00000424320    964
## 4 ENSG00000118473 ENST00000320161   4693
## 5 ENSG00000118473 ENST00000371036   7294
## 6 ENSG00000118473 ENST00000407289   7901
```

<sup>135</sup> Reads per kilobase per million.

### 6.14.2 Método TMM: Media ajustada de M-valores

Es, quizás, el método más popular de normalización. Fue propuesto en [119]. Comentan los autores en [119, pág. 2] el siguiente ejemplo hipotético. Tenemos dos experimentos de secuenciación de RNA o muestras. Llamamos a estas dos muestras A y B. Supongamos dos conjuntos de genes disjuntos y con el mismo número de elementos:  $S_1$  y  $S_2$ . El primer conjunto se expresa igualmente en A y B, es decir, tenemos el mismo número de transcritos en ambas muestras. Sin embargo,  $S_2$  solamente se expresa en la muestra A y no en la muestra B. Además asumimos que no se expresa ningún otro gen. Si la profundidad de secuenciación es la misma en las dos muestras entonces el número de

transcritos que observaremos de un gen de  $S_1$  será la mitad en A que en B aunque sabemos que se expresa exactamente igual. En resumen, en la población (total de transcritos) el número total de transcritos de este gen es el mismo en ambas muestras. La probabilidad de observar uno de ellos depende de su frecuencia y del número de transcritos observados. Por ello, depende de que otros genes se expresen más o menos. En resumen, de la composición del RNA.

Sea  $X_{ij}$  ( $x_{ij}$ ) el conteo aleatorio (observado) del gen  $i$  en la muestra  $j$ . Denotamos  $L_i$  la longitud del gen  $i$  y  $m_j$  es el total de lecturas de la librería  $j$  (o tamaño de la librería  $j$ ). En [119] se asume que la media de  $X_{ij}$  verifica

$$E[X_{ij}] = \frac{\mu_{ij}L_i}{c_j}m_j,$$

siendo  $c_j = \sum_{i=1}^N \mu_{ij}L_i$ . Notemos que  $c_j$  nos está representando el total de RNA en la muestra. No se asume ninguna distribución de probabilidad específica para  $X_{ij}$ . La producción total de RNA,  $c_j$ , no es conocida. Sin embargo, sí es fácil estimar el cociente de estos valores para dos muestras, es decir, estimar el cociente  $f_j = c_j/c_r$ .

Elegimos una muestra como muestra de referencia. Por ejemplo, la muestra  $r$  denota a partir de ahora la muestra tomada (arbitrariamente) como de referencia.

Nos fijamos en la muestra  $j$  y vamos a determinar la constante por la que multiplicaremos los conteos originales. En lo que sigue tanto  $j$  como  $r$  son fijos y las cantidades definidas dependen de  $i$  que denota el gen. Se define

$$M_{ij}^{(r)} = \log_2 \frac{x_{ij}/m_j}{x_{ir}/m_r} = \log_2(x_{ij}/m_j) - \log_2(x_{ir}/m_r),$$

y

$$A_{ij}^{(r)} = \frac{1}{2} \left( \log_2(x_{ij}/m_j) + \log_2(x_{ir}/m_r) \right)$$

<sup>136</sup> Recordemos que varía solamente  $i$  y mantenemos fijos  $j$  y  $r$ .

Se eliminan los valores extremos tanto de los  $M_{ij}^{(r)}$  como de los  $A_{ij}^{(r)}$ .<sup>136</sup> En concreto eliminamos un porcentaje de los  $M_i$  más pequeños y el mismo porcentaje de los más grandes. Lo mismo hacemos para los valores  $A_i$ . También eliminamos aquellos índices  $i$  tales que  $x_{ij} = 0$  o bien  $x_{ir} = 0$ . El conjunto de índices  $i$  restante lo denotamos por  $G^*$ . Finalmente, el factor de normalización sería

$$\log_2(TMM_j^{(r)}) = \frac{\sum_{i \in G^*} w_{ij}^{(r)} M_{ij}^{(r)}}{\sum_{i \in G^*} w_{ij}^{(r)}}$$

con

$$w_{ij}^{(r)} = \frac{m_j - x_{ij}}{m_j x_{ij}} + \frac{m_r - x_{ir}}{m_r x_{ir}}.$$

La muestra de referencia  $r$  es fija y lo que acabamos de calcular es el factor por el que multiplicamos los conteos originales de la muestra  $j$ .

<sup>137</sup> Obviamente tenemos que

$TMM_r^{(r)} = 1$  y esta muestra de referencia no se normaliza.

<sup>138</sup> El código relativo a métodos de normalización procede de [66, capítulo 4].

Este factor viene dado por  $TMM_j^{(r)}$ .<sup>137</sup>

Tanto este método como los que siguen están implementados en [34, edgeR].<sup>138</sup>

Utilizamos los datos tamidata::PRJNA297664 que es un

```
data(PRJNA297664, package="tamidata")
```

Accedemos a la matriz de expresión con los conteos.

```
pacman::p_load(SummarizedExperiment)
counts0 = assay(PRJNA297664)
```

```
pacman::p_load(edgeR)
deg.n = DGEList(counts = counts0, group = rep(1, ncol(counts0)))
deg.n = calcNormFactors(deg.n, method = "TMM")
deg.n = estimateCommonDisp(deg.n)
counts1 = deg.n$pseudo.counts
```

En la figura 6.1(a) y (b) tenemos la función de densidad y de distribución estimadas de los conteos antes de la normalización para las distintas muestras. Los dibujos se generan con el siguiente código.

```
pacman::p_load("ggplot2")
counts0 = data.frame(counts0)
df0 = reshape2::melt(counts0)
png(paste0(dirTamiFigures, "PRJNA297664_counts0_ecdf.png"))
ggplot(df0, aes(x=value, colour = variable)) + stat_ecdf()
dev.off()
png(paste0(dirTamiFigures, "PRJNA297664_counts0_ecdf.png"))
ggplot(df0, aes(x=value, colour = variable)) + geom_density()
dev.off()

counts1 = data.frame(counts1)
df0 = reshape2::melt(counts1)
png(paste0(dirTamiFigures, "PRJNA297664_counts0_ecdf.png"))
ggplot(df0, aes(x=value, colour = variable)) + stat_ecdf()
dev.off()
png(paste0(dirTamiFigures, "PRJNA297664_counts0_ecdf.png"))
ggplot(df0, aes(x=value, colour = variable)) + geom_density()
dev.off()
```

### 6.14.3 Mediana de los cocientes

Se propuso en [7]. Se consideran los factores de normalización

$$s_j = \text{mediana}_{\{i: \tilde{m}_i \neq 0\}} \frac{x_{ij}}{\tilde{m}_i} \quad (6.5)$$

siendo  $\tilde{m}_i = \left( \prod_{j=1}^n x_{ij} \right)^{\frac{1}{n}}$ . La idea es normalizar para cada gen diviendo el conteo por la correspondiente media geométrica de los distintos conteos para un mismo gen. Una vez que tenemos los datos normalizados intra gen buscamos una constante de normalización por muestra como la correspondiente mediana en la muestra. Estas constantes son utilizadas no para normalizar directamente el conteo sino para incorporarlo como factor que multiplica a la media del conteo en los métodos DESeq y DESeq2.

Utilizando los mismos datos tamidata::PRJNA297664 el método anterior lo podemos aplicar con el siguiente código.

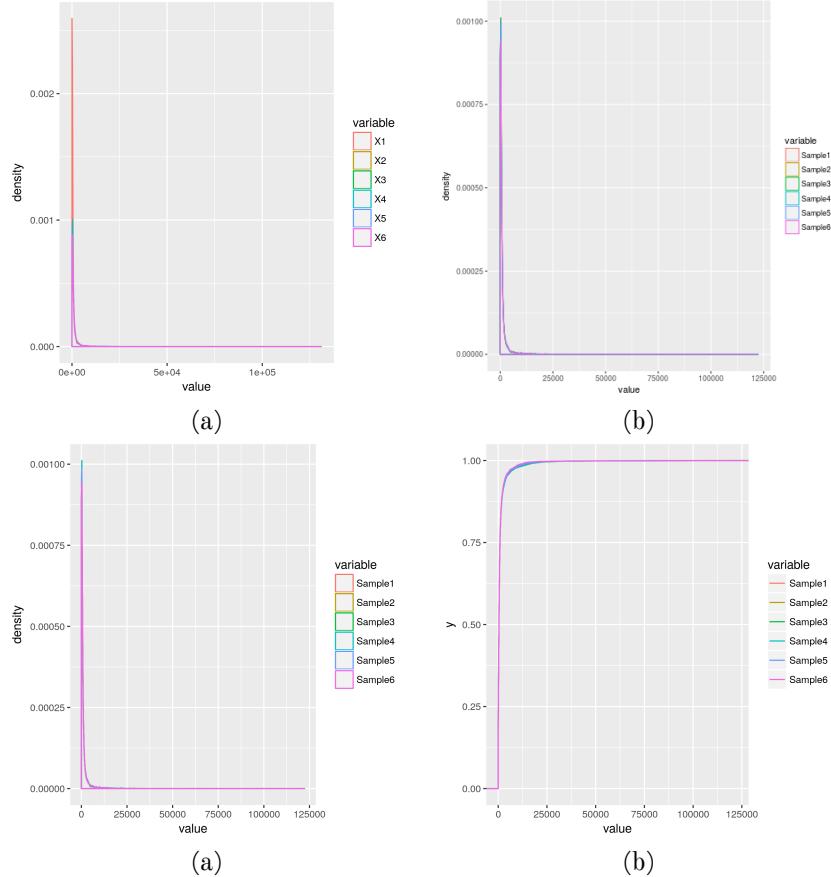


Figura 6.1: (a) Estimaciones de la función de densidad de los conteos para las distintas muestras de PRJNA297664 con los conteos originales. (b) Lo mismo que (a) con la función de distribución. En (c) y (d) tenemos los dibujos análogos utilizando los conteos después de aplicarles una normalización TMM.

```
deg.n = DGEList(counts = counts0, group = rep(1, ncol(counts0)))
deg.n = calcNormFactors(deg.n, method = "RLE")
deg.n = estimateCommonDisp(deg.n)
counts2 = deg.n$pseudo.counts
```

#### 6.14.4 Homogeneizando los percentiles

Este método forma parte del método propuesto en [121] y comentado en § 10.3. Indicamos aquí cómo aplicarlo a los datos tamidata::PRJNA297664. Como vemos en el código que sigue no se especifica método pues es el que aplica por defecto.

```
deg.n = DGEList(counts = counts0, group = rep(1, ncol(counts0)))
deg.n = estimateCommonDisp(deg.n)
counts3 = deg.n$pseudo.counts
```

### 6.15 Estudios de caso

#### 6.15.1 SRP064411

<sup>139</sup> En esta sección vamos a obtener los datos, mapear las lecturas, contar y finalmente normalizar los conteos para los datos SRP064411. La descripción la podemos encontrar en <http://www.ncbi.nlm.nih.gov/geo/query/acc.cgi?acc=GSE73681>.

1. Los datos los podemos bajar de <ftp://ftp-trace.ncbi.nlm.nih.gov/sra/sra-instant/reads/ByStudy/sra/SRP%2FSRP0642FSRP064411/>. También los podemos bajar de <http://www.ebi.ac.uk/arrayexpress/experiments/E-GEO-D-73681/?page=1&pagesize=500>.

Podemos bajar los datos accediendo a las direcciones indicadas y con el propio navegador. Suponemos que fijamos el directorio de trabajo donde tenemos los ficheros sra que acabamos de bajar.

2. Ejecutamos (y lo repetimos para cada fichero sra)

```
fastq-dump -I --split-files SRR2549634.sra
```

3. Bajamos el fichero de índices para poder trabajar con **Bowtie2**. Nuestro datos corresponden a Sacaromices cerevisiae. En [http://support.illumina.com/sequencing/sequencing\\_software/igenome.html](http://support.illumina.com/sequencing/sequencing_software/igenome.html) los tenemos para distintas especies. En concreto, bajamos [ftp://igenome:G3nom3s4u@ussd-ftp.illumina.com/Saccharomyces\\_cerevisiae/Ensembl/R64-1-1/Saccharomyces\\_cerevisiae\\_Ensembl\\_R64-1-1.tar.gz](ftp://igenome:G3nom3s4u@ussd-ftp.illumina.com/Saccharomyces_cerevisiae/Ensembl/R64-1-1/Saccharomyces_cerevisiae_Ensembl_R64-1-1.tar.gz).
4. Descomprimimos el fichero de índices y sacamos los ficheros.

```
gzip -d Saccharomyces_cerevisiae_Ensembl_R64-1-1.tar
      → .gz
tar xvf Saccharomyces_cerevisiae_Ensembl_R64-1-1.tar
```

Tenemos ahora un directorio `Saccharomyces\_cerevisiae_Ensembl_R64-1-1`.

<sup>139</sup> Todo el análisis incluido en esta sección lo tenemos <http://www.uv.es/ayala/docencia/tami/org/SRP064411.html> y podemos reproducirlo ejecutando el código como se indica allí. No se puede reproducir exclusivamente con el material de esta sección.

5. Alienamos las lecturas. Para ello ejecutamos, para cada fichero fastq, lo siguiente sustituyendo los directorios por los correspondientes a nuestro caso. Supongamos que denotamos dirIndice donde tenemos los índices. En mi caso dirIndice es */DOCENCIA/tami-data/Saccharomyces\_cerevisiae/Ensembl/R64-1-1/Sequence/Bowtie2Index/genome*. Ejecutamos lo siguiente sustituyendo dirIndice por el directorio correspondiente.

```
bowtie2 -x dirIndice -U SRR2549634_1.fastq -S
→ SRR2549634_1.sam
```

6. Repetimos el punto anterior para cada uno de los ficheros fastq. Notemos que al tener lecturas simples (no apareadas) solamente tenemos un fichero fastq por muestra.
7. Se obtienen unos buenos porcentajes de lecturas alineadas superiores al 95% en todos los casos.
8. Transformamos de formato sam a formato bam ordenado utilizando samtools.

```
samtools view -bS SRR2549634_1.sam | samtools sort -
→ SRR2549634_1
```

Y como siempre repetimos para cada uno de los ficheros sam.

9. Ejecutamos el siguiente código que nos produce el SummarizedExperiment con el que podemos trabajar.

```
library(Rsamtools)
library(GenomicFeatures)
sampleTable = read.table("bamfiles.txt")
dirActualData = "/home/gag/DOCENCIA/tami-data/SPR064411/sra/"
fls = paste(dirActualData,sampleTable[,1],sep="")
bamLst = BamFileList(fls, index=character(),yieldSize=100000,obeyQname=TRUE)
gtffile = "~/DOCENCIA/tami-data/Saccharomyces_cerevisiae/Ensembl/R64-1-1/Annotation"
txdb = makeTxDbFromGFF(gtffile, format="gtf")
genes = exonsBy(txdb, by="gene")
library(GenomicAlignments)
SPR064411_SE = summarizeOverlaps(features = genes, read=bamLst,
    mode="Union",
    singleEnd=TRUE, ## No son lecturas apareadas
    ignore.strand=TRUE,
    fragments=FALSE)
```

10. Tenemos que añadir los metadatos o datos fenotípicos. Son los siguientes donde lo que aparece en la primera columna es el nombre que tiene la muestra en nuestro SummarizedExperiment, los colnames.

	SampleName	Run	Treatment	Rep
SRR2549634_1.bam	GSM1900735	SRR2549634	0	1
SRR2549636_1.bam	GSM1900737	SRR2549636	0	3
SRR2549638_1.bam	GSM1900739	SRR2549638	1	2
SRR2549635_1.bam	GSM1900736	SRR2549635	0	2
SRR2549637_1.bam	GSM1900738	SRR2549637	1	1
SRR2549639_1.bam	GSM1900740	SRR2549639	1	3

Creamos un `DataFrame`<sup>140</sup> con estas variables.

<sup>140</sup> Que no es un data.frame.

```
SampleName = c("GSM1900735", "GSM1900737", "GSM1900739", "GSM1900736",
  "GSM1900738", "GSM1900740")
Run = c("SRR2549634", "SRR2549636", "SRR2549638", "SRR2549635",
  "SRR2549637", "SRR2549639")
Treatment = factor(c(0,0,1,0,1,1), levels=0:1, labels=c("wild-type", "sec66del"))
Rep = c(1,3,2,2,1,3)
colData(SRP064411_SE) = DataFrame(SampleName, Run, Treatment, Rep)
save(SRP064411_SE, file="SRP064411_SE.rda")
```

Los datos SRP064411\_SE los tenemos en [11, tamidata].



# **Parte III**

## **Expresión diferencial**



## Capítulo 7

# Expresión diferencial marginal

### 7.1 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.

### 7.2 Algo de notación

En lo que sigue denotaremos la matriz con los datos de expresión de los distintos genes como  $x = [x_{ij}]_{i=1,\dots,N; j=1,\dots,n}$  donde el valor  $x_{ij}$  nos da el nivel de expresión **observado** del gen  $i$ -ésimo en la muestra  $j$ -ésima. Asociada a la muestra  $j$  tenemos una covariable (o variable fenotípica)

$y$ . El valor de la covariable  $y$  en la muestra  $j$  la vamos a denotar por  $y_j$ .<sup>1</sup>

Como es de uso habitual en Estadística denotaremos

$$x_{\cdot j} = \sum_{i=1}^N x_{ij}; \quad x_{i \cdot} = \sum_{j=1}^n x_{ij},$$

$$\bar{x}_{\cdot j} = \frac{1}{N} \sum_{i=1}^N x_{ij}; \quad \bar{x}_{i \cdot} = \frac{1}{n} \sum_{j=1}^n x_{ij}.$$

En la notación previa hemos indicado 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  $X = [X_{ij}]_{i=1, \dots, N; j=1, \dots, n}$  denota la matriz de expresión aleatoria cuando todavía no se ha realizado la experimentación.

### 7.3 Selección no específica o filtrado independiente

El material de esta sección se ha obtenido de [69, capítulo 5] y [61]. Como material complementario es muy interesante [21].

La mayor parte de los genes no se expresan de un modo diferenciado entre condiciones. Y hay muchos.<sup>2</sup> Más genes evaluados *simultáneamente* suponen una dificultad mayor en el tratamiento estadístico posterior. Por ello si podemos de un modo simple *quitar* genes que no parecen tener actividad pues mucho mejor.<sup>3</sup>

Es costumbre realizar una preselección, un filtrado que no utilice información de la covariable  $y$ . Utilizamos el perfil de expresión del gen pero no información sobre las muestras, la posible clasificación de muestras en distintos grupos. A este filtrado se le da en la literatura la denominación de *filtrado independiente* o bien **selección no específica**.

Veamos un posible método. Consideramos el gen  $i$ -ésimo y tomamos una medida de localización como la mediana. Denotamos este valor por  $u_i$ . Determinamos un cierto percentil de estos valores  $u_i$ . Por ejemplo, si denotamos por  $p_1$  el orden del percentil entonces  $q_{p_1}(u)$  sería el correspondiente percentil de orden  $p_1$  de los valores  $u_i$ . Mantendremos los genes tales que su valor  $u_i$  sea mayor que  $q_p(u)$ . Ahora consideramos una medida de dispersión como el coeficiente intercuartílico. Del mismo modo,  $v_i$  sería el coeficiente intercuartílico del  $i$ -ésimo gen y el correspondiente percentil sería  $q_{p_2}(v)$  donde el orden  $p_2$  no tiene por qué coincidir con  $p_1$ . Nos quedamos con los genes en el siguiente conjunto

$$\{i : i = 1, \dots, N; u_i \geq q_{p_1}(u); v_i \geq q_{p_2}(v)\},$$

es decir, aquellos que tienen un nivel de expresión alto y una variabilidad alta. En otras palabras, genes que se expresan y que lo hacen de un modo variable en las distintas muestras.

---

<sup>1</sup> Por ejemplo, si estamos comparando los niveles de expresión en dos grupos (un grupo control y un grupo de enfermos por ejemplo) entonces  $y_j$  tomará el valor 1 si está en el primer grupo (control) y valor 2 si está en el segundo grupo (valor 2).

<sup>2</sup> A veces uno piensa que demasiados.

<sup>3</sup> Aunque esto puede tener muchos riesgos como descartar genes que actúan conjuntamente con otros y marginalmente no tengan una actividad apreciable. Es pues algo peligroso eliminar genes sin tener muy claro como lo hacemos.

En el método que acabamos de explicar podemos sustituir la mediana como localización y el rango intercuartílico como dispersión por la media y la desviación estándar.

Otro procedimiento que se ha propuesto y utilizado con frecuencia es que un gen se exprese con claridad en algunas muestras, lo que podemos llamar el método k-sobre-A. La idea es simple, un gen lo conservamos si en al menos k muestras tiene un nivel de expresión por encima de un valor mínimo A.

En lo que sigue vemos cómo utilizar estas ideas con R/Bioconductor. En § 7.3.1 se propone una opción muy sencilla para preseleccionar genes utilizando simplemente la función *apply*. En § 7.3.2 utilizamos el paquete [61, *genefilter*]. Finalmente en § 7.3.3 utilizamos la función *genefilter::nsFilter*.

Vamos a abordar el problema con los datos ALL::ALL. En primer lugar aplicamos la selección de muestras de § 5.4.

```
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)]
```

### 7.3.1 Utilizando apply

Vamos a implementar un par de procedimientos de selección independiente con la función *apply*.

Un procedimiento muy básico puede ser calcular para el rango intercuartílico del perfil de expresión de cada gen.

```
bcrneg.iqr = apply(exprs(bcrneg), 1, IQR)
```

Y luego podemos determinar el cuantil (por ejemplo, el percentil 0.5 o mediana) de los rangos intercuartílicos. Si estás por encima de este percentil conservamos el gen. De lo contrario, no lo consideramos en lo que sigue.

```
sel.iqr = (bcrneg.iqr > quantile(bcrneg.iqr, 0.5))
```

Una idea como la que acabamos de aplicar para seleccionar genes tiene el inconveniente de que si un grupo tiene pocos datos entonces aunque hubiera una expresión diferencial la variabilidad total no tiene porqué ser muy grande.

Ahora filtramos atendiendo a la mediana del perfil de expresión.

```
bcrneg.median = apply(exprs(bcrneg), 1, median)
sel.median = (bcrneg.median > quantile(bcrneg.median, 0.5))
```

Y nos quedamos con los genes que verifican ambas condiciones. Nuestro ExpressionSet original es *bcrneg*, ¿cómo nos quedamos con estos genes y todas las muestras? Con el siguiente código.

```
bcrneg1 = bcrneg[sel.iqr & sel.median,]
```

Media y desviación estándar

Sustituimos mediana y rango intercuartílico por media y desviación estándar respectivamente.

Es claro que podríamos sustituir la desviación estándar por alguna otra medida de variabilidad, por ejemplo, el rango intercuartílico. El método anterior se podría implementar con el siguiente código.

```
bcrneg.sd = apply(exprs(bcrneg), 1, sd)
sel.sd = (bcrneg.sd > quantile(bcrneg.sd, 0.5))
bcrneg.mean = apply(exprs(bcrneg), 1, mean)
sel.mean = (bcrneg.mean > quantile(bcrneg.mean, 0.5))
bcrneg2 = bcrneg[sel.sd & sel.mean,]
```

## k sobre A

Otra opción natural sería fijar un nivel de actividad mínima para un gen y quedarnos con aquellos que superen este nivel mínimo de actividad. Consideremos el siguiente criterio: si el nivel de expresión mínimo es  $c$  y tenemos  $n$  muestras podemos pedir que un gen determinado se considere activo si en al menos  $k$  muestras del total de  $n$  su nivel de expresión supere este nivel mínimo de actividad.

¿Cómo hacerlo con R? Empezamos fijando el nivel mínimo  $c$ . ¿Y cómo? Podemos ver los percentiles de todas las expresiones (todos los genes y todas las muestras).

```
quantile(exprs(bcrneg))
```

```
##          0%        25%        50%        75%       100%
## 1.984919  4.117796  5.468801  6.832439 14.044896
```

Nos quedamos con la mediana para  $c$ .

```
c = 5.468801
```

Determinamos qué niveles de expresión lo superan.

```
overc = exprs(bcrneg) > c
```

Podemos ver los resultados para la fila 433.

```
overc[433,]
```

Determinamos el número de muestras por fila que superan el valor  $c$ . Notemos que aplicamos una suma. Cuando sumamos un vector lógico (con valores TRUE y FALSE) entonces el valor TRUE se interpreta como el número 1 y el valor FALSE se interpreta como 0.

```
count.c = apply(overc, 1, sum)
```

Por ejemplo, las primeras filas son

```
head(count.c)

##   1000_at    1001_at 1002_f_at 1003_s_at    1004_at
##      79          3         0        77        76
##   1005_at
##      79
```

Y finalmente determinamos si estamos por encima de 5 muestras.

```
sel.c = count.c >= 5
```

Y veamos cuántos genes conservamos.

```
table(sel.c)

## sel.c
## FALSE  TRUE
## 4736 7889
```

Una vez elegido un procedimiento de los comentados eliminamos aquellos genes que no pasan el criterio de selección. Y con los genes que nos quedan seguiríamos estudiando si hay o no expresión diferencial. Observemos que no hemos utilizado para nada los grupos que pretendemos comparar.

### 7.3.2 Utilizando genefilter

En la anterior sección utilizamos funciones básicas de R para realizar la selección no específica. Podemos utilizar un paquete diseñado para esto. El paquete `genefilter` [61] nos permite implementar distintos procedimientos de filtrado independiente de un modo sencillo.

Seguimos utilizando los datos del ExpressionSet `bcrneg`. Supongamos un procedimiento de filtrado no específico tan simple como el siguiente: mantendremos aquellos genes tal que su nivel de expresión es al menos de 200 en al menos 5 de las muestras. Para hacer esto con el paquete [61, `genefilter`] hemos de realizar tres pasos:

1. Crear la función que implementa cada criterio de filtrado.
2. Combinar todos los criterios de filtrado en una única función.
3. Aplicar la función de filtrado final a la matriz de expresión.

El siguiente código implementa todo el procedimiento.

```
library(BioBase)
library(genefilter)
f1 = kOverA(5, 5.468801)
ffun = filterfun(f1)
wh1 = genefilter(exprs(bcrneg), ffun)
```

Finalmente podemos ver cuántos genes permanecen en nuestro estudio. Son aquellos que tiene el valor de `wh1` igual a `TRUE`.

```
table(wh1)
```

```
## wh1
## FALSE TRUE
## 4736 7889
```

Vamos a incorporar los otros dos criterios comentados previamente. Primero necesitamos una función que nos diga si la desviación estándar supera el valor  $c$ .

Primero vemos el modelo que nos ofrece la función `kOverA`.

```
kOverA = function (k, A = 100, na.rm = TRUE)
{
  function(x) {
    if (na.rm)
      x = x[!is.na(x)]
    sum(x > A) >= k
  }
}
```

Y nos definimos algo similar utilizando la desviación estándar.

```
sdOverc = function (c, na.rm = TRUE)
{
  function(x) {
    if (na.rm)
      x = x[!is.na(x)]
    sd(x) >= c
  }
}
```

Y ahora otra función que evalúe si el rango intercuartílico supera un cierto valor.

```
iqrOverc = function (c, na.rm = TRUE)
{
  function(x) {
    if (na.rm)
      x = x[!is.na(x)]
    IQR(x) >= c
  }
}
```

```
f1 = kOverA(5, 200)
f2 = sdOverc(150)
f3 = iqrOverc(72)
ffun = filterfun(f1,f2,f3)
wh123 = genefilter(exprs(bcrneg), ffun)
```

¿Qué genes verifican este criterio de preselección?

```
table(wh123)
```

```
## wh123
## FALSE
## 12625
```

### 7.3.3 Utilizando nsFilter

Supongamos que queremos aplicar la siguiente selección nos vamos a quedar con aquellas sondas tales que el rango intercuartílico (para todas las muestras) es mayor que la mediana de los rangos intercuartílicos. Luego vamos a necesitar conocer la anotación de los genes. Si esto es así podemos filtrar aquellos genes de los cuales desconocemos su anotación. Para ello necesitamos conocer primero el paquete de anotación que utilizan nuestros datos.<sup>4</sup>

```
annotation(ALL)
## [1] "hgu95av2"
```

Por tanto hemos de cargar este paquete.

```
library(hgu95av2.db)
```

Realizamos el filtrado correspondiente.<sup>5</sup>

```
bcrnegfilt1 = nsFilter(bcrneg, var.func=IQR, var.cutoff=0.5,
                        require.GOBP=TRUE)
```

La función que utilizamos para medir variabilidad podemos ir modificándola. Por ejemplo, podemos sustituir `var.func` por la varianza o por la desviación estándar. Así podríamos realizar

```
bcrnegfilt2 = nsFilter(bcrneg, var.func=sd, var.cutoff=0.5,
                        require.GOBP=TRUE)
```

¿Cómo podemos combinar ambas selección? Utilizando la función `nsFilter` tenemos un `ExpressionSet`. Una manera simple puede ser la siguiente.

```
sel = intersect(featureNames(bcrnegfilt1),
                 featureNames(bcrnegfilt2))
```

Y nuestro `ExpressionSet` filtrado sería

```
bcrneg1 = bcrneg[sel, ]
```

Y ya podemos seguir trabajando con los genes que hemos filtrado.

## 7.4 Fold-change

<sup>141</sup> 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  $\log_2$ ) de los valores originales los denotamos por  $u_{ij} = \log_2(x_{ij})$  y  $v_{ij} = \log_2(y_{ij})$ . Para cada gen,

---

<sup>4</sup> En <http://www.bioconductor.org/packages/release/data/annotation/> tenemos un listado de paquetes de anotación.

<sup>5</sup> Observemos que ahora no pasamos la matriz de expresión sino el `ExpressionSet` mismo.

<sup>141</sup> 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.

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 ([140]) lo define como

$$FC_i^{(1)} = \frac{\bar{x}_{i\cdot}}{\bar{y}_{i\cdot}}. \quad (7.1)$$

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

$$FC_i^{(2)} = \bar{u}_{i\cdot} - \bar{v}_{i\cdot}, \quad (7.2)$$

tenemos la diferencia de las medias de las log2 expresiones.

Un procedimiento<sup>142</sup> que se ha utilizado frecuentemente en la literatura de microarrays consiste en tomar el log2 del fold-change en la definición 7.2, 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

$$\log_2 \left( \frac{\bar{x}_{i\cdot}}{\bar{y}_{i\cdot}} \right) \geq c.$$

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.<sup>143</sup> 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 [140] o en [128] en donde esencialmente se modifica la estimación del error estándar.

## 7.5 Los peligros de la selección no específica

En este tema hemos presentado como habitual la combinación de una selección no específica o independiente de genes con un análisis

<sup>142</sup> Que no es estadístico y que por tanto no vamos a considerar en este manual.

<sup>143</sup> Up (down) regulation.

posterior marginal de los genes que no han sido seleccionados utilizando procedimientos de corrección por comparaciones múltiples. Este procedimiento, aún habitual en la literatura, tiene sus riesgos. Un trabajo de gran interés a consultar sobre este problema es [21].

## 7.6 Expresión diferencial de un solo gen

Seguimos trabajando con los datos golub. Cargamos los datos.

```
library(Bioconductor)
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*.

```
DDR1 = exprs(gse21942) [1,]
```

En la figura 7.1(a) mostramos los datos para las distintas muestras (con distintos colores y caracteres distintos).

```
pacman::p_load("ggplot2")
df=data.frame(id = 1:ncol(gse21942), expression = DDR1, type = golub.cl)
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 7.1(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
```

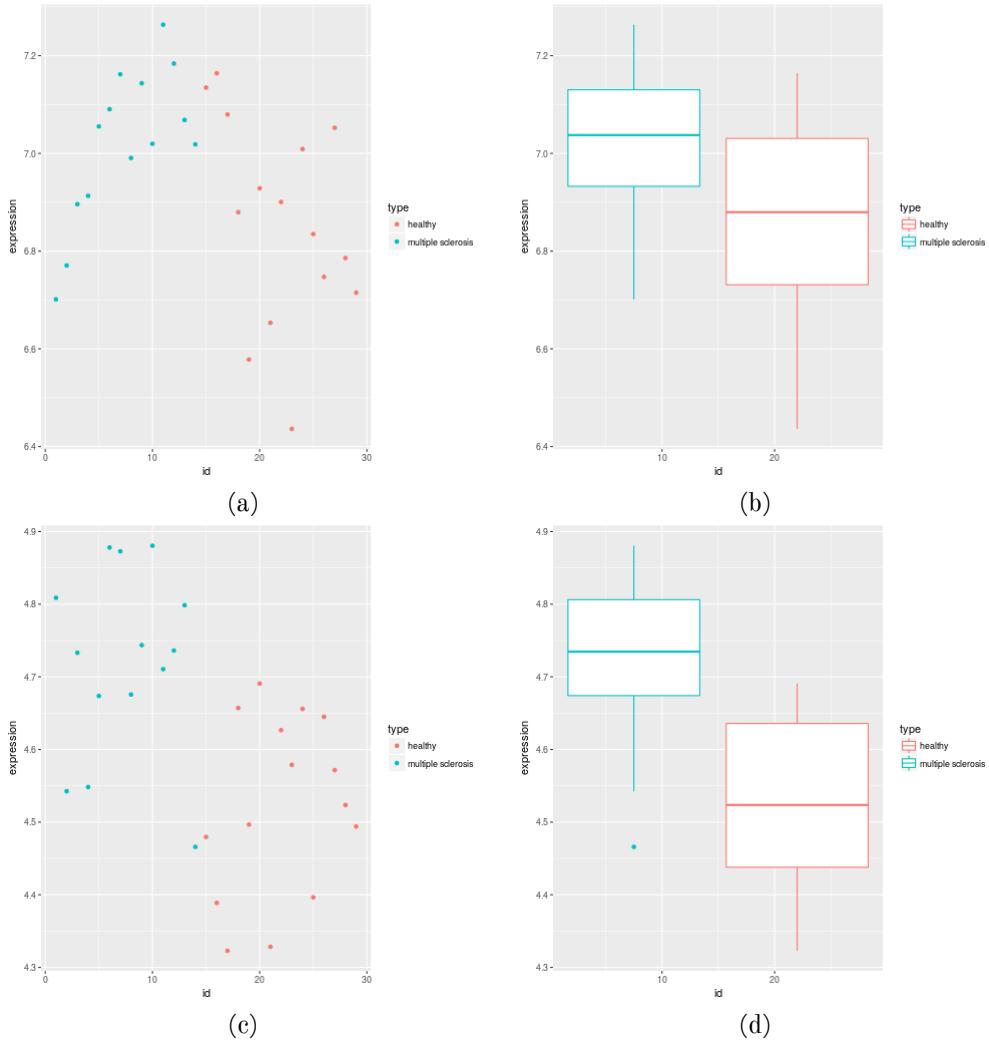


Figura 7.1: a) Perfil de expresión para el gen DDR1 utilizando casos y controles. b) Diagrama de cajas para gen DDR1 en los dos grupos. c) Perfil de expresión para el gen THRA en los dos grupos. d) Diagrama de cajas para gen DDR1 en los dos grupos.

Guardamos los datos en THRA.

```
THRA = exprs(gse21942)[7,]
```

En las figuras 7.1(c) y 7.1(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. En ?? recordamos el test de la t (o simplemente t-test) para comparar muestras independientes.

## 7.7 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  $X$ :  $X_1, \dots, X_n$  variables aleatorias independientes y con una misma distribución. Y otra muestra de  $Y$ :  $Y_1, \dots, Y_m$  variables aleatorias independientes y con la distribución. En § 25.3.2 se estudia con detalle los contrastes de hipótesis de comparación de medias de dos poblaciones normales.<sup>144</sup> Utilizamos los datos tamidata::gse21942.

<sup>144</sup> El capítulo § 25 está dedicado a esta cuestión.

```
data(gse21942, package="tamidata")
```

### Con genefilter::rowttests

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  $\alpha$  previamente elegido?

Si lo hacemos en este ejemplo, ¿cuántos 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  $\alpha$  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.

**Ej. 17** — Utilizando los datos `tamidata::gse1397` y el factor `pData(gse1397[, "type"]` realizar un análisis de expresión diferencial marginal

## Con GeneSelector::RankingTstat

Vamos a obtener todos los t-tests con `GeneSelector::RankingTstat`.

```
pacman::p_load(GeneSelector)
data(gse21942, package="tamidata")
ordT = RankingTstat(exprs(gse21942), pData(gse21942)[, "FactorValue..DISEASE.STATE."],
                     type="unpaired")
```

¿Qué clase nos devuelve?

```
class(ordT)

## [1] "GeneRanking"
## attr(", package"
## [1] "GeneSelector"
```

Los Slots que contiene esta clase `GeneRanking` son los siguientes.

```
getSlots("GeneRanking")
##           x      y   statistic   ranking
## "matrix"    "factor" "numeric"  "numeric"
##      pval      type     method
## "vector" "character" "character"
```

En particular tenemos el método `show`.

```
show(ordT)
## Ranking by ordinaryT,
## number of genes: 54675.
```

Con `toplist` tenemos

```
toplist(ordT)
##   index statistic      pval
## 1 37952  12.48294 1.000755e-12
## 2 34497 -12.33016 1.330269e-12
## 3 35846 -11.22211 1.131872e-11
## 4 10395  11.01843 1.703682e-11
## 5  2740 -10.91684 2.093037e-11
## 6 44792  10.86605 2.320943e-11
## 7 54126 -10.75827 2.893130e-11
## 8 32495 -10.74854 2.951439e-11
## 9 19131  10.61269 3.904854e-11
## 10 11672  10.59116 4.082890e-11
```

Ordena las filas atendiendo al p valor (`pval`) o bien al módulo de del estadístico. Como vemos nos devuelve la posición del gen en el `ExpressionSet` original, el valor del t-estadístico y el p-valor.

## 7.8 Comparando más de dos condiciones

En todo lo anterior hemos considerado la situación más habitual. Tenemos las muestras clasificadas en dos grupos o condiciones y pretendemos determinar aquellos genes que se expresan de un modo distinto entre ambas condiciones, que se expresan diferencialmente. Sin embargo, podemos tener más de dos grupos y pretender realizar un análisis similar. Esto nos conduce al análisis de la varianza (o breviadamente `anova`). Vamos a recordar brevemente qué es y cómo se aplica un análisis `anova`.

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  $Y_i$  denota la expresión aleatoria de un gen bajo la condición  $i$ . Se asume que

$$Y_i \sim N(\mu_i, \sigma^2),$$

es decir, que dicha expresión aleatoria se distribuye como una normal con media dependiente (posiblemente) de la condición pero con una varianza común  $\sigma^2$ .<sup>145</sup> La no existencia de expresión diferencial entre las

<sup>145</sup> Recordemos que en el caso de dos grupos podemos asumir o no una misma varianza.

condiciones se traduce, asumiendo este modelo probabilístico, en que las expresiones medias son la misma. En definitiva, *no existe expresión diferencial* es equivalente (bajo el modelo) a

$$H_0 : \mu_1 = \dots = \mu_I, \quad (7.3)$$

$$H_1 : \text{Existe } i \neq j \text{ } \mu_i \neq \mu_j. \quad (7.4)$$

Supongamos que, para un gen dado, denotamos por  $y_{ij}$  la  $j$ -ésima muestra observada bajo la condición  $i$  ( $i = 1, \dots, I$  y  $j = 1, \dots, n_i$ ). Se consideran las *sumas de cuadrado intra*

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

y la *suma de cuadrados entre* como

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

El estadístico para contrastar esta hipótesis nula es

$$F = \frac{SS(\text{Entre})/(I-1)}{SS(\text{Intra})/(n-I)}$$

*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,

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

**Ejemplo 7.1 (Anova con R)** Lo ilustramos la expresión bajo diferentes condiciones de un gen de los datos de expresión gse20986 ([§ 5.11](#)).

```
data(gse20986, package="tamidata")
```

Seleccionamos un gen cualquiera.

```
y = exprs(gse20986)[678,]
```

y realizamos un análisis de la varianza.

```
summary(aov(y ~ pData(gse20986)[, "tissue"]))

##                                     Df   Sum Sq  Mean Sq
## pData(gse20986)[, "tissue"]    3 0.005309 0.001770
## Residuals                      8 0.020212 0.002527
##                                     F value Pr(>F)
## pData(gse20986)[, "tissue"]      0.7  0.578
## Residuals
```

De acuerdo con el *p*-valor observado no rechazaríamos la hipótesis nula a ninguno de los niveles de significación habituales (0.05 o 0.01).

En el siguiente ejemplo analizamos, buscando diferencias entre cualquier par de grupos (tejidos en el ejemplo).

**Ejemplo 7.2 (GSE20986)** Vamos a realizar un análisis de expresión diferencial de los datos gse20986. Cargamos paquetes necesarios

```
library(multtest)
library(genefilter)
```

Tenemos cuatro grupos. Tenemos interés en la comparación entre todos los grupos (tejidos) al mismo tiempo. En definitiva buscamos aquellos genes que muestran una expresión diferencial entre al menos dos de los grupos. Un posible planteamiento es el modelo de análisis de la varianza y la comparación entre los grupos se reformula como una comparación entre valores medios.<sup>6</sup> Esto supone una comparación simultánea de las medias de los cuatro grupos y esto se realiza con un análisis de la varianza.

Calculamos los p-valores con la función genefilter::rowFtests.

```
gse20986.aov =
  rowFtests(gse20986, pData(gse20986) [, "tissue"])
```

Por ejemplo, vemos los dos primeros valores.

```
head(gse20986.aov, n=2)

##           statistic    p.value
## 1007_s_at   9.782456 0.004715281
## 1053_at     14.144595 0.001455658
```

## 7.9 Ejercicios

**Ej. 18** — 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.

**Ej. 19** — Utilizando el paquete [61, 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.

---

<sup>6</sup>No es la única opción. Podríamos usar una opción no paramétrica.

- 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.

**Ej. 20** — Realizar una selección no específica de los datos ALL utilizando el paquete [61, 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 8

# Comparaciones múltiples

### 8.1 Introducción

Empezamos con algunos comentarios bibliográficos. Esta sección se basa fundamentalmente en [40]. Sin embargo, un artículo a consultar es [116].

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  $i = 1, \dots, N$  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.

<sup>1</sup>

Estos contrastes nos lo planteamos para cada uno de los  $N$  genes que evaluamos. Denotemos por  $G = \{1, \dots, N\}$  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 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

---

<sup>1</sup> De este modo, creo que englobamos de un modo más natural el caso en que consideramos asociada a las muestras una covariante 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 8.1: Errores tipo I y II en contraste de múltiples hipótesis

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 8.1 (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  $\alpha$ . 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  $\alpha$ . 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  $c_\alpha$  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  $\alpha$ . 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 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.

---

<sup>2</sup>Una situación así la tenemos cuando observamos las muestras bajo dos condiciones distintas y utilizamos un test de la t de comparación de medias.

Las distintas medidas de error que se utilizan esencialmente cuantifican valores asociados a las variables que hemos definido en la tabla 8.1. 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.

**Tasa de error por comparación** <sup>3</sup> Por  $E(V)$  denotamos la media o valor medio de la variable  $V$ . Se define la tasa de error por comparación como el cociente entre la media de  $V$  y el número de contrastes.

$$PCER = \frac{E(V)}{N},$$

es la proporción esperada de errores tipo I entre las  $m$  decisiones que tomamos.

#### FWER: Tasa de error global <sup>4</sup>

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** <sup>5</sup> 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 dejáramos 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

---

<sup>3</sup>Per-comparison error rate.

<sup>4</sup>Familywise error rate.

<sup>5</sup>False discovery rate.

que se define como

$$FDR = E(Q) = E\left(\frac{V}{R}|R > 0\right)P(R > 0) + 0 \times P(R = 0) = E\left(\frac{V}{R}|R > 0\right)P(R > 0),$$

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  $\alpha$ .

**pFDR: Tasa de falsamente rechazados modificada** <sup>6</sup> 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  $\alpha$  cuando su tasa de error es menor o igual que  $\alpha$  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.

## 8.2 Control fuerte y control débil del error

Cuando hemos definido las tasas de error hemos considerado el comportamiento aleatorio de  $V$  y  $R$  (conjunto y marginal de hecho). Ese comportamiento aleatorio depende de qué hipótesis nulas son ciertas y qué hipótesis son falsas. Por ejemplo, cuando hemos definido

$$FWER = P(V \geq 1),$$

realmente la probabilidad depende de qué hipótesis nulas son ciertas y qué hipótesis nulas son falsas. Lo correcto hubiera sido escribir:

$$FWER = P(V \geq 1 | \cap_{i \in M_0} H_i),$$

es decir, la probabilidad de rechazar al menos una hipótesis nula equívocadamente condicionando a que se verifican las hipótesis de  $M_0$ .<sup>7</sup> Un

---

<sup>6</sup>Positive false discovery rate.

<sup>7</sup>Un detalle, no conocemos  $M_0$ .

detalle importante es que cuándo controlamos una tasa de error tipo I es saber qué hipótesis nulas asumimos que son ciertas. Un par de definiciones que son importantes.

**Definición 8.1 (Control fuerte)** *Se dice que un procedimiento de contraste de múltiples hipótesis realiza un control fuerte de la tasa de error tipo I elegida cuando dicha tasa de error es menor o igual que el límite que especifiquemos para cualquier combinación de hipótesis ciertas y falsas, en otras palabras, para cualquier subconjunto  $M_0 \subset \{1, \dots, m\}$  de hipótesis ciertas (el resto las suponemos falsas).*

**Definición 8.2 (Control débil)** *Se dice que un procedimiento de contraste de múltiples hipótesis realiza un control débil de la tasa de error tipo I elegida cuando dicha tasa de error es menor o igual que el límite que especifiquemos asumiendo que todas las hipótesis nulas son ciertas. Asumimos pues que se verifica la hipótesis nula*

$$H_0^C = \cap_{i=1}^N H_i.$$

En el control débil asumimos algo que no tiene muchos visos de realidad. Estamos asumiendo que **todas las hipótesis nulas son ciertas**. Si pensamos en nuestro problema estamos diciendo que ningún gen tiene una expresión diferencial entre condiciones. No sé pero mal lo tenemos si es así. Parece poco realista asumir esto. Lo natural es asumir que algunas hipótesis nulas son ciertas y otras falsas. Es decir, que existe un conjunto  $N_0$  que nos da las que son ciertas y aquellos que no están en  $N_0$  son falsas. ¿Y quién es  $N_0$ ? Ese es nuestro problema precisamente. En el control fuerte nos preparamos para cualquier  $N_0$  sea el que sea. En lo que sigue cuando pongamos probabilidades referidas a  $V$  (o cualquier otra variable) siempre han de entenderse como probabilidades condicionadas a la realidad (desconocida para nosotros), es decir, probabilidades condicionadas a  $\cap_{i \in N_0} H_i$  siendo  $N_0$  el conjunto de hipótesis nulas ciertas (en nuestro caso, los genes que están en  $N_0$  no se expresan de un modo diferencial y sí lo hacen los que no están en  $N_0$ ).

### 8.3 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 tendrá 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.

### 8.4 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  $\alpha_i$  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  $\alpha_i$  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  $\alpha$  el valor que especificamos para controlar alguna de las tasas de error tipo I.

## 8.5 Métodos que controlan la FWER

Veremos procedimientos que actúan en un solo paso y procedimientos por pasos bien de bajada o de subida.

### 8.5.1 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  $\alpha$  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

$$p_i \leq \frac{\alpha}{N} \quad (8.1)$$

o, equivalentemente, si

$$Np_i \leq \alpha. \quad (8.2)$$

Si tenemos en cuenta que  $0 < \alpha \leq 1$  entonces la desigualdad 8.2 es equivalente a que

$$\min\{Np_i, 1\} \leq \alpha. \quad (8.3)$$

De este modo el p-valor ajustado será  $\tilde{p}_i = \min\{Np_i, 1\}$ .<sup>8</sup>

**Método de Bonferroni** Este método nos da un control fuerte de la FWER al nivel  $\alpha$ . 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:

$$\tilde{p}_i = \min\{Np_i, 1\}$$

---

<sup>8</sup>El método de Bonferroni controla la FWER de un modo débil. Veámoslo. Denotemos por  $E_i$  el suceso consistente en que no rechazamos  $H_i$  o, si se prefiere, aceptamos  $H_i$ . Se busca que  $P(\cap_{i=1}^N E_i | \cap_{i=1}^N H_i) = 1 - \alpha$ . Las probabilidades que siguen se entienden que son condicionadas a que todas las hipótesis nulas son simultáneamente ciertas. Tenemos

$$P(\cap_{i=1}^N E_i) = 1 - P(\cup_{i=1}^N E_i^c) \leq 1 - \sum_{i=1}^N P(E_i^c). \quad (8.4)$$

Pero  $P(E_i^c)$  es la probabilidad de rechazar  $H_i$  siendo cierta  $\cap_{i=1}^N H_i$ . Supongamos que consideramos una región crítica de tamaño  $\alpha/m$ , es decir,  $P(E_i^c) \leq \alpha/m$ . Teniendo en cuenta 8.4 se sigue

$$P(\cap_{i=1}^N E_i) \leq 1 - \sum_{i=1}^N P(E_i^c) \leq 1 - m \frac{\alpha}{N}. \quad (8.5)$$

**Método de Sídák** Nos permite un control débil del FWER. Los p-valores ajustados son

$$\tilde{p}_i = 1 - (1 - p_i)^N.$$

Asume la independencia de los p valores sin ajustar, en definitiva, asume la independencia de los tests. En nuestro contexto no es muy asumible esta hipótesis pues los genes tienen expresiones relacionadas, no independientes.

**Método de minP en un solo paso** Los p-valores ajustados son

$$\tilde{p}_i = P\left(\min_{1 \leq l \leq N} P_l \leq p_i \mid H_o^C\right),$$

siendo  $H_o^C$  la hipótesis nula completa (la intersección de todas las hipótesis nulas) y  $P_l$  la variable aleatoria que nos da el p-valor aleatorio de la  $l$ -ésima hipótesis.

**maxT en un solo paso** Podemos considerar como una opción alternativa como el máximo de los valores de los estadísticos observados. Es el maxT en un solo paso donde los p-valores ajustados son

$$\tilde{p}_i = P\left(\max_{1 \leq l \leq N} |T_l| \geq |t_i| \mid H_o^C\right),$$

Los procedimientos basados en maxT o minP nos dan un control débil de FWER.

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.

### 8.5.2 Los métodos por pasos de bajada

Los p-valores originales son  $p_1, \dots, p_m$ . Los ordenamos de menor a mayor:

$$p_{r_1} \leq p_{r_2} \leq \dots \leq p_{r_N}.$$

Tendremos las correspondientes hipótesis nulas  $H_{r_1} \leq H_{r_2} \leq \dots \leq H_{r_N}$

**Método de Holm** Definimos

$$i^* = \min\{i : p_{r_i} > \frac{\alpha}{m - i + 1}\}$$

y rechazamos las hipótesis nulas  $H_{r_i}$  para  $i = 1, \dots, i^* - 1$ . Si no existe  $i^*$  entonces rechazamos todas las hipótesis nulas. Si consideramos los p-valores ajustados entonces vienen dados por

$$\tilde{p}_{r_i} = \max_{k=1, \dots, i} \{\min\{m - k + 1)p_{r_k}, 1\}\}$$

El método de Holm es menos conservador que el método de Bonferroni.

**Método de Sídák por pasos de bajada** Definimos los p-valores ajustados como:

$$\tilde{p}_{r_i} = \max_{k=1, \dots, i} \{1 - (1 - p_{r_k})^{m-k+1}\}.$$

**min P por pasos de bajada** Se definen los p-valores ajustados como:

$$\tilde{p}_{r_i} = \max_{k=1,\dots,i} \left\{ P \left( \min_{l \in \{r_k, \dots, r_m\}} P_l \leq p_{r_k} \mid H_o^C \right) \right\}$$

**max T por pasos de bajada** Los p-valores ajustados son:

$$\tilde{p}_{r_i} = \max_{k=1,\dots,i} \left\{ P \left( \max_{l \in \{r_k, \dots, r_m\}} |T_j| \geq |t_{r_k}| \mid H_o^C \right) \right\}$$

donde  $|t_{r_1}| \geq |t_{r_2}| \geq \dots \geq |t_{r_N}|$  son los estadísticos ordenados.

### 8.5.3 Método por pasos de subida

Vamos a ver el método de Hochberg. Se pretende, como en los anteriores, controlar el FWER a un nivel  $\alpha$ . Consideramos

$$i^* = \max \{ i : p_{r_i} \leq \frac{\alpha}{m-i+1} \}$$

y se rechazan las hipótesis nulas  $H_{r_i}$  para  $i = 1, \dots, i^*$ . Los p-valores ajustados de Hochberg son

$$\tilde{p}_{r_i} = \min_{k=i,\dots,N} \left\{ \min \{ (m-k+1)p_{r_k}, 1 \} \right\}.$$

Este procedimiento es una versión en sentido inverso del procedimiento de Holm.

## 8.6 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  $p_{r_1} \leq \dots \leq p_{r_N}$  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  $i = 1, \dots, i^*$ . Si no existe  $i^*$  entonces no rechazamos ninguna hipótesis. Los p-valores ajustados se definen como

$$\tilde{p}_{r_i} = \min_{k=i,\dots,N} \left\{ \min \left\{ \frac{N}{k} p_{r_k}, 1 \right\} \right\}.$$

**Benjamini y Yekutieli** Propuesto en [15]. Como en el anterior,  $p_{r_1} \leq \dots \leq p_{r_N}$  son los p-valores originales ordenados. En este caso los p-valores ajustados se definen como

$$\tilde{p}_{r_i} = \min_{k=i,\dots,N} \left\{ \min \left\{ \frac{m \sum_{j=1}^N 1/j}{k} p_{r_k}, 1 \right\} \right\}.$$

## 8.7 Utilizando genefilter y p.adjust

En esta sección incluimos análisis de expresión diferencial (marginal) utilizando los paquetes [111, multtest] y [61, genefilter]. No vamos a realizar en principio ninguna selección no específica y nos planteamos trabajar con todos los genes.

### 8.7.1 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-test's 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)
```

Podemos utilizar otros procedimientos de ajuste. Para ver las opciones lo mejor es consultar la ayuda de stats::p.adjust.

### 8.7.2 Genes con expresión diferencial

Lo primero es fijar una tasa de error  $\alpha$ . En concreto podemos considerar el valor  $\alpha = 0.05$ .

```
alpha = 0.05
```

Consideraremos 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
```

## 8.8 El q-valor

Esta sección se basa en [132]. Supongamos que ordenamos de menor a mayor los p-valores calculados para los distintos contrastes.

$$p_{(1)} \leq \dots \leq p_{(N)}$$

Cuando hacemos este ordenamiento, como hemos visto en los procedimientos anteriores, declarar significativo uno de estos p-valores supone declarar significativos a todos los que son menores o iguales que él. Los métodos anteriores buscaban un punto de corte para estos p-valores ordenados. Por debajo de este punto de corte admitimos que los genes se expresan diferencialmente (rechazamos al hipótesis nula correspondiente) y por encima aceptamos la hipótesis nula. La idea del q-valor es asociar a cada test un valor numérico que nos diga lo extremo que es su p-valor considerando el resto de p-valores. De un modo intuitivo, si consideramos un test determinado nos da la proporción esperada de falsos positivos en la que incurrimos cuando declaramos significativo ese test.

### ¿Qué es el q-valor?

Possiblemente el mejor modo de entender qué es el q-valor sea ver un buen procedimiento para estimarlo propuesto en [132] e implementado en el paquete [9, qvalue]. Fijamos un valor  $t$  y consideremos que rechazamos  $H_i$  cuando  $P_i \leq t$ .<sup>9</sup> Consideremos los siguientes valores

$$V(t) = |\{P_i : P_i \leq t; H_i \text{ es cierta}; i = 1, \dots, N\}| \quad (8.6)$$

y

$$R(t) = |\{P_i : P_i \leq t; i = 1, \dots, N\}| \quad (8.7)$$

Se puede aproximar la pFDR con

$$E\left[\frac{V(t)}{R(t)}\right]$$

Tenemos un gran número de contrastes  $N$  por ello aproximadamente se tiene que

$$E\left[\frac{V(t)}{R(t)}\right] \approx \frac{EV(t)}{ER(t)}.$$

---

<sup>9</sup>Como es habitual denotamos por  $P_i$  el p-valor antes de ser observado, esto es, el valor aleatorio antes de tomar los datos.

Podemos estimar  $R(t)$  simplemente contando el número de  $p_i$  observados que son menores o iguales a  $t$ ,

$$\hat{R}(t) = |\{p_i : p_i \leq t\}|$$

Para estimar  $V(t)$  hemos de tener en cuenta que si la hipótesis nula  $H_i$  es cierta entonces el p-valor aleatorio  $P_i$  se distribuye uniforme en el intervalo  $[0, 1]$  y por tanto la probabilidad de que sea menor o igual a  $t$  es precisamente  $t$ ,

$$P(P_i \leq t | H_i) = t.$$

Por tanto

$$EV(t) = N_0 t.$$

Pero no conocemos  $N_0$  número de hipótesis nulas ciertas. Lo que vamos a hacer es estimar la proporción de hipótesis nulas ciertas  $\pi_0 = N_0/N$  (notemos que el total de hipótesis,  $N$ , sí es conocido). Bajo  $H_i$ , el p-valor aleatorio  $P_i$  es uniforme en  $[0, 1]$ , por tanto los p-valores correspondientes a hipótesis nulas ciertas se distribuyen sobre todo el intervalo  $[0, 1]$ , sin embargo, aquellos p-valores muy pequeños, muy próximos a cero, es más probable que correspondan a hipótesis nulas falsas. Fijamos un valor  $\lambda \in [0, 1]$ , un estimador razonable para  $\pi_0$  es

$$\hat{\pi}_0 = \frac{|\{p_i : p_i > \lambda; i = 1, \dots, N\}|}{N(1 - \lambda)}.$$

Podemos pues estimar pFDR con

$$\widehat{pFDR} = \frac{\hat{\pi}_0 N t}{|\{p_i : p_i \leq t\}|}.$$

El q-valor asociado a un contraste sería el mínimo valor de pFDR que se alcanza cuando el contraste es rechazado. Es decir, el q-valor asociado al test  $i$ -ésimo sería

$$q(p_i) = \min_{t \geq p_i} pFDR(t)$$

y su estimador sería

$$\hat{q}(p_i) = \min_{t \geq p_i} \widehat{pFDR}(t).$$

El estimador  $\hat{\pi}_0$  depende de un valor  $\lambda$ . De hecho el procedimiento exacto no utiliza un valor  $\lambda$  concreto sino que estima utilizando distintos valores y luego hace un ajuste tomando como estimador el valor estimado a partir del ajuste en 1.

El algoritmo exacto es el siguiente:

1. Sean  $p_{(1)} \leq \dots \leq p_{(N)}$  los p-valores ordenados.
2. Para un rango de valores de  $\lambda$ , por ejemplo,  $\lambda$  de 0 a 0.95 con incrementos de 0.01 calculamos

$$\hat{\pi}_0(\lambda) = \frac{|\{p_i : p_i > \lambda; i = 1, \dots, N\}|}{N(1 - \lambda)}.$$

3. Determinamos  $\hat{f}$  el spline natural cúbico con 3 grados de libertad a partir de los valores  $\hat{\pi}_0(\lambda)$ .

4. Fijamos como estimador de  $\pi_0$  el valor

$$\hat{\pi}_0 = \hat{f}(1).$$

5. Calculamos

$$\hat{q}(p_{(m)}) = \min_{t \geq p_{(N)}} \frac{\hat{\pi}_0 N t}{|\{p_i : p_i \leq t\}|} = \hat{\pi}_0 p_{(N)}.$$

6. Para  $i = N - 1, \dots, 1$  calculamos

$$\hat{q}(p_{(i)}) = \min_{t \geq p_{(i)}} \frac{\hat{\pi}_0 N t}{|\{p_i : p_i \leq t\}|} = \min\left\{\frac{\hat{\pi}_0 N p_{(i)}}{i}, \hat{q}(p_{(i+1)})\right\}.$$

7. El q-valor estimado para el  $i$ -ésimo contraste más significativo es  $\hat{q}(p_{(i)})$ .

### Calculando el q-valor

El método de estimación que hemos visto en § 8.8 está implementado en el paquete [9, qvalue].<sup>10</sup>

Utilizamos los datos tamidata::gse1397.

```
pacman::p_load("Biobase")
data(gse1397, package="tamidata")
tt = genefilter::rowttests(gse1397, pData(gse1397) [, "type"])
pvalue = tt$p.value
```

Utilizamos el paquete [9, qvalue].

```
pacman::p_load(qvalue)
qobj = qvalue(pvalue)
```

Podemos ver una descripción gráfica con

```
plot(qobj)
```

que aparece en la figura 8.1. En concreto tenemos:

**Arriba izquierda:** Los distintos valores estimados de  $\pi_0$ , proporción de hipótesis nulas ciertas, cuando variamos el valor de  $\lambda$ . El valor estimado por el método antes propuesto nos da un 96.3% de genes sin expresión diferencial.

**Arriba derecha:** La transformación que nos lleva de los p-valores originales (ordenados de menor a mayor) a los q-valores.

**Abajo izquierda:** Para distintos puntos de corte para los q-valores el número de genes que declaramos con expresión diferencial significativa.

**Abajo derecha:** En abscisas el número de test significativos y en ordenadas el número de falsos positivos esperado.

Si simplemente queremos los q-valores los obtenemos con

---

<sup>10</sup>También se pueden bajar programas que no requieren R para distintos sistemas operativos. Se puede encontrar en <http://genomics.princeton.edu/storeylab/qvalue/>.

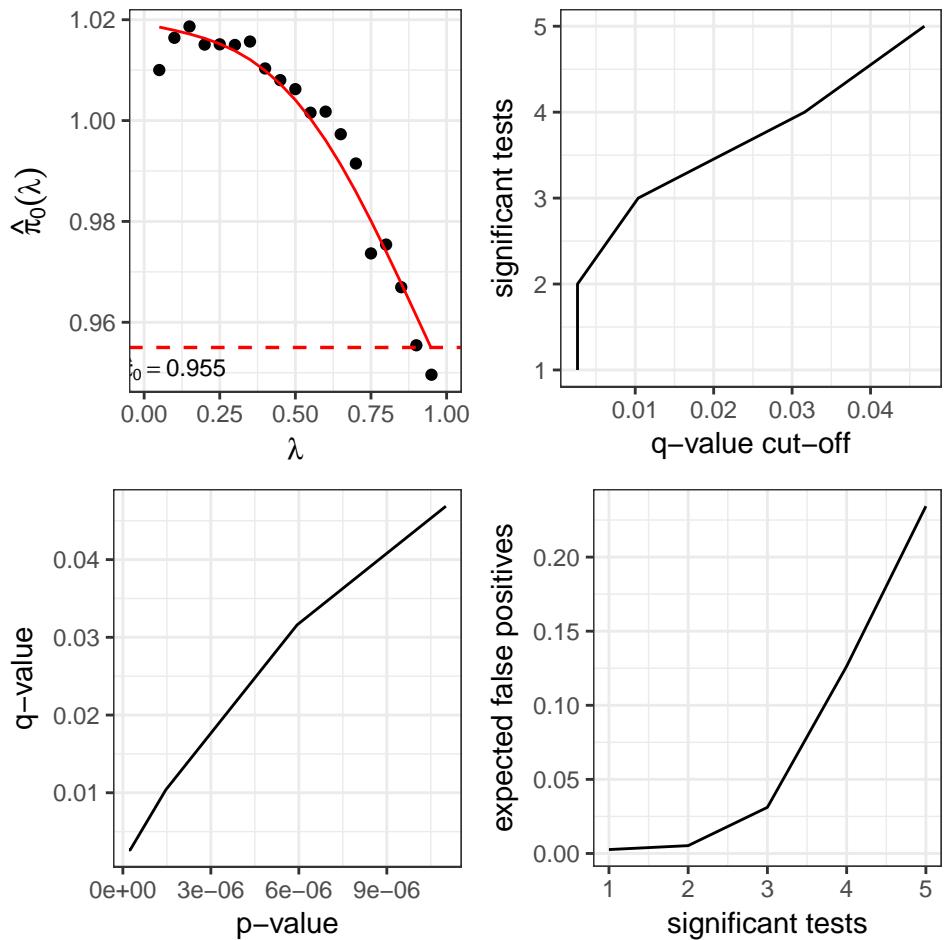


Figura 8.1: Distintos gráficos asociados a un objeto `qvalue::qvalue`.

```
q.value = qvalue(pvalue)$qvalues
```

## 8.9 Ejercicios

**Ej. 21** — 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.<sup>146</sup>
3. Utilizando el método de corrección de Benjamini-Hochberg obtener los p-valores ajustados.<sup>147</sup>
4. Si utilizamos un valor de FDR igual a 0.05: ¿qué genes declaramos como significativos?
5. Repetir los apartados anteriores comparando las 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.

**Ej. 22** — 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 base::intersect y Biobase::featureNames.

<sup>146</sup> Utilizad genefilter::rowttests.

<sup>147</sup> Utilizad [111, multtest].

# Capítulo 9

## Expresión diferencial con microarrays

### 9.1 Método SAM

Es la abreviatura de **S**ignificance **A**nalysis of **M**icroarrays. Fue propuesto en [140].<sup>1</sup>

#### 9.1.1 El método

El conjunto de muestras, como es habitual, lo denotaremos como por  $\{1, \dots, n\}$ . Empezamos con el caso en que pretendemos comparar dos grupos. Esta información la podemos tener con un vector  $y = (y_1, \dots, y_n)$  donde  $y_j$  vale 1 si la muestra  $j$  está en el primer grupo y vale 2 si la muestra en esa columna está en el segundo grupo. Para un gen dado las expresiones del grupo 1 (o del grupo 2) corresponden con los valores donde  $y_j = 1$  (respectivamente  $y_j = 2$ ). Consideramos  $J_g = \{j : y_j = g\}$  con  $g = 1, 2$ . Tendremos  $J_1$  y  $J_2$  donde  $J_g$  son las muestras del grupo  $g$  con  $g = 1, 2$ .<sup>2</sup> Denotamos por  $n_1$  y  $n_2$  los cardinales de  $J_1$  y  $J_2$ . Podemos denotar

$$\bar{x}_{ig} = \bar{x}_{i,J_g} = \frac{\sum_{j \in J_g} x_{ij}}{n_g}$$

siendo  $n_g$  el número de muestras en  $J_g$ . Para el  $i$ -ésimo gen consideramos la *diferencia relativa* dada por

$$d_i = \frac{\bar{x}_{i1} - \bar{x}_{i2}}{s_i + s_0} \quad (9.1)$$

donde

$$s_i = \sqrt{a \sum_{g=1}^2 \sum_{j \in J_g} (x_{ij} - \bar{x}_{i,g})^2} \quad (9.2)$$

<sup>1</sup>Tiene un gran interés consultar la dirección <http://www-stat.stanford.edu/~tibs/SAM/>. De hecho, no se puede entender el método tal cual se aplica hoy a partir de la referencia original. Ha habido una gran evolución posterior del procedimiento incluyendo el manejo de datos de secuencia. Existe una versión comercial para su uso con Excel. En la parte de los detalles técnicos del manual de la versión comercial se puede encontrar la explicación detallada del método y, por lo tanto, de las salidas que realizan las funciones. Este manual técnico lo podemos conseguir en <http://www-stat.stanford.edu/~tibs/SAM/sam.pdf>.

<sup>2</sup>Notemos que  $J_1$  y  $J_2$  son una partición de  $\{1, \dots, n\}$  de modo que  $J_1 \cup J_2 = \{1, \dots, n\}$  y son disjuntos  $J_1 \cap J_2 = \emptyset$ .

y

$$a = \frac{1/n_1 + 1/n_2}{n_1 + n_2}$$

Observemos que el estadístico  $d_i$  es el estadístico del contraste de igualdad de medias o simplemente el t-estadístico al que le modificamos el error estándar. ¿Para qué se hace esta modificación? Si el valor  $s_i$  es muy pequeño entonces el valor  $d_i$  es muy grande. En valores de expresión pequeños esto puede pasar. Los distintos valores  $d_i$  no son comparables para los distintos genes (la expresión que utilizan los propios autores es que no son intercambiables). Por ello se le suma un valor  $s_0$  que *estabiliza* la varianza (y nos permite comparar los t-valores entre sí). Este valor hay que *estimarlo* o calcularlo a partir de los propios datos (§ 9.1.2). Veamos el procedimiento enumerando los pasos.

1. Calculamos los estadísticos  $d_i$  según la ecuación 9.1.
2. Ordenamos de menor a mayor los  $d_i$ 's observados:  $d_{(1)} \leq \dots \leq d_{(N)}$  de modo que  $d_{(i)}$  es el i-ésimo menor.
3. Realizamos  $B$  permutaciones aleatorias del vector  $y$  que nos indica el grupo de la columna. Para cada permutación calculamos los nuevos valores  $d_i$  y los denotamos por  $d_i(b)$ . Ordenamos, como antes, estos nuevos valores de menor a mayor:  $d_{(1)}(b) \leq \dots \leq d_{(N)}(b)$ .
4. Calculamos, para las  $B$  permutaciones, el valor medio de estos estadísticos ordenados, es decir, calculamos

$$\bar{d}_{(i)} = \sum_{b=1}^B \frac{d_{(i)}(b)}{B}.$$

Bajo la hipótesis nula de que no hay diferencias entre los dos grupos los valores de  $d_{(i)}$  y de  $\bar{d}_{(i)}$  debieran de ser *similares*.

5. Representamos gráficamente  $d_{(i)}$  frente a  $\bar{d}_{(i)}$ .
6. Para un valor fijo positivo  $\Delta$ , empezando en el origen y moviéndonos a la derecha determinamos el primer  $i = i_1$  tal que  $d_{(i)} - \bar{d}_{(i)} > \Delta$ . Todos los genes a la derecha de  $i_1$  son llamados *significativamente positivos* (estos genes verifican la desigualdad anterior). Del mismo modo, empezando en el origen y moviéndonos a la izquierda determinamos el primer  $i = i_2$  tal que  $\bar{d}_{(i)} - d_{(i)} > \Delta$ . Todos los genes con la diferencia anterior más a la izquierda de  $i_2$  los llamamos *significativamente negativos*. Para cada  $\Delta$  definimos un punto de corte superior como

$$u(\Delta) = \min\{d_i : i \text{ es significativamente positivo}\}$$

y

$$l(\Delta) = \max\{d_i : i \text{ es significativamente negativo}\}$$

7. Para distintos valores de  $\Delta$ , calculamos:

- El número total de genes significativos (como hemos visto en el paso anterior).

- En la  $b$ -ésima permutación aleatoria hemos obtenido  $d_i(b)$  con  $i = 1, \dots, N$ . Calculamos el número de genes *falsamente llamados* en la aleatorización  $b$ -ésima como

$$c_b = |\{i : d_i(b) > u(\Delta) \text{ o } d_i(b) < l(\Delta)\}|$$

Tendremos los valores  $c_b$  con  $b = 1, \dots, B$ . Notemos que los valores  $d_i(b)$  han sido calculados bajo la hipótesis de no asociación, es cierta la hipótesis nula para cada gen. En consecuencia debieran de estar en el intervalo  $[l(\Delta), u(\Delta)]$ . Calculamos la mediana y el percentil de orden 0.90 de los valores  $c_b$  con  $b = 1, \dots, B$  y los denotamos como  $q_{0.5}(c)$  y  $q_{0.90}(c)$ .

8. Vamos a estimar  $\pi_0$ , la proporción de genes sin expresión diferencial, de genes que no se asocian con la covariable  $y$ . En definitiva cuál es la proporción de hipótesis nulas ciertas.

(a) Calculamos los percentiles de orden 0.25 y 0.75,  $q_{0.25}$  y  $q_{0.75}$ , de todos los valores  $d_i(b)$  que hemos obtenido realizando las permutaciones:  $\{d_i(s); i = 1, \dots, N; s = 1, \dots, B\}$  (tenemos un total de  $N \times B$  valores).

(b) Calculamos

$$\hat{\pi}_0 = \frac{|\{d_i : d_i \in (q_{0.25}, q_{0.75})\}|}{N/2}$$

donde  $\{d_1, \dots, d_N\}$  son los  $d$  valores originales.

(c) Truncamos el valor de  $\hat{\pi}_0$  en 1, es decir, consideramos

$$\hat{\pi}_0 = \min\{\hat{\pi}_0, 1\}.$$

<sup>3</sup>

9. Multiplicamos la mediana,  $q_{0.5}(c)$ , y el percentil 0.9,  $q_{0.90}(c)$ , determinados en el paso 7 por  $\hat{\pi}_0$ .
10. Elegimos un valor de  $\Delta$  y determinamos el conjunto de genes significativos.
11. La tasa de falsos positivos FDR es estimada como el cociente entre la mediana,  $q_{0.5}(c)$ , (o el percentil 0.90,  $q_{0.90}(c)$ ) del número de genes falsamente llamados y el número de genes declarados significativos.
12. El **q-valor** (que aparece en la salida de [137, samr]) es la tasa pFDR cuando la lista de genes significativos está compuesta por este gen y todos los genes que son más significativos que este. Se calcula buscando cuál es el valor más pequeño de  $\Delta$  para el cual este gen es declarado como significativo. Si este valor es  $\hat{\Delta}$  entonces la pFDR asociada a este valor es el q-valor asociado al gen.

---

<sup>3</sup>En la opción en que estamos comparando varios grupos que vemos más adelante los valores de  $d$  son todos positivos y se trabaja con los percentiles 0 y 0.5.

### 9.1.2 Cálculo de $s_0$

La selección de  $s_0$  se hace del siguiente modo:

1. Consideramos  $s = (s_1, \dots, s_N)$  las desviaciones estándar muestrales.
2. Sea  $q_\alpha(s)$  el percentil de orden  $\alpha$  de los valores  $s_i$ . Definimos

$$d_i^{(\alpha)} = \frac{\bar{x}_{i1} - \bar{x}_{i2}}{s_i + q_\alpha(s)}.$$

3. Consideramos los percentiles  $q_{j/100}(s)$  con  $j = 1, \dots, 100$ .

4. Para cada  $\alpha \in \{0, 0.1, \dots, 1\}$ :

- (a) Se calcula

$$v_j = \frac{1}{0.64} \text{MAD}(\{d_i^{(\alpha)} : s_i \in [q_{j/100}(s), q_{(j+1)/100}(s)]\})$$

para  $j = 1, \dots, 100$  donde MAD es la mediana de las desviaciones absolutas respecto de la mediana ([§ 20.4](#)).

- (b) Calculamos el valor  $CV(\alpha)$ , el coeficiente de variación de los valores  $v_j$ .<sup>148</sup>
- (c) Se elige como valor de  $\alpha$ :  $\hat{\alpha}$  que minimiza  $CV(\alpha)$ .
- (d) Finalmente tomamos para  $s_0$ :  $\hat{s}_0 = q_{\hat{\alpha}}(s)$ .

<sup>148</sup> El coeficiente de variación es el cociente de la media y la desviación estándar.

### 9.1.3 SAM con samr

Utilizamos los datos golub. Cargamos el paquete [[137](#), samr].

```
pacman::p_load("samr")
```

Realizamos el análisis. Tenemos dos grupos y en el tipo de respuesta le indicamos la opción *Two class unpaired*. La variable que indica el grupo ha de tomar valores 1 y 2. Vamos a tomar una tasa de falsamente rechazados o FDR muy pequeña. En concreto,  $FDR = 0.001$ .

```
data(golub, package="multtest")
fac0 = golub.cl +1
samfit = SAM(golub, fac0, resp.type="Two class unpaired",
nperms=1000, fdr.output=.001)
```

¿Qué valor de  $\Delta$  se ha utilizado?

```
samfit$del
##     delta
## 1.472335
```

¿Y qué genes encuentra el procedimiento significativo?

```
(sigtabla = samfit$siggenes.table)
```

Podemos comprobar en la salida anterior que diferencia entre genes *up* y genes *down*. ¿Qué es un gen *up* en este paquete? Indica asociación positiva entre expresión y la covariable *y* que nos indica el grupo (para golub recordad que la covariable es golub.cl que codifica ALL como 0 y AML como 1). En definitiva un gen *up* es que tiene una media mayor cuando la covariable es mayor, es decir, para los datos golub que la media en el grupo AML (valor de golub.cl 1) es mayor que la media en el grupo ALL (valor de golub.cl 0). Como es lógico los genes *down* son lo contrario. También podemos ver, para cada uno de ellos: el valor de  $d_i$ , su numerador y denominador, el cociente de las medias y el q-valor.

Tenemos también una ilustración gráfica en donde en abscisas nos muestra  $\bar{d}_i$  y en ordenadas  $d_i$ . La línea en trazo continuo indica la línea donde  $\bar{d}_i = d_i$ . Las líneas en trazo discontinuo por arriba y por abajo indican los genes *up* y los genes *down*. En la figura 9.1 podemos ver los resultados.

```
plot(samfit)
```

```
## pdf
## 2
```

Podemos ver también una exploración de posibles valores para  $\Delta$  con

```
head(samfit$delta.table, n=1)

##           delta # med false pos 90th perc false pos
## [1,] 1.472335      0.5368732      3.758112
##           # called median FDR 90th perc FDR      cutlo
## [1,]      337 0.001593095     0.01115167 -2.675221
##           cuthi
## [1,] 2.678885
```

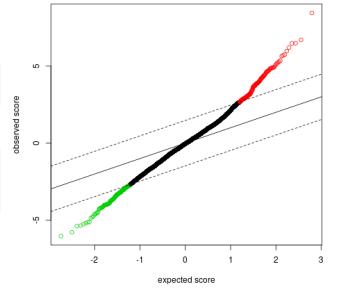


Figura 9.1: Método SAM utilizando [137, samr]. En negro los genes no significativos. En rojo los genes *up* que corresponden con asociación positiva con la covariable *y* en verde los que tienen asociación negativa.

### 9.1.4 SAM con otro tipo de covariables

Acabamos de utilizar este procedimiento en la comparación de dos grupos. Este mismo procedimiento se puede adaptar a los casos en que el vector *y* (covariable) nos indica la pertenencia de la muestra a más de un grupo o bien es un valor numérico.

El método es el mismo modificando las definiciones de  $d_i$  y  $s_i$  dadas en las ecuaciones 9.1 y 9.2.

**Comparación de más de dos grupos** Tenemos  $K$  grupos a comparar ( $K > 2$ ) por lo que  $y_j \in \{1, \dots, K\}$ . Supongamos que  $J_k$  denota los índices de las observaciones en grupo  $k$ .<sup>4</sup> Y supongamos que  $n_k$  es el cardinal de  $J_k$  ( $\sum_{k=1}^K n_k = n$ ). Utilizando la notación de ?? definimos  $d_i$  como

$$d_i = \frac{r_i}{s_i + s_0} \quad (9.3)$$

con

$$r_i = \sqrt{\frac{\sum_{k=1}^K n_k \sum_{j=1}^K n_k (\bar{x}_{i,J_k} - \bar{x}_{i,\cdot})^2}{\prod_{k=1}^K n_k}} \quad (9.4)$$

<sup>4</sup> Los conjuntos  $J_k$  son disjuntos entre sí y su unión es el total de muestras, es decir,  $\cup_{k=1}^K J_k = \{1, \dots, n\}$ .

y

$$s_i = \sqrt{\frac{1}{\sum_{k=1}^K (n_k - 1)} \left( \sum_{k=1}^K \frac{1}{n_k} \right) \sum_{k=1}^K \sum_{j \in J_k} (x_{ij} - \bar{x}_{i,J_k})^2}. \quad (9.5)$$

**Covariable continua** Supongamos que los valores de  $y$  son valores numéricos en este caso definimos  $d_i$  como en 9.3 y los valores de  $r_i$  y  $s_i$  son

$$r_i = \frac{\sum_{j=1}^n y_j (x_{ij} - \bar{x}_{i.})}{\sum_{j=1}^n (y_j - \bar{y})^2} \quad (9.6)$$

y

$$s_i = \frac{\hat{\sigma}_i}{\sqrt{\sum_{j=1}^n (y_j - \bar{y})^2}} \quad (9.7)$$

donde

$$\hat{\sigma}_i = \sqrt{\frac{\sum_{j=1}^n (x_{ij} - \hat{x}_{ij})^2}{n - 2}}, \quad (9.8)$$

y

$$\hat{x}_{ij} = \hat{\beta}_{i0} + r_i y_j, \quad (9.9)$$

$$\hat{\beta}_{i0} = \bar{x}_{i.} - r_i \bar{y}. \quad (9.10)$$

### 9.1.5 Ejercicios

#### Ejercicio 23

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 21.

#### Ejercicio 24

Utilizando los datos tamidata::gse20986 se pide determinar los genes significativos cuando comparamos los cuatro grupos considerados en el estudio.

## 9.2 Limma

Hasta el momento nos hemos preocupado de estudiar la posible asociación entre el perfil de expresión de un gen y una sola covariable (o variable fenotípica). Nuestro mayor interés (aunque hemos considerado otros casos sobre todo con el método SAM) ha estado en el caso en que la covariable nos define dos poblaciones a comparar dos tipos de cancer; una cepa mutante frente a una salvaje.

En este tema nos ocupamos de considerar la relación entre las expresiones del gen y más de una covariable. Situaciones habituales (y que consideramos aquí) será cuando tengamos las muestras clasificadas según dos criterios y queramos considerar el efecto simultáneo sobre la expresión. Ejemplos concretos de esta situación son:

1. Consideramos dos cepas (salvaje y mutante) y las observamos en varios instantes temporales. Supongamos que las muestras que observamos en cada instante temporal son independientes. Tenemos un diseño cruzado para cada gen.
2. Supongamos que tomamos muestras de un mismo individuo en dos instantes temporales (antes y después de un tratamiento por ejemplo). Además los individuos los tenemos clasificados en dos grupos. Para cada individuo tenemos una muestra apareada.

Los diseños experimentales anteriores son habituales en la literatura y los estudiamos en este tema. Y para ello se recurre a la teoría de modelos lineales adaptada a este contexto particular. El paquete básico en este tema es [129, limma].<sup>149</sup>

### 9.2.1 El modelo limma

El material de esta sección fue publicado en [128]. Lo que se hace es ajustar un modelo lineal para cada fila de la matriz de expresión considerando que tenemos valores independientes para las distintas muestras.<sup>150 151</sup> Si trabajamos con datos de microarrays entonces supondremos que trabajamos con su logaritmo en base 2 ( $\log_2$ ). Las expresiones aleatorias correspondientes al gen  $i$ -ésimo las denotamos por

$$X_i = (X_{i1}, \dots, X_{in})'.$$

Los correspondientes valores observados los denotaremos utilizando minúsculas, es decir;

$$x_i = (x_{i1}, \dots, x_{in})'.$$

El vector  $X_i$  es el vector respuesta para el  $i$ -ésimo ajuste que vamos a realizar.<sup>152</sup> Denotamos, como es habitual, por  $E(X_i)$  el vector de medias del vector  $X_i$  que tiene en cada componente la media de la variable  $X_{ij}$ , es decir,  $EX_i = (EX_{i1}, \dots, EX_{in})'$ . Se asume:

1.

$$E(X_i) = A\alpha_i \quad (9.11)$$

siendo  $A$  una matriz de diseño de rango completo en columnas<sup>5</sup> y  $\alpha_i$  es un vector de coeficientes.

2. Suponemos que la matriz de covarianzas del vector aleatorio  $X_i$  que es proporcional (siendo desconocida la constante de proporcionalidad) a una matriz conocida, es decir, asumimos que

$$\text{var}(X_i) = \sigma_i^2 W_i, \quad (9.12)$$

donde  $W_i$  es una matriz definida no negativa **conocida**. El vector  $X_i$  puede tener datos faltantes y la matriz  $W_i$  puede tener elementos en la diagonal principal nulos.

Como es habitual en la teoría de modelos lineales<sup>153</sup> estamos interesados en ciertos contrastes sobre el vector de coeficientes  $\alpha_i$  que corresponden al diseño utilizado en la experiencia y, por lo tanto, evalúan problemas de interés desde un punto de vista biológico. Un contraste tiene la forma general

$$\beta_i = C'\alpha_i.$$

---

<sup>5</sup>Todas sus columnas linealmente independientes.

<sup>149</sup> La viñeta de este paquete es, en sí misma, un buen libro y, sin duda, la referencia a consultar.

<sup>150</sup> Y olvidando por tanto todo el proceso de corrección de fondo y normalización.

<sup>151</sup> En § 27 se incluye un repaso de la teoría de modelos lineales.

<sup>152</sup> Asumimos que se ha procesado corrigiendo fondo y normalizando y que los datos están en escala  $\log_2$ .

<sup>153</sup> § 27.

Estamos interesados en contrastar si  $\beta_{ij}$  son nulas para los distintos  $j$  y para cada gen  $i$ , es decir, en la hipótesis nula:  $H_0 : \beta_{ij} = 0$  vs  $H_0 : \beta_{ij} \neq 0$ .

Ajustamos un modelo lineal para cada gen y obtenemos los estimadores  $\hat{\alpha}_i$ , el estimador  $s_i^2$  de  $\sigma_i^2$  y la matriz de covarianzas estimada

$$\text{var}(\hat{\alpha}_i) = V_i s_i^2 \quad (9.13)$$

siendo  $V_i$  una matriz definida positiva que no depende de  $s_i^2$ . Los estimadores de los contrastes son

$$\hat{\beta}_i = C' \hat{\alpha}_i,$$

y de su matriz de covarianzas es

$$\text{var}(\hat{\beta}_i) = C' V_i C s_i^2.$$

No se asume necesariamente normalidad para  $X_i$  ni tampoco asumimos que el ajuste de los modelos lineales se hace mediante el procedimiento de los mínimos cuadrados. Pero **si** se asume que:

1.  $\hat{\beta}_i$  sí que tiene una distribución aproximadamente normal con vector de medias  $\beta_i$  y matriz de covarianzas  $C' V_i C s_i^2$ .
2. También asumimos que  $s_i^2$  siguen aproximadamente una distribución ji-cuadrado escalada.

La matriz  $V_i$  podría depender de  $\alpha_i$ . Si esta dependencia se produce entonces consideramos  $V_i$  evaluada en  $\hat{\alpha}_i$  y que la dependencia puede ser ignorada al menos en una aproximación de primer orden.

Si denotamos por  $v_{ij}$  el  $j$ -ésimo elemento de la diagonal de  $C' V_i C$  entonces todas las hipótesis indicadas previamente se traducen en que:

1.  $\hat{\beta}_{ij} | \beta_{ij}, \sigma_i^2 \sim N(\beta_{ij}, v_{ij} \sigma_i^2)$ .
2.  $s_i^2 | \sigma_i^2 \sim \frac{\sigma_i^2}{d_i} \chi_{d_i}^2$  siendo  $d_i$  los grados de libertad residuales del modelo ajustado para el  $i$ -ésimo gen.

Asumiendo estas hipótesis se tiene que el siguiente estadístico

$$t_{ij} = \frac{\hat{\beta}_{ij}}{s_i \sqrt{v_{ij}}}$$

aproximadamente tiene una distribución t de Student con  $d_i$  grados de libertad.

En lo que sigue se asumirá (y no tienen porqué ser cierto) que  $\hat{\beta}_i$  y  $s_i^2$  son independientes para los distintos genes.

Tenemos muchos test simultáneos. Se trata de modelizar el comportamiento entre genes. Y para ello se opta por modelizar mediante distribuciones de probabilidad sobre los parámetros.<sup>154</sup> En concreto sobre  $\sigma_i^2$  la siguiente distribución,

$$\frac{1}{\sigma_i^2} \sim \frac{1}{d_0 s_0^2} \chi_{d_0}^2.$$

Para un  $j$  dado, suponemos que  $\beta_{ij}$  es no nula con una probabilidad conocida

$$P(\beta_{ij} \neq 0) = p_j$$

<sup>154</sup> Tenemos un modelo bayesiano.

Notemos que  $p_j$  tiene el sentido de la proporción esperada de genes que se expresan diferencialmente. Para los coeficientes no nulos se asume

$$\beta_{ij} | \sigma_i^2, \beta_{ij} \neq 0 \sim N(0, v_{0j}\sigma_i^2).$$

Asumiendo el modelo jerárquico que acabamos de especificar se tiene que la media de la distribución a posteriori de  $1/\sigma_i^2$  condicionada a  $s_i^2$  viene dada por

$$E\left[\frac{1}{\sigma_i^2} | s_i^2\right] = \frac{1}{\tilde{s}_i^2},$$

con

$$\tilde{s}_i^2 = \frac{d_0 s_0^2 + d_i s_i^2}{d_0 + d_i}.$$

Podemos definir el estadístico t moderado<sup>155</sup> como

<sup>155</sup> Moderated t-statistic.

$$\tilde{t}_{ij} = \frac{\hat{\beta}_{ij}}{\tilde{s}_i^2 \sqrt{v_{ij}}}.$$

Se demuestra que los t-estadísticos moderados  $\tilde{t}_{ij}$  y las varianzas muestrales residuales  $s_i^2$  se distribuyen independientemente. Y de hecho, tendremos que bajo la hipótesis nula  $H_0 : \beta_{ij} = 0$ , el t-estadístico moderado  $\tilde{t}_{ij}$  sigue una distribución t de Student con  $d_i + d_0$  grados de libertad.<sup>156</sup> Los grados de libertad que estamos añadiendo  $d_0$  expresan la ganancia de información que obtenemos de utilizar todos los genes siempre asumiendo el modelo jerárquico. Los valores  $d_0$  y  $s_0$  que se suponen conocidos en lo previo serán estimados a partir de los datos.

<sup>156</sup> Ver sección 4 de [128].

Se consideran los odds a posteriori dados por

$$O_{ij} = \frac{P(\beta_{ij} \neq 0 | \tilde{t}_{ij}, s_i^2)}{P(\beta_{ij} = 0 | \tilde{t}_{ij}, s_i^2)}. \quad (9.14)$$

Estos odds prueban que tienen la siguiente expresión.

$$O_{ij} = \frac{p_j}{1 - p_j} \left( \frac{v_{ij}}{v_{ij} + v_{0j}} \right)^{1/2} \left( \frac{\tilde{t}_{ij}^2 + d_0 + d_i}{\tilde{t}_{ij}^2 \frac{v_{ij}}{v_{ij} + v_{0j}} + d_0 + d_i} \right)^{(1-d_0+d_i)/2} \quad (9.15)$$

Finalmente los autores proponen trabajar con los log-odds dados por

$$B_{ij} = \ln O_{ij}. \quad (9.16)$$

En el procedimiento que acabamos de comentar falta estimar los parámetros de las distribuciones a priori. Hemos de estimar  $d_0$ ,  $s_0$ ,  $v_{0j}$  (para todos los  $j$ ) y las proporciones  $p_j$ . En principio, son valores que el propio usuario debiera de especificar. Representan la información previa y por lo tanto es la información que a priori y sin usar nada de la información muestral hemos de especificar. Es claro que esto no parece muy factible. Los autores proponen lo que se conoce como un método empírico bayesiano y utilizan los propios datos para determinar estos valores.<sup>157</sup>

<sup>157</sup> Ver sección 6 de [128].

Utilizando los log-odds  $B$  podemos ordenar los genes de mayor a menor expresión diferencial. De hecho, si observamos la ecuación 9.15 y la definición 9.16 es fácil ver que, asumiendo que los  $d_i$  y los  $v_i$  no varían entre genes<sup>158</sup>, entonces los log-odds  $B$  son funciones monótonas de  $|\tilde{t}_{ij}|$ . En resumen, la ordenación de los genes que obtenemos con los log-odds o con los estadísticos  $|\tilde{t}_{ij}|$  es la misma.

<sup>158</sup> O varían muy poco.

Es claro que los log-odds tienen una interpretación sencilla y, por ello, son útiles. Sin embargo, para decidir qué genes son significativos para el contraste en que estemos interesados es mejor opción trabajar con los t-estadísticos moderados.<sup>159</sup>

<sup>159</sup> Cuya definición es una especie de mezcla inteligente de Estadística clásica y bayesiana.

<sup>160</sup> Cosa nada desdeñable.

Trabajar con los t-estadísticos moderados,  $\tilde{t}$ , en lugar de con los log-odds tiene otra ventaja. Los  $\tilde{t}$  dependen solamente de  $d_0$  y  $s_0^2$ . Sin embargo, los log-odds dependen además de  $v_{0j}$  y de los  $p_j$ . Obviamente queremos menos estimaciones previas de hiperparámetros.<sup>160</sup>

Lo más importante, sin duda, el t-estadístico moderado tiene la ventaja sobre el t-estadístico usual de que es menos probable que aparezcan valores muy grandes simplemente porque las varianzas muestrales infraestiman las varianzas poblaciones. Trabajamos con pequeñas muestras lo que hace que la estimación de la varianza sea poco precisa. Una infraestimación notable de esta varianza ocasiona que el estadístico sea muy grande aunque no tengamos una diferencia de medias grande. Este mismo problema se comenta en § 9.1. Allí se proponía añadir una constante (estimada a partir de los propios datos) a la desviación estándar de la diferencia de medias (el error estándar). ¿Esto es lo mismo? Es similar en espíritu pero no es lo mismo. Si los grados de libertad verifican  $d_o < +\infty$  y  $d_i > 0$  entonces los t-estadísticos moderados tienen la siguiente expresión

$$\tilde{t}_{ij} = \left( \frac{d_0 + d_i}{d_i} \right)^{1/2} \frac{\hat{\beta}_{ij}}{\sqrt{s_{*,i}^2 v_{ij}}},$$

siendo

$$s_{*,i}^2 = s_i^2 + \frac{d_0}{d_i} s_0^2.$$

Cada varianza muestral es incrementada un valor positivo en principio distinto para cada gen. La idea que vimos en § 9.1 consistía en considerar

$$t_{*,ij} = \frac{\hat{\beta}_{ij}}{(s_i + a)\sqrt{v_{ij}}}$$

donde  $a$  es un valor positivo determinado por un método ad-hoc. En este método lo que se incrementa es la desviación estándar y no la varianza. El t-estadístico moderado incrementa la varianza. Además el incremento puede ser distinto para cada gen.

¿Y si queremos contrastar más de un contraste al mismo tiempo? En lugar de t-estadísticos moderados tendremos F-estadísticos moderados. Se comprueba que los F-estadísticos moderados no son más que los F-estadísticos usuales en los cuales sustituimos las varianzas  $s_i^2$  por  $\tilde{s}_i^2$ . Además su distribución es  $F_i \sim F_{r,d_0+d_i}$ , siendo  $r$  el número de contrastes linealmente independientes. Vemos que los grados de libertad del denominador también se incrementan.

En las siguientes secciones vamos a considerar distintos diseños experimentales y los analizaremos con [129, limma].<sup>161</sup>

<sup>161</sup> Este paquete es una de los fundamentales en el proyecto Bioconductor y de los más antiguos.

## 9.2.2 Datos gse44456 con limma

Leemos los datos tamidata::gse44456.

```
data(gse44456, package="tamidata")
```

Vamos a considerar la variable que nos da la tasa de crecimiento celular.

```
pData(gse44456) [, "postmortenint"]

## [1] 16.75 27.00 29.00 24.00 24.00 24.00 17.00
## [8] 12.00 21.00 36.00 19.00 36.00 68.00 46.00
## [15] 23.00 48.00 58.50 62.00 30.00 21.00 19.50
## [22] 24.00 59.50 50.00 22.00 37.00 41.00 16.00
## [29] 18.00 16.00 9.00 11.00 20.00 56.00 15.00
## [36] 43.00 11.00 32.00 21.50
```

Cargamos Limma.

```
library(limma)
```

Determinamos la matriz de diseño.

```
design = model.matrix(~ pData(gse44456) [, "postmortenint"])
colnames(design) = c("intercept", "postmortenint")
```

Podemos ver que la matriz `design`.

```
head(design)

##   intercept postmortenint
## 1          1       16.75
## 2          1       27.00
## 3          1       29.00
## 4          1       24.00
## 5          1       24.00
## 6          1       24.00
```

Ajustamos los modelos lineales.

```
fit = lmFit(gse44456, design)
```

```
fit1 = eBayes(fit)
```

Si la constante por la que multiplicamos growthrate 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")

##           logFC    AveExpr        t
## 7898750 -0.011508519  6.747570 -6.443278
## 8170468  0.013115884  6.012464  5.689956
## 7969651 -0.010989325  7.051056 -5.687741
## 8111415  0.008178560  2.540914  5.065649
## 7895171  0.012638970  8.895633  4.980637
## 7959012  0.012611550  4.428257  4.935664
## 7972269 -0.009755820  4.360153 -4.919346
## 8152664 -0.015744665  4.187651 -4.870722
```

```

## 7896252 -0.008523726 11.280638 -4.834066
## 7921533 -0.013989202 8.399406 -4.826789
##          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
## 8111415 9.095965e-06 0.060272779 1.924620
## 7895171 1.195878e-05 0.060272779 1.654193
## 7959012 1.381685e-05 0.060272779 1.511583
## 7972269 1.455932e-05 0.060272779 1.459918
## 8152664 1.701344e-05 0.060272779 1.306232
## 7896252 1.912948e-05 0.060272779 1.190642
## 7921533 1.957948e-05 0.060272779 1.167723

```

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.

### 9.2.3 Un diseño cruzado con limma

Seguimos utilizando tamidata::gse44456.

```

pacman:::p_load("limma")
data(gse44456, package="tamidata")

```

Observamos los datos fenotípicos o metadatos.

```

head(pData(gse44456))

##                                     case gender age
## GSM1085665_HE32H001.CEL control male  68
## GSM1085666_HE32H003.CEL alcoholic male  51
## GSM1085667_HE32H004.CEL control male  50
## GSM1085668_HE32H005.CEL alcoholic male  50
## GSM1085669_HE32H006_2_.CEL control male  56
## GSM1085670_HE32H007_2_.CEL alcoholic male  59
##                                         cirrhosis smoker
## GSM1085665_HE32H001.CEL           No      No
## GSM1085666_HE32H003.CEL           No      Yes
## GSM1085667_HE32H004.CEL           No      No
## GSM1085668_HE32H005.CEL           Yes     Yes
## GSM1085669_HE32H006_2_.CEL       No      Yes
## GSM1085670_HE32H007_2_.CEL       No      No
##                                         postmortenint pH
## GSM1085665_HE32H001.CEL        16.75 6.59
## GSM1085666_HE32H003.CEL        27.00 5.58
## GSM1085667_HE32H004.CEL        29.00 6.68
## GSM1085668_HE32H005.CEL        24.00 6.59
## GSM1085669_HE32H006_2_.CEL     24.00 6.53
## GSM1085670_HE32H007_2_.CEL     24.00 6.57
##                                         batch
## GSM1085665_HE32H001.CEL      Batch 1
## GSM1085666_HE32H003.CEL      Batch 1
## GSM1085667_HE32H004.CEL      Batch 2

```

```
## GSM1085668_HE32H005.CEL      Batch 2
## GSM1085669_HE32H006_2_.CEL Batch 1
## GSM1085670_HE32H007_2_.CEL Batch 1
```

**Analizamos la posible dependencia** con el alcoholismo recogida en la covariable `case`

```
alcoholism = pData(gse44456)[,"case"] # Simplificamos el código
design = model.matrix(~ 0 + alcoholism)
colnames(design) = c("control","alcoholic")
```

Ajustamos el modelo.

```
fit = lmFit(gse44456,design)
```

```
(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")

##          logFC AveExpr      t
## 7927186 -0.5424921 7.826424 -5.914411
## 8125919 -1.1358866 8.313619 -5.628792
## 8021081 -1.2885800 8.593822 -5.481851
## 7961595  0.5322101 4.180459  5.440839
## 7995838 -0.9825655 8.540374 -5.096932
## 8130867  0.5858452 7.566787  5.083113
## 8114814  0.3685955 7.694306  5.003841
## 7922889  0.5172199 8.438433  4.946850
## 8021741  0.7651623 9.183204  4.720399
## 8099476  0.5231415 5.584413  4.658028
##          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
## 7961595 2.687326e-06 0.02236997 3.831026
## 7995838 8.199033e-06 0.04757580 2.959048
## 8130867 8.572989e-06 0.04757580 2.924055
## 8114814 1.106824e-05 0.05264847 2.723446
## 7922889 1.329382e-05 0.05533056 2.579390
## 8021741 2.741801e-05 0.10143750 2.008880
## 8099476 3.342454e-05 0.10201553 1.852423
```

**En el segunda análisis nos fijamos** en dos covariables, el alcoholismo y el sexo de la persona. Es un diseño factorial  $2 \times 2$ . 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?

En jerga estadística hablaríamos de los efectos principales de los factores y de la posible interacción.

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]] = paste(pData(gse44456) [, "case"] [i],
                         pData(gse44456) [, "gender"] [i], sep="")
casegender = factor(unlist(casegender))
```

Consideremos la siguiente matriz de diseño.

```
design = model.matrix(~ 0 + casegender)
colnames(design) = levels(casegender)
```

Podemos ver que cada coeficiente corresponde con la expresión media para la correspondiente combinación de factores.

```
fit = 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 = 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")
```

### 9.3 Un comentario técnico importante

En este tema hemos trabajado con p-valores que están relacionados entre sí. Si denotamos por  $P_i$  el valor aleatorio de el p-valor correspondiente al contraste de  $H_i$  frente a  $K_i$  entonces su distribución (si asumimos distribución continua para el estadístico como en el test de la t de Student), bajo la hipótesis nula  $H_i$ , será uniforme en el intervalo unitario  $[0, 1]$ , es decir,  $P_i \sim U(0, 1)$ . Hay procedimientos de corrección de estos p-valores que consideran que los  $P_i$  son independientes. A priori sabemos que esto no es cierto. Los genes tienen interdependencias que se traducen en dependencias entre las distribuciones de los  $P_i$ . Sin embargo, hay otro efecto también muy destacable y no tan evidente a priori. Aquellos genes que tienen una expresión mayor son detectados como diferencialmente expresados con más probabilidad. La potencia del test es mayor.



# Capítulo 10

## Expresión diferencial con datos RNASeq

### 10.1 Introducción

Tenemos datos de expresión de gen obtenidos mediante la técnica conocida como RNA-seq. Pero, ¿qué tenemos realmente? ¿Con qué vamos a trabajar?

Ya hemos realizado todo el preprocesado. Ya hemos alineado y contado lecturas a nivel de gen (o si podemos isoforma). Tenemos pues una matriz de expresión **observada**  $x = [x_{ij}]_{i=1,\dots,N; j=1,\dots,n}$  donde hemos considerado  $N$  genes y tenemos  $n$  muestras. Además tendremos covariables, variables fenotípicas o metadatos.<sup>162</sup> 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  $y = [y_{jk}]_{j=1,\dots,n; k=1,\dots,p}$ . Habitualmente  $p = 1$  y  $y = (y_1, \dots, y_n)'$  es simplemente un vector.

<sup>162</sup> Un estadístico dice covariable, un bioinformático dice variable fenotípica y un moderno de los de los Big Data diría cualquier cosa, por qué no, metadatos.

### 10.2 Una muestra por condición

¿Y si no tenemos réplicas en cada condición que queremos comparar? Queremos comparar dos condiciones y disponemos de una muestra en cada una de las condiciones. ¿Qué podemos hacer para comparar los conteos para cada gen?<sup>163</sup> En esta (mala) situación la información que tenemos para cada gen la podemos recoger en una tabla de contingencia  $2 \times 2$  (10.1). En esta tabla  $y_j$  recoge el número de lecturas alineadas sobre el gen que estamos estudiando en la muestra  $j$  (con  $j = 1, 2$ ). Los tamaños de las librerías en cada muestra son  $m_j$  (con  $j = 1, 2$ ).

Vamos a considerar dado el total de conteos asociados al gen  $i$ -ésimo, esto es, consideraremos constante el conteo  $x_{i1} + x_{i2}$ .<sup>164</sup> El gen

<sup>163</sup> Lo primero sería pensar en tomar alguna muestra más por condición que los milagros solamente son los jueves. [http://es.wikipedia.org/wiki/Los\\_jueves,\\_milagro](http://es.wikipedia.org/wiki/Los_jueves,_milagro) y <https://www.youtube.com/watch?v=-yDNya7XP-4>.

Tabla 10.1: Una muestra por condición en donde la primera sería caso y la segunda control.

	Muestra 1 (Caso)	Muestra 2 (Control)	Total
Gen i	$x_{i1}$	$x_{i2}$	$x_{i\cdot} = x_{i1} + x_{i2}$
Resto	$m_1 - x_{i1}$	$m_2 - x_{i2}$	$m_1 + m_2 - x_{i\cdot}$
Total	$m_1$	$m_2$	$m_1 + m_2$

que consideramos tiene un conteo aleatorio  $X_{i1}$  en la primera muestra (por ejemplo, correspondiente a caso). Su distribución de probabilidad será binomial con  $x_{i1} + x_{i2}$  pruebas y una probabilidad de éxito (lectura procedente de la primera muestra) desconocida  $p$ . Si no hay diferencias entre las muestras el valor de  $p$  debiera de coincidir con  $m_1/(m_1 + m_2)$ , es decir, debe coincidir la proporción con la proporción que la primera muestra representa en el total de las dos muestras. Por tanto, evaluar la expresión diferencial del gen se traduce en un contraste sobre  $p$  la probabilidad de que la lectura corresponda al caso. Una primera referencia en este sentido es [80].

En edgeR::binomTest utilizan como p-valor la suma de las probabilidades de los conteos que tienen una probabilidad menor o igual al observado bajo la hipótesis nula.

Consideremos los datos tamidata::PRJNA297664. Nos quedaremos con una muestra “wild” y otra “SEC66 deletion”. Compararemos entre sí estas dos muestras.

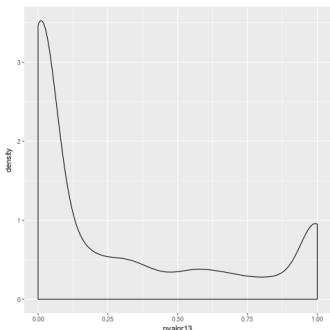
```
pacman::p_load("SummarizedExperiment")
data(PRJNA297664, package="tamidata")
se = PRJNA297664
rm(PRJNA297664)
```

La variable que indica el tipo es **treatment**.

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

## [1] Wild           Wild           SEC66 deletion
## [4] Wild           SEC66 deletion SEC66 deletion
## Levels: Wild SEC66 deletion
```

Vamos a comparar las muestras 1 y 3 con la función edgeR::binomTest.



```
pacman::p_load("edgeR")
pvalor13 = edgeR::binomTest(assay(se) [, 1], assay(se) [, 3])
```

Podemos hacernos una idea de cómo son estos p-valores con una estimación de su función de densidad con (figura ??).

También podemos aplicar para el mismo problema el test de Fisher. En cualquier caso, **solo tiene sentido hacer este análisis cuando tenemos una muestra en cada condición**.<sup>165</sup>

¿Qué hacemos con más de una muestra? ¿Realizar todas las comparaciones dos a dos? No. Obviamente hay que buscar procedimientos más adecuados.

¿Se puede realizar algo similar cuando tenemos varias muestras por cada una de las dos condiciones? Es tentador simplemente sumar los conteos de cada gen en cada condición. Tendríamos el total de lecturas por gen y condición. Sin embargo, si consideramos la tabla 10.1 correspondiente estaríamos ignorando la variabilidad intra condición de los conteos correspondientes. En este caso hemos de acudir a alguno de los procedimientos que vemos en las siguientes secciones.

**De porqué una muestra por condición no es buena consejera.** Vamos a evaluar de un modo empírico porqué necesitamos más muestras por condición. Porqué esa práctica (más frecuente de lo que se debiera) de

<sup>165</sup> Y tampoco mucho.

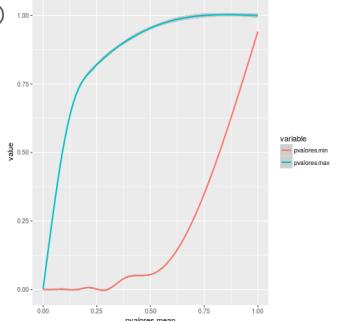
tener una muestra por condición no se debe de utilizar. Siguiendo con PRJNA297664 tenemos 9 posibles pares formados por una muestra por condición. Vamos a calcular los p-valores para cada una de estas nueve posibles comparaciones y ver cómo varían estos p-valores.

```
pares = rbind(c(1,3),c(1,5),c(1,6),c(2,3),c(2,5),c(2,6),c(4,3),
               c(4,5),c(4,6))
aux = function(rr) binomTest(assay(se)[,rr[1]], assay(se)[,rr[2]])
pvalores = apply(pares, 1, aux)
```

Por cada gen tenemos el p-valor resultado de comparar cada uno de los pares. Tenemos pues 9 p-valores por gen. Calculamos el p-valor medio, el mínimo y el máximo y los ordenamos de menor a mayor. Finalmente representamos en abscisas el p-valor medio y en ordenadas dos líneas correspondientes al mínimo y al máximo. Dada la gran cantidad de puntos hemos de representar una versión suavizada del mínimo y máximo.

```
png("figures/RNASeqDE10.png")
pvalores.mean = apply(pvalores, 1, mean) ## Media de p-valores
pvalores.min = apply(pvalores, 1, min) ## Mínimo de p-valores
pvalores.max = apply(pvalores, 1, max) ## Máximo de p-valores
df = data.frame(pvalores.mean, pvalores.min, pvalores.max)
df = df[sort(pvalores.mean, index.return=TRUE)$ix,] ## Ordenamos
df1 = reshape2::melt(df, id="pvalores.mean")
pp = ggplot(df1, aes(x=pvalores.mean, y=value, color=variable))
pp + geom_smooth() # Suavizamos mínimo y máximo
dev.off()
```

En la figura ?? tenemos el resultado. Podemos ver la tremenda variabilidad que tenemos.



## 10.3 edgeR

Este paquete implementa los procedimientos propuestos en [121, 120]<sup>166</sup>

### 10.3.1 Estimación de una dispersión común

Consideraremos una característica en  $n$  muestras (o librerías) de tamaños distintos.<sup>167</sup> Sea  $m_i$  el tamaño de la  $i$ -ésima librería (total de lecturas). Sea  $\lambda$  la proporción que hay en una librería cualquiera de la característica en que estamos interesados. Vamos a asumir que  $Y_i$  tiene una distribución binomial negativa con media  $\mu_i = m_i\lambda$  y con dispersión  $\phi$ .  $Y$  tiene una distribución binomial negativa con media  $\mu$  y dispersión  $\phi$ ,  $Y \sim BN(\mu, \phi)$ , 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  $y = 0, 1, \dots$ . Su media es  $E(Y) = \mu$  y su varianza  $var(Y) = \mu + \phi\mu^2$ . Ver § 21.8.2.

Supongamos el caso ideal (e irreal) en que todas las librerías tienen el mismo tamaño:  $m_i = m$  para  $i = 1, \dots, n$ . Bajo esta condición tenemos que  $Z = \sum_{i=1}^n Y_i \sim NB(nm\lambda, \phi/n)$ . Además la logverosimilitud condicionada al valor de  $Z = z$  no depende de  $\lambda$  y podemos estimar

<sup>166</sup> Hay que leer primero [121] y luego [120] pues es la secuencia temporal de los trabajos aunque se publicaron en orden invertido.

<sup>167</sup> Son réplicas de una misma experimentación con tamaños distintos.

el valor de  $\phi$ . Por tanto, se estimaría  $\phi$  maximizando la verosimilitud condicionada dada por

$$l_{y|z} = \left[ \sum_{i=1}^n \log \Gamma(y_i + 1/\phi) \right] + \log \Gamma(n/\phi) - \log \Gamma(z + n/\phi) - n \log \Gamma(1/\phi), \quad (10.1)$$

donde  $\mathbf{y} = (y_1, \dots, y_n)'$  son los conteos observados. Si los tamaños de las librerías son distintos la verosimilitud no tiene una expresión simple. La idea en [121] 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,  $m^* = \sqrt[n]{\prod_{i=1}^n m_i}$ . ¿Cómo transformamos los conteos  $y_i$ ? El procedimiento propuesto es:

1. Inicializamos  $\phi$  (por ejemplo, con el estimador máximo verosímil condicionado sin realizar ningún ajuste).
2. Dado el valor estimado de  $\phi$ , estimamos  $\lambda$  maximizando la verosimilitud para el valor estimado de la dispersión.
3. Suponemos que cada conteo  $y_i$  es un valor observado de una distribución binomial negativa con media  $m_i\lambda$  y parámetro de dispersión  $\phi$ . Calculamos los percentiles

$$p_i = P(Y < y_i | m_i\lambda, \phi) + \frac{1}{2}P(Y = y_i | m_i\lambda, \phi), \quad (10.2)$$

para  $i = 1, \dots, n$ .

4. Suponemos ahora una distribución binomial negativa con media  $m^*\lambda$  y dispersión  $\phi$ . Determinamos qué valor sería el percentil de orden  $p_i$  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  $\phi$  con los pseudodatos utilizando la verosimilitud condicionada a la (pseudo) suma total de un gen.
6. Repetimos desde 2 hasta 5 hasta que converja  $\phi$ .

A este procedimiento de estimación de  $\phi$  y  $\lambda$  los autores lo denominan *máxima verosimilitud condicionada ajustada por cuantiles*.<sup>168</sup> Lo fundamental es entender que estamos modificando los datos para conseguir un tamaño común de las librerías, una misma profundidad de secuenciación.

### 10.3.2 Un test exacto

Tenemos dos condiciones a comparar.

**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  $k = 1, \dots, n_j$ . El tamaño de la librería de la condición  $j$  en el gen  $i$  es  $m_{ij}$ . Supongamos que aplicamos el método qCML dentro de cada condición

<sup>168</sup> Quantile-adjusted conditional maximum likelihood (qCML).

j de modo que por condición tenemos un tamaño de librería común. Con los pseudodatos (que denotamos también  $y_{ijk}$ ) 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  $z_{ij} = y_{ij} = \sum_{k=1}^{n_j} y_{ijk}$  para la dispersión  $\phi$  dada por

$$l_i(\phi) = \sum_{j=1}^2 \left( \sum_{k=1}^{n_j} \log \Gamma(y_{ijk} + \phi^{-1}) + \log \Gamma(n_j \phi^{-1}) - \log \Gamma(z_{ij} + n_j \phi^{-1}) - n_j \log \Gamma(\phi^{-1}) \right). \quad (10.3)$$

En consecuencia la logverosimilitud común es<sup>169</sup>

$$l(\phi) = \sum_{i=1}^N l_i(\phi). \quad (10.4)$$

<sup>169</sup> Sería más correcto hablar de pseudoverosimilitud ya que los distintos genes son dependientes.

La estimación de  $\phi$  en § 10.3.1 se obtiene maximizando la función dada en la ecuación 10.4. 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

$$EY_{ijk} = m_{ij}\lambda_{ij},$$

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  $\lambda$ , es decir, el contraste siguiente:

$$\begin{aligned} H_i : & \lambda_{i1} = \lambda_{i2}, \\ K_i : & \lambda_{i1} \neq \lambda_{i2}. \end{aligned}$$

**Bajo la hipótesis nula** el valor  $\lambda$  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  $m^*$ .<sup>170</sup> Estos nuevos pseudodatos 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  $k = 1, \dots, n_j$ . 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

$$EY_{ijk} = m^* \lambda_{ij},$$

para cualquier  $i, j, k$ . Bajo la hipótesis nula  $H_0$  no tendríamos diferencia en el valor de  $\lambda$  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 pseudodatos (que obtienen un tamaño de librería común)  $y_{ijk}$  con  $k = 1, \dots, n_j$  podemos estimar  $\lambda_i$ . 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

<sup>170</sup> Importante, los pseudodatos no son los mismos utilizados para estimar la dispersión común.

$Y_{i2}$  son independientes. La suma  $Y_{i1} + Y_{i2}$ , también tiene una distribución binomial negativa:  $Y_{i1} + Y_{i2} = \sum_{i=1}^2 \sum_{k=1}^{n_i} Y_{ijk} \sim NB((n_1 + n_2)m^* \hat{\lambda}_i, \hat{\phi}_C/(n_1 + n_2))$ . Podemos considerar la distribución condicionada del vector aleatorio  $(Y_{i1}, Y_{i2})$  a la suma  $Y_{i1} + Y_{i2}$ , 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.<sup>171</sup>

<sup>171</sup> Es un test similar al test de Fisher bilateral en donde las probabilidades hipergeométricas son sustituidas por probabilidades por la distribución binomial negativa. Ver [3, Página 93].

<sup>172</sup> Buena desde el punto de vista del modelo y su estimación pero nos aleja de los datos.

<sup>173</sup> No un criterio de optimidad. Simplemente argumentan lo razonable de la elección.

### 10.3.3 Dispersiones posiblemente distintas

En el test exacto que hemos visto en § 10.3.2 hemos asumido una dispersión común  $\phi$  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.<sup>172</sup> En [120] proponen no asumir un mismo valor para cada gen y considerar que las dispersiones dependen del gen. Tenemos ahora, en lugar del valor común  $\phi$ , un valor (posiblemente) distinto para el gen  $i$ ,  $\phi_i$ . ¿Y cómo lo estimamos? Proponen un compromiso entre la contribución que hacen los conteos del  $i$ -ésimo gen,  $l_i$  (ecuación 10.3), a la logverosimilitud global,  $l$  (ecuación 10.4), y la propia logverosimilitud global. En concreto hablan de la logverosimilitud condicionada ponderada definida como

$$WL(\phi_i) = l_i(\phi_i) + \alpha l(\phi_i), \quad (10.5)$$

siendo  $\alpha$  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  $\alpha$ . Un mayor  $\alpha$  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 [120, Sección, 3.2, pág. 2883] dan una motivación<sup>173</sup>

Para explicar cómo lo hacen hemos de considerar la **función score** definida como

$$S_i(\phi) = \frac{\partial l_i(\phi)}{\partial \phi} \quad (10.6)$$

y la información esperada definida como

$$I_i(\phi) = E(J_i), \text{ siendo } J_i = -\frac{\partial^2 l_i(\phi)}{\partial \phi^2}. \quad (10.7)$$

El procedimiento de estimación de  $\alpha$  y la posterior estimación de  $\phi_i$  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  $\tau_0$  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  $\alpha$  en el paso anterior, maximizamos la función  $WL(\phi_i)$  definida en la ecuación 10.5.

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  $\delta = \phi/(\phi + 1)$ .<sup>174</sup>

<sup>174</sup> Lo cual es irrelevante. Estimamos  $\delta$  estimamos  $\phi$  pues hay una correspondencia 1-1.

### 10.3.4 edgeR: PRJNA297664

En esta sección analizamos la posible expresión diferencial de los datos tamidata::PRJNA297664.

```
data(PRJNA297664, package="tamidata")
```

```
pacman::p_load("edgeR", "SummarizedExperiment")
dge = DGEList(counts=assay(PRJNA297664),
  group=colData(PRJNA297664)[, "treatment"])
dge.c = estimateCommonDisp(dge) ##Estimamos dispersión común
dge.c$common.dispersion

## [1] 0.01170892

dge.t = estimateTagwiseDisp(dge.c) ##Dispersiones por gen
et.c = exactTest(dge.c)
et.t = exactTest(dge.t)
```

Determinamos los genes significativos.

```
topTags(et.c)

## Comparison of groups: SEC66 deletion-Wild
##                logFC    logCPM      PValue
## YBR171W    -10.150220 6.114922 2.128721e-258
## YCR021C     -1.928779 8.463733 4.873506e-47
## YBR054W     -1.878579 7.156683 3.518798e-43
## YGL255W     -1.846348 7.518466 1.930761e-42
## YNR034W-A   -2.176676 4.670571 1.175800e-40
## YBR093C     -1.694281 8.563323 2.731791e-37
## YFR053C      1.621598 6.384697 2.625927e-31
## YER150W     -1.560027 5.479642 2.325258e-26
## YDR171W     -1.383170 7.792428 2.142035e-25
## YDR214W      1.362108 7.982939 1.024220e-24
##                      FDR
## YBR171W    1.516926e-254
## YCR021C    1.736430e-43
## YBR054W    8.358319e-40
```

```

## YGL255W    3.439650e-39
## YNR034W-A  1.675750e-37
## YBR093C    3.244457e-34
## YFR053C    2.673194e-28
## YER150W    2.071224e-23
## YDR171W    1.696016e-22
## YDR214W    7.298590e-22

topTags(et.t)

## Comparison of groups: SEC66 deletion-Wild
##                  logFC      logCPM      PValue
## YBR171W    -10.150503  6.114922 5.636512e-296
## YGL255W    -1.846051  7.518466 5.459134e-30
## YBR093C    -1.694086  8.563323 5.843701e-27
## YNR034W-A  -2.174452  4.670571 7.411466e-22
## YDR214W    1.362104  7.982939 2.334672e-20
## YLR109W    1.240894  9.510958 4.583075e-20
## YHR215W    -1.208860  9.533270 2.172522e-18
## YAR071W    -1.211709  9.219219 3.764430e-18
## YMR186W    1.109256 11.622933 9.483967e-18
## YKL161C    1.100257  5.723199 1.070358e-16
##                  FDR
## YBR171W    4.016579e-292
## YGL255W    1.945090e-26
## YBR093C    1.388074e-23
## YNR034W-A  1.320353e-18
## YDR214W    3.327375e-17
## YLR109W    5.443166e-17
## YHR215W    2.211627e-15
## YAR071W    3.353166e-15
## YMR186W    7.509195e-15
## YKL161C    7.627369e-14

```

## 10.4 Un método de cuasi-verosimilitud

En esta sección tratamos el método propuesto en [92] e implementado en [**R-QuasiSeq**].

Utilizamos en lo que sigue la notación de [92].

## 10.5 SAMseq

Este método fue propuesto en [86] y está implementado en el paquete [137, samr]. Supongamos el caso con dos grupos. Denotamos como siempre por  $x_{ij}$  el conteo de la  $i$ -ésima característica en la  $j$ -ésima muestra ( $i = 1, \dots, N$  y  $j = 1, \dots, n$ ). El tamaño de la librería o profundidad de secuenciación será  $m_j = \sum_{i=1}^N x_{ij}$ . Como siempre el valor observado es  $x_{ij}$  mientras que el valor aleatorio (antes de observar los datos) lo denotamos con  $X_{ij}$ . Los índices correspondientes a los dos grupos de muestras los denotamos por  $C_1$  y  $C_2$ . Estos conjuntos de índices constituyen una partición de  $\{1, \dots, n\}$ . Suponemos que las clases  $C_1$  y  $C_2$  la componen  $n_1$  y  $n_2$  elementos (con  $n_1 + n_2 = n$ ).

Supongamos, en un primer momento, que todas las profundidades de secuenciación son iguales  $m_1 = \dots = m_n = m$ . Fijamos la característica  $i$ . Ordenamos (de menor a mayor) los conteos  $x_{i1}, \dots, x_{in}$ . La posición de  $x_{ij}$  en  $x_{i*} = (x_{i1}, \dots, x_{in})$  la denotamos por  $r_{ij}(x_{i*})$ . Si consideramos el valor aleatorio  $X_{i*}$  entonces tenemos la variable aleatoria  $R_{ij}(X_{i*})$ . El estadístico de Mann-Whitney-Wilcoxon<sup>175</sup> sería

$$T_i = \sum_{j \in C_1} R_{ij}(X_{i*}) - \frac{n_1(n+1)}{2}. \quad (10.8)$$

Se supone que no tenemos empates en  $X_{i*}$ . Si  $T_i$  es muy grande significa que en la característica  $i$  los valores en la primera clase ( $C_1$ ) tienden a ser mayores que en la segunda ( $C_2$ ). Un valor muy pequeño indica lo contrario. Notemos que  $T_i$  puede tomar valores negativos. En general, si el valor absoluto  $|T_i|$  es muy grande, lo que tenemos es que ambos grupos de valores son claramente distintos, uno mayor que el otro. Valores próximos a cero indican que no hay diferencias entre los grupos. Hasta aquí no hay problema ninguno. Simplemente se propone utilizar un test de Mann-Whitney-Wilcoxon para comparar los órdenes en los dos grupos que comparamos. El problema aparece cuando los tamaños de las librerías son distintas que es lo habitual. De hecho, suelen ser muy distintos. Una primera idea que no da buenos resultados es dividir los conteos originales por los correspondientes tamaños de las librerías.

Suponemos fijos los tamaños de las librerías. Esto es, aunque suponemos conteos aleatorios la suma de las correspondientes columnas de conteos las asumimos como fijas.<sup>176</sup> Denotamos el mínimo tamaño de las librerías con  $m_0 = \min_{j=1, \dots, n} m_j$  y la librería donde se da este mínimo suponemos que es la  $j_0$ . Supongamos que, para la librería  $j$ , del total de lecturas elegimos al azar  $m_0$  (correspondiente a la librería con menor tamaño). Esto supone que cada lectura original de esta librería es conservada con probabilidad  $\frac{m_0}{m_j}$  y eliminada con probabilidad  $1 - \frac{m_0}{m_j}$ . Si denotamos por  $X'_{ij}$  el número de lecturas que nos quedan después de este proceso de muestreo tendremos que

$$X'_{ij}|X_{ij} \sim \text{Binomial}(X_{ij}, \frac{m_0}{m_j}).$$

Los autores llaman a esto un **muestreo hacia abajo**.<sup>177</sup> Este procedimiento de muestreo nos produce unos nuevos conteos y un nuevo valor del estadístico dado en 10.14 que podemos denotar por  $T'_i$ . Notemos que si el tamaño mínimo es mucho menor que el resto de tamaños estamos descartando muchas lecturas.

Una segunda opción es igualar los tamaños de las librerías a un valor intermedio como la media geométrica  $m^* = \left( \prod_{j=1}^n m_j \right)^{1/n}$ . Generamos conteos aleatorios con

$$X'_{ij}|X_{ij} \sim \text{Poisson}\left(\frac{m^*}{m_j} X_{ij}\right). \quad (10.9)$$

Sería el muestreo Poisson. Para evitar empates entre los valores generados se sustituyen estos valores por

$$X'_{ij} + \epsilon_{ij}$$

siendo  $\epsilon_{ij}$  variables aleatorias independientes y con distribución común uniforme en el intervalo  $[0, 0.1]$ .<sup>178</sup> Como vemos estamos generando va-

<sup>175</sup> A veces se le llama de Mann-Whitney, otras de Wilcoxon y otras con los tres nombres.

<sup>176</sup> Una hipótesis necesaria.

<sup>177</sup> Down sampling.

<sup>178</sup> Un pequeño truco para evitar empates.

lores por lo que lo que obtenemos también es aleatorio. Supongamos que en un muestreo obtenemos  $T'_i(b)$ . Repetimos este muestreo  $B$  veces. Consideramos el estadístico

$$T_i^* = \frac{1}{B} \sum_{b=1}^B T'_i(b). \quad (10.10)$$

Este es el estadístico con el que finalmente se trabaja para contrastar. No conocemos la distribución de este estadístico. Hemos indicado que la región crítica natural, la zona en donde rechazamos la hipótesis nula de que no hay diferencia significativa entre muestras vendría dada por  $T_i^* > c$  siendo  $c$  un valor a determinar. Dado un valor positivo  $c$  consideraremos como significativa a la característica  $i$  si  $T_i^* > c$ . Lo que vamos a hacer es elegir un conjunto de posibles valores para  $c$  y **estimaremos** la tasa de hipótesis nulas falsamente rechazadas si utilizamos ese valor de  $c$ . El procedimiento<sup>179</sup> es el siguiente:

1. Calculamos  $T_1^*, \dots, T_N^*$  a partir de nuestros datos.
2. Permutamos  $S$  veces las columnas de  $X$  manteniendo la asignación de los grupos (las etiquetas de clase) y calculamos  $T_1^*(s), \dots, T_N^*(s)$  utilizando los datos permutados con  $s = 1, \dots, S$ .
3. Tomamos una serie de valores para  $c$  y calculamos:

$$\hat{V} = \frac{1}{S} \sum_{i=1}^N \sum_{s=1}^S 1_{|T_i^*(s)| > c} \quad (10.11)$$

que estima la cantidad de hipótesis nulas falsamente rechazadas y

$$\hat{R} = \sum_{i=1}^N 1_{|T_i^*| > c} \quad (10.12)$$

que estima la cantidad de hipótesis nulas rechazadas.

4. El valor de FDR asociado a cada valor  $c$  es estimado con

$$\widehat{FDR}(c) = \frac{\hat{\pi}_0 \hat{V}}{\hat{R}}. \quad (10.13)$$

<sup>179</sup> O proporción de características que no son significativamente distintas entre los dos grupos que comparamos.

donde  $\pi_0$  es la proporción de hipótesis nulas ciertas.<sup>180</sup> El estimador de  $\pi_0$  utilizado es

$$\hat{\pi}_0 = \frac{2 \sum_{i=1}^N 1_{|T_i^*| \leq q}}{N},$$

siendo  $q$  la mediana de  $\{|T_i^*(s)| : i = 1, \dots, N; s = 1, \dots, S\}$ .

### 10.5.1 SAMseq: PRJNA297664

```
topTags(et.c)
```

```
## Comparison of groups: SEC66 deletion-Wild
##              logFC    logCPM      PValue
## YBR171W    -10.150220 6.114922 2.128721e-258
```

```

## YCR021C    -1.928779 8.463733 4.873506e-47
## YBR054W    -1.878579 7.156683 3.518798e-43
## YGL255W    -1.846348 7.518466 1.930761e-42
## YNR034W-A   -2.176676 4.670571 1.175800e-40
## YBR093C    -1.694281 8.563323 2.731791e-37
## YFR053C     1.621598 6.384697 2.625927e-31
## YER150W    -1.560027 5.479642 2.325258e-26
## YDR171W    -1.383170 7.792428 2.142035e-25
## YDR214W     1.362108 7.982939 1.024220e-24
##
## FDR
## YBR171W    1.516926e-254
## YCR021C    1.736430e-43
## YBR054W    8.358319e-40
## YGL255W    3.439650e-39
## YNR034W-A   1.675750e-37
## YBR093C    3.244457e-34
## YFR053C    2.673194e-28
## YER150W    2.071224e-23
## YDR171W    1.696016e-22
## YDR214W    7.298590e-22

topTags(et.t)

## Comparison of groups: SEC66 deletion-Wild
##                  logFC      logCPM      PValue
## YBR171W    -10.150503  6.114922 5.636512e-296
## YGL255W    -1.846051  7.518466 5.459134e-30
## YBR093C    -1.694086  8.563323 5.843701e-27
## YNR034W-A   -2.174452  4.670571 7.411466e-22
## YDR214W     1.362104  7.982939 2.334672e-20
## YLR109W     1.240894  9.510958 4.583075e-20
## YHR215W    -1.208860  9.533270 2.172522e-18
## YAR071W    -1.211709  9.219219 3.764430e-18
## YMR186W     1.109256 11.622933 9.483967e-18
## YKL161C     1.100257  5.723199 1.070358e-16
##
## FDR
## YBR171W    4.016579e-292
## YGL255W    1.945090e-26
## YBR093C    1.388074e-23
## YNR034W-A   1.320353e-18
## YDR214W    3.327375e-17
## YLR109W    5.443166e-17
## YHR215W    2.211627e-15
## YAR071W    3.353166e-15
## YMR186W    7.509195e-15
## YKL161C    7.627369e-14

```

## 10.6 Un método de cuasi-verosimilitud

En esta sección tratamos el método propuesto en [92] e implementado en [**R-QuasiSeq**].

Utilizamos en lo que sigue la notación de [92].

## 10.7 SAMseq

Este método fue propuesto en [86] y está implementado en el paquete [137, samr]. Supongamos el caso con dos grupos. Denotamos como siempre por  $x_{ij}$  el conteo de la  $i$ -ésima característica en la  $j$ -ésima muestra ( $i = 1, \dots, N$  y  $j = 1, \dots, n$ ). El tamaño de la librería o profundidad de secuenciación será  $m_j = \sum_{j=1}^N x_{ij}$ . Como siempre el valor observado es  $x_{ij}$  mientras que el valor aleatorio (antes de observar los datos) lo denotamos con  $X_{ij}$ . Los índices correspondientes a los dos grupos de muestras los denotamos por  $C_1$  y  $C_2$ . Estos conjuntos de índices constituyen una partición de  $\{1, \dots, n\}$ . Suponemos que las clases  $C_1$  y  $C_2$  la componen  $n_1$  y  $n_2$  elementos (con  $n_1 + n_2 = n$ ).

Supongamos, en un primer momento, que todas las profundidades de secuenciación son iguales  $m_1 = \dots = m_n = m$ . Fijamos la característica  $i$ . Ordenamos (de menor a mayor) los conteos  $x_{i1}, \dots, x_{in}$ . La posición de  $x_{ij}$  en  $x_{i*} = (x_{i1}, \dots, x_{in})$  la denotamos por  $r_{ij}(x_{i*})$ . Si consideramos el valor aleatorio  $X_{i*}$  entonces tenemos la variable aleatoria  $R_{ij}(X_{i*})$ . El estadístico de Mann-Whitney-Wilcoxon<sup>181</sup> sería

$$T_i = \sum_{j \in C_1} R_{ij}(X_{i*}) - \frac{n_1(n+1)}{2}. \quad (10.14)$$

Se supone que no tenemos empates en  $X_{i*}$ . Si  $T_i$  es muy grande significa que en la característica  $i$  los valores en la primera clase ( $C_1$ ) tienden a ser mayores que en la segunda ( $C_2$ ). Un valor muy pequeño indica lo contrario. Notemos que  $T_i$  puede tomar valores negativos. En general, si el valor absoluto  $|T_i|$  es muy grande, lo que tenemos es que ambos grupos de valores son claramente distintos, uno mayor que el otro. Valores próximos a cero indican que no hay diferencias entre los grupos. Hasta aquí no hay problema ninguno. Simplemente se propone utilizar un test de Mann-Whitney-Wilcoxon para comparar los órdenes en los dos grupos que comparamos. El problema aparece cuando los tamaños de las librerías son distintas que es lo habitual. De hecho, suelen ser muy distintos. Una primera idea que no da buenos resultados es dividir los conteos originales por los correspondientes tamaños de las librerías.

Suponemos fijos los tamaños de las librerías. Esto es, aunque suponemos conteos aleatorios la suma de las correspondientes columnas de conteos las asumimos como fijas.<sup>182</sup> Denotamos el mínimo tamaño de las librerías con  $m_0 = \min_{j=1, \dots, n} m_j$  y la librería donde se da este mínimo suponemos que es la  $j_0$ . Supongamos que, para la librería  $j$ , del total de lecturas elegimos al azar  $m_0$  (correspondiente a la librería con menor tamaño). Esto supone que cada lectura original de esta librería es conservada con probabilidad  $\frac{m_0}{m_j}$  y eliminada con probabilidad  $1 - \frac{m_0}{m_j}$ . Si denotamos por  $X'_{ij}$  el número de lecturas que nos quedan después de este proceso de muestreo tendremos que

$$X'_{ij} | X_{ij} \sim \text{Binomial}(X_{ij}, \frac{m_0}{m_j}).$$

<sup>181</sup> A veces se le llama de Mann-Whitney, otras de Wilcoxon y otras con los tres nombres.

<sup>182</sup> Una hipótesis necesaria.

Los autores llaman a esto un **muestreo hacia abajo**.<sup>183</sup> Este procedimiento de muestreo nos produce unos nuevos conteos y un nuevo valor del estadístico dado en 10.14 que podemos denotar por  $T'_i$ . Notemos que si el tamaño mínimo es mucho menor que el resto de tamaños estamos descartando muchas lecturas.

<sup>183</sup> Down sampling.

Una segunda opción es igualar los tamaños de las librerías a un valor intermedio como la media geométrica  $m^* = \left( \prod_{j=1}^n m_j \right)^{1/n}$ . Generamos conteos aleatorios con

$$X'_{ij} | X_{ij} \sim \text{Poisson}\left(\frac{m^*}{m_j} X_{ij}\right). \quad (10.15)$$

Sería el muestreo Poisson. Para evitar empates entre los valores generados se sustituyen estos valores por

$$X'_{ij} + \epsilon_{ij}$$

siendo  $\epsilon_{ij}$  variables aleatorias independientes y con distribución común uniforme en el intervalo  $[0, 0.1]$ .<sup>184</sup> Como vemos estamos generando valores por lo que lo que obtenemos también es aleatorio. Supongamos que en un muestreo obtenemos  $T'_i(b)$ . Repetimos este muestreo  $B$  veces. Consideramos el estadístico

$$T_i^* = \frac{1}{B} \sum_{b=1}^B T'_i(b). \quad (10.16)$$

Este es el estadístico con el que finalmente se trabaja para contrastar. No conocemos la distribución de este estadístico. Hemos indicado que la región crítica natural, la zona en donde rechazamos la hipótesis nula de que no hay diferencia significativa entre muestras vendría dada por  $T_i^* > c$  siendo  $c$  un valor a determinar. Dado un valor positivo  $c$  consideraremos como significativa a la característica  $i$  si  $T_i^* > c$ . Lo que vamos a hacer es elegir un conjunto de posibles valores para  $c$  y **estimaremos** la tasa de hipótesis nulas falsamente rechazadas si utilizamos ese valor de  $c$ . El procedimiento<sup>185</sup> es el siguiente:

1. Calculamos  $T_1^*, \dots, T_N^*$  a partir de nuestros datos.
2. Permutamos  $S$  veces las columnas de  $X$  manteniendo la asignación de los grupos (las etiquetas de clase) y calculamos  $T_1^*(s), \dots, T_N^*(s)$  utilizando los datos permutados con  $s = 1, \dots, S$ .
3. Tomamos una serie de valores para  $c$  y calculamos:

$$\hat{V} = \frac{1}{S} \sum_{i=1}^N \sum_{s=1}^S 1_{|T_i^*(s)| > c} \quad (10.17)$$

que estima la cantidad de hipótesis nulas falsamente rechazadas y

$$\hat{R} = \sum_{i=1}^N 1_{|T_i^*| > c} \quad (10.18)$$

que estima la cantidad de hipótesis nulas rechazadas.

4. El valor de FDR asociado a cada valor  $c$  es estimado con

$$\widehat{\text{FDR}}(c) = \frac{\hat{\pi}_0 \hat{V}}{\hat{R}}. \quad (10.19)$$

donde  $\pi_0$  es la proporción de hipótesis nulas ciertas.<sup>186</sup> El estimador de  $\pi_0$  utilizado es

$$\hat{\pi}_0 = \frac{2 \sum_{i=1}^N 1_{|T_i^*| \leq q}}{N},$$

siendo  $q$  la mediana de  $\{|T_i^*(s)| : i = 1, \dots, N; s = 1, \dots, S\}$ .

<sup>184</sup> Un pequeño truco para evitar empates.

<sup>185</sup> Ver [132].

<sup>186</sup> O proporción de características que no son significativamente distintas entre los dos grupos que comparamos.

### 10.7.1 SAMseq: PRJNA297664

```
pacman::p_load("SummarizedExperiment", "samr")
data(PRJNA297664, package="tamidata")

PRJNA297664samseq = SAMseq(x = assay(PRJNA297664),
  y = colData(PRJNA297664) [,"treatment"],
  resp.type="Two class unpaired", geneid = rownames(PRJNA297664),
  genenames=rownames(PRJNA297664), nperms = 1000, nresamp = 40,
  fdr.output = 0.20)
```

<sup>187</sup> No mostrado.

Podemos ver los resultados con<sup>187</sup>

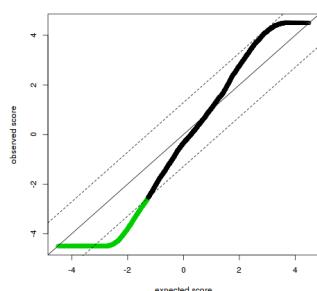


Figura 10.1: Genes significativos con los datos PRJNA297664.

```
print(PRJNA297664samseq)
```

Un dibujo con los resultados que podemos ver en figura 10.1

```
plot(PRJNA297664samseq)
```

## 10.8 DESeq2

Este método se propuso en [91].

### 10.8.1 Método

Como es habitual en el texto denotamos la matriz de conteos aleatoria con  $X$  donde el elemento en posición  $(i, j)$  es el conteo aleatorio  $X_{ij}$  correspondiente al  $i$ -ésimo gen y a la  $j$ -ésima muestra. Se asume que  $X_{ij}$  sigue una distribución binomial negativa (§ 21.8.2) con media  $\mu_{ij}$  y dispersión  $\alpha_i$ . Se supone que la media de este conteo aleatorio la podemos expresar como

$$\mu_{ij} = s_{ij}q_{ij}, \quad (10.20)$$

siendo  $s_{ij}$  un factor de normalización que tiene en cuenta las distintas profundidades de secuenciación así como otros posibles sesgos como contenido GC del gen o su longitud.<sup>188</sup> La dependencia de las covariables (variables fenotípicas) se introduce en  $q_{ij}$ . En concreto si asumimos para la muestra  $j$  un vector de covariables  $y_j = (y_{j1}, \dots, y_{jp})'$  y denotamos un vector de coeficientes (dependiente de la fila o gen) con  $\beta_i = (\beta_{i1}, \dots, \beta_{ip})'$ . Se asume el siguiente modelo

$$\log_2 q_{ij} = y_j' \beta_i = \sum_{k=1}^p y_{jk} \beta_{ik}.$$

<sup>188</sup> En la propia modelización incorporar el proceso de normalización que, en principio, puede depender del gen y de la muestra. Por defecto se utilizan  $s_{ij}$  que no dependen de  $i$  y están definidos en § 6.14.3.

<sup>189</sup> Un detalle importante sobre la codificación de un factor. No se utilizan las habituales variables dummy que codifican uno para una categoría y cero para las demás dejando la media como el valor en una categoría de referencia. Los autores utilizan un parámetro para la media y una variable dummy por cada nivel del factor experimental. Esto ocasiona que la ma-

<sup>189</sup>

Los contrastes vendrán dados por  $c' \beta_i$ . El error estándar (su desviación típica) del contraste  $c' \beta_i$  viene dada por  $SE(c' \beta_i) = \sqrt{c' \Sigma_i c}$ , donde  $\Sigma_i$  denota la matriz de covarianzas de  $\beta_i$ . Cuando se contrasta si estos contrastes<sup>190</sup> son nulos se utiliza la matriz de diseño estándar en donde no se introduce el parámetro adicional que hemos comentado anteriormente. Esta sí es una matriz de rango completo por columnas.

**Estimando las dispersiones.** Sobre las distintas dispersiones  $\phi_i$  se asume una distribución lognormal.

$$\log \phi_i \sim N(\log \tilde{\phi}(\bar{\mu}_i), \sigma_d^2) \quad (10.21)$$

donde la media es una función  $\tilde{\phi}$  que evaluamos en

$$\bar{\mu}_i = \frac{1}{n} \sum_{j=1}^n \frac{x_{ij}}{s_{ij}}. \quad (10.22)$$

¿Qué papel desempeña la función  $\tilde{\phi}$ ? Nos describe cómo depende la media de la distribución a priori (sobre el parámetro de dispersion) de la media muestral de los conteos normalizados observados que denotamos  $\bar{\mu}_i$ . La varianza  $\sigma_d^2$  modeliza la variabilidad de las dispersiones alrededor de este comportamiento medio que acabamos de indicar. Se utiliza la siguiente función  $\tilde{\phi}$ .<sup>191</sup>

$$\tilde{\phi}(\bar{\mu}) = \frac{a_1}{\bar{\mu}} + \phi_0. \quad (10.23)$$

En resumen, tenemos tres parámetros en la distribución a priori.<sup>192</sup> Para tener especificada esta distribución a a priori tenemos que estimar  $a_1, \phi_0, \sigma_d^2$ . Se utiliza un método empírico bayesiano.<sup>193</sup> ¿Cómo los estimamos?

**Estimación de las dispersiones  $\phi_i$ 's** Se empieza ajustando un modelo lineal generalizado con componente aleatoria binomial negativa. Se realiza una primera estimación en donde aplicando el método de los momentos y los conteos estimamos las medias y los parámetros de dispersion. De este modo se tiene una primera estimación de las medias que denotamos  $\hat{\mu}_{ij}^0$ . Una vez estimadas las medias podemos estimar las dispersiones. Se considera la verosimilitud ajustada de Cox-Reid. Sea  $l(\phi)$  la logverosimilitud dada por (eliminamos la referencia a la fila  $i$  pero obviamente esta función se considera para cada gen)

$$l(\phi) = \sum_{j=1}^n \log f_{NB}(x_j | \mu_j, \phi)$$

donde  $f_{NB}(x_j : \mu_j, \phi)$  es la función de probabilidad de la binomial negativa con media  $\mu_j$  y dispersión  $\phi$ .<sup>194</sup> Denotamos por  $D$  la matriz de diseño. La verosimilitud ajustada de Cox-Read viene dada por

$$l_{CR}(\phi | \boldsymbol{\mu}, , \phi) = l(\phi) - \frac{1}{2} \log(\det(D'WD)), \quad (10.24)$$

siendo  $\boldsymbol{\mu} = (\mu_1, \dots, \mu_n)$  y  $W$  la matriz de pesos diagonal de los mínimos cuadrados iterativamente reponderados. Dado que la función de enlace es  $g(\mu) = \log(\mu)$  y la varianza viene dada por  $V(\mu|\phi) = \mu + \phi\mu^2$  entonces el  $j$  elemento de la diagonal es

$$w_{jj} = \frac{1}{g'(\mu_j)^2 V(\mu|\phi)} = \frac{1}{\frac{1}{\mu_j} + \phi}.$$

Se maximiza la función dada en 10.24 y obtenemos un estimador que denotamos por  $\phi^{gw}$  para cada gen. Para el  $i$ -ésimo gen tenemos  $(\bar{\mu}_i, \phi_i^{gw})$ .

<sup>191</sup> En [91, pág. 15] hay una interesante discusión sobre la elección de esta función.

<sup>192</sup> Por ser parámetros sobre una distribución de probabilidad para un parámetro realmente hemos de hablar de hiperparámetros en terminología bayesiana.

<sup>193</sup> Los propios datos son utilizados para estimar los hiperparámetros de la distribución a priori.

<sup>194</sup> Definida en 21.21.

**Tendencia de la dispersión** Pretendemos estimar la función propuesta en 10.23. Se aplica iterativamente un modelo lineal generalizado utilizando la familia de distribuciones gamma. ¿Por qué iterativamente? Porque se van eliminando en cada iteración genes donde el ajuste es malo. ¿En qué sentido? En una iteración dada ajustamos el modelo lineal generalizado. Con el modelo ajustado en esa iteración tendremos una predicción y el valor original ( $\phi_i^{gw}$ ). Si el cociente del primero respecto del segundo no está en el intervalo  $[10^{-4}, 15]$  es eliminado del estudio. El proceso continúa iterativamente hasta que la suma de los logaritmos de los cocientes de los nuevos coeficientes respecto de los antiguos es menor que  $10^{-6}$ .

**Distribución a priori sobre la dispersión** Consideremos los residuos logarítmicos dados por

$$\log \phi_i^{gw} - \log \tilde{\phi}(\bar{\mu}_i).$$

La desviación estándar de estos residuos es estimada (evitando observaciones anómalas) utilizando el estimador basado en MAD (mediana de las desviaciones absolutas respecto de la mediana, § 20.4). Es decir con

$$s_{lr} = \frac{1}{\Phi^{-1}\left(\frac{3}{4}\right)} \text{MAD}(\{\log \phi_i^{gw} - \log \tilde{\phi}(\bar{\mu}_i) : i = 1, \dots, N\}). \quad (10.25)$$

Finalmente, la varianza de la distribución a priori es estimada con

$$\sigma_d^2 = \max\{s_{lr}^2 - \psi_1((n-p)/2), 0.25\}, \quad (10.26)$$

donde  $\psi_1$  es la función trigamma;  $n$ , el número de muestras y  $p$  el número de covariables por muestra.

El estimador 10.26 no es adecuado cuando los grados de libertad residuales (número de muestras  $n$  menos número de parámetros a estimar  $p$ ) son de tres o inferior. En este caso se propone un método basado en simulación que se puede consultar en [91, pág. 16].

**Estimaciones de la dispersión.** El estimador final de la dispersión  $\phi_i$  se obtiene

$$\phi_i^{MAP} = \operatorname{argmax}_{\phi} \left( l_{CR} \left( \phi; \bar{\mu}_i^0, x_i \right) + \Lambda_i(\phi) \right) \quad (10.27)$$

siendo

$$\Lambda_i(\phi) = \frac{-(\log \phi - \log \tilde{\phi}(\bar{\mu}_i))^2}{2\sigma_d^2}.$$

**Outliers de dispersion.** En el punto anterior hemos indicado cómo estimamos la dispersión para cada gen. No obstante, si un gen verifica que

$$\log \phi_i^{gw} > \log \tilde{\phi}(\bar{\mu}_i) + 2s_{lr}$$

entonces consideramos el gen como un outlier en términos de dispersión. Para estos genes utilizamos como estimador de su dispersión no el propuesto en el apartado anterior sino  $\phi_i^{gw}$ . Además estos genes no son utilizados cuando ajustamos  $\tilde{\phi}$ .

**Distribución sobre los coeficientes del modelo** Asumimos que

$$\beta_{ir} \sim N(0, \sigma_r^2). \quad (10.28)$$

Para el  $i$ -ésimo gen aplicamos el algoritmo (habitual) de los mínimos cuadrados iterativamente reponderados y obtenemos los estimadores máximo verosímiles  $\beta_{ir}^{MLE}$ . Consideramos el cuantil empírico de orden  $1-p$  de los  $|\beta_{ir}^{MLE}|$  y lo denotamos por  $q_{1-p}(|\beta_r|)$ . Si considerados  $q_N(1-p)$  el cuantil  $1-p$  de una normal estándar entonces estimamos con  $\sigma_r = \frac{q_{1-p}(|\beta_r|)}{q_N(1-p/2)}$ . Se toma por defecto  $p = 0.05$ .

### Estimadores de $\beta_i$

Se estima  $\beta_i$  como el valor que maximiza

$$\sum_{j=1}^n f_{NB}(x_{ij} | \mu_j(\boldsymbol{\beta}), \phi_i) + \Lambda(\boldsymbol{\beta}) \quad (10.29)$$

siendo

$$\mu_j(\boldsymbol{\beta}) = s_{ij} e^{\sum_{r=1}^p y_{jr} \beta_r},$$

$$\Lambda(\boldsymbol{\beta}) = \sum_{r=1}^p \frac{-\beta_r^2}{2\sigma_r^2},$$

y  $\phi_i = \phi_i^{MAP}$  excepto para los genes que son outliers de dispersión en donde  $\phi_i = \phi_i^{gw}$ . La función 10.29 es maximizada utilizando el algoritmo de regresión ridge iterativamente reponderada. En concreto en una iteración dada actualizamos los valores de los coeficientes con

$$\boldsymbol{\beta} \leftarrow (D' WD + \lambda I)^{-1} D' W z,$$

donde  $\lambda_r = \frac{1}{\sigma_r^2}$  y

$$z_j = \log \frac{\mu_j}{s_j} + \frac{x_j - \mu_j}{\mu_j}$$

siendo  $\mu_j(\boldsymbol{\beta}) = s_{ij} e^{\sum_{r=1}^p y_{jr} \beta_r}$  donde los valores de  $\beta_r$  son la estimación en la iteración actual.

**Test de Wald.** En un test de Wald se comparan los estimadores de los coeficientes  $\beta_{ir}$  con sus errores estándar estimados  $SE(\beta_{ir})$  que corresponden con el elemento en posición  $(r, r)$  de la matriz de covarianzas de  $\boldsymbol{\beta}_i$ . Esta matriz viene dada por

$$cov(\boldsymbol{\beta}_i) = (D' WD + \lambda I)^{-1} (D' WD) (D' WD + \lambda I)^{-1}.$$

El cociente del estimador con el error estándar tiene aproximadamente una distribución normal estándar bajo la hipótesis nula de que el coeficiente es nulo. Aplicando el test de Wald tendremos p-valores para el test bilateral de que el coeficiente no es nulo. Finalmente estos p-valores pueden ser ajustados mediante algún procedimiento como el método de Benjamini-Hochberg.

En el trabajo se discute la integración de un método de filtrado independiente basado en la media de los conteos normalizados y la adaptación a hipótesis nulas compuestas así como detalles adicionales que no comentamos. En esencia este es el método propuesto.

### 10.8.2 Paquete

Vamos a analizar los datos tamidata2::PRJNA218851.

```
pacman::p_load(SummarizedExperiment, DESeq2)
```

Leemos los datos.

```
data(PRJNA218851, package = "tamidata2")
```

Tenemos la covariable `Stage` que indica el tipo de tejido analizado.

Tenemos tres muestras por individuo. Por tanto tenemos un factor intra sujeto con tres niveles. No vamos a utilizar este diseño. Simplemente vamos a considerar que tenemos tres grupos independientes de muestras.

Empezamos generando el objeto de clase `DESeqDataSet` a partir de nuestro objeto de clase `RangedSummarizedExperiment`.

```
dds = DESeqDataSet(PRJNA218851, design = ~ Stage)
```

Eliminamos genes con conteos bajos.

```
keep = rowSums(counts(dds)) >= 10
dds = dds[keep, ]
```

Fijamos el nivel `control` como categoría de referencia.

```
dds$Stage <- relevel(dds$Stage, ref = "Normal")
```

Podemos hacer el estudio de la expresión diferencial.

```
dds = DESeq(dds)
```

Podemos evaluar el coeficiente correspondiente a la comparación entre el nivel `cancer` y el de referencia `control`.

```
resLFC2 = lfcShrink(dds, coef=2)
```

Ahora evaluamos los tests correspondientes al coeficiente que compara el nivel `metastasis` con el nivel de referencia `control`.

```
resLFC3 = lfcShrink(dds, coef=3)
```

### 10.9 Material adicional

En [123] tenemos un estudio comparativo de distintos procedimientos para expresión diferencial utilizando distintos paquetes de Bioconductor.

## 10.10 Ejercicios

Ej. 25 — Para los datos tamidata::PRJNA297664 se pide:

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?



## Capítulo 11

# Tamaño muestral y potencia

Los paquetes que vamos a utilizar son

**Microarrays** [112, sizepower],[109, OCplus], [146, ssize],

**RNASeq** [135, RNASeqPower], [R-RnaSeqSampleSize] y sus datos asociados en [162, RnaSeqSampleSizeData]. [155, PROPER],

**Ambos** [141, SSPA]

**ChIP-Seq** [164, CSSP]



# Capítulo 12

## Generando un informe

En temas previos hemos visto cómo obtener subconjuntos de genes con expresión diferencial. Ya tenemos la lista. Lo habitual es querer explorar este grupo de genes. Y el modo más útil posiblemente sea generar un archivo en [HTML](#) con enlaces a bases de datos donde nos proporcionan información sobre ellos como puede ser [Gene Ontology](#). Esto nos permitirá consultar posteriormente nuestros resultados con nuestro navegador.<sup>1</sup>

La complejidad del problema que se aborda hace que este paso sea muy importante. En la práctica estadística clásica se generaba un informe y el investigador interpretaba los resultados y redacta un informe. En este contexto el problema es mayor. Básicamente se explora en la gran cantidad de genes y por ello los resultados han de ser validados y, posiblemente, nuevas hipótesis generadas.

En este tema nos proponemos el siguiente objetivo: **generar un fichero en que cada fila contenga distintos identificadores de aquellos genes que declaramos con expresión diferencial significativa así como los p-valores originales y los ajustados.**

Utilizamos como ejemplo los datos GSE20986.<sup>195</sup>

<sup>195</sup> § 5.11.

```
library(Biobase)
data(gse20986, package="tamidata")
eset = gse20986
```

En concreto nos vamos a fijar en la comparación entre las muestras extraídas de células endoteliales vasculares de la retina y de células endoteliales de la vena umbilical (abreviadamente hablamos de células de la retina y huvec). Con el siguiente código<sup>196</sup> nos aplicamos un t-test, a los p-valores obtenidos le aplicamos el método de Benjamini-Hochberg con 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.

<sup>196</sup> § 7 y § 8.

```
library(genefilter)
library(multtest)
eset = gse20986[, c(2, 3, 5, 10:12)]
tissue = factor(rep(1:2, each = 3), levels = 1:2,
  labels = c("retina", "huvec"))
tt = rowttests(eset, tissue)
```

<sup>1</sup> Que esperemos no sea Internet Explorer.

```
p.BH = mt.rawp2adjp(tt$p.value, "BH")
sig = which(p.BH$adjp[p.BH$index, 2] < 0.05)
```

En el vector `sig` tenemos las filas que ocupan, en la matriz de expresión, los genes declarados significativos por el procedimiento indicado. Podemos ver las primeras filas (no están ordenados según que el p-valor original o ajustado sea menor, simplemente son las primeras filas de la matriz) son:

```
head(sig)
## [1] 10 33 146 264 385 626
```

El número de significativos es

```
nsig = length(sig)
```

Guardamos los p-valores y los p-valores ajustados.

```
pvalor = p.BH$adjp[1:nsig,1]
pajustado = p.BH$adjp[1:nsig,2]
```

En § 12.1 vemos cómo obtener distintos identificadores de este grupo de genes utilizando el paquete [54, `annotate`]. En § 12.2 y § 12.3 generamos ficheros HTML utilizando estos identificadores con los paquetes [85, `R2HTML`] y [68, `ReportingTools`] respectivamente. En § 12.2 y § 12.3 ofrecemos dos opciones alternativas (la mejor es la segunda) y necesitan lo que vemos en § 12.1.

## 12.1 Identificando el grupo de genes

El primer problema es asociar a los genes seleccionados un identificador. ¿Qué anotación tienen nuestros datos? Necesitamos el paquete [54, `annotate`].

```
library(annotate)
```

La función `annotate::annotation` nos indica la anotación.

```
annotation(eset)
## [1] "hgu133plus2"
```

Cargamos (después de instalarlo) el paquete correspondiente.

```
library(hgu133plus2.db)
```

Obtenemos los identificadores de nuestros genes. Tenemos un chip de Affymetrix por ello tendremos los identificadores AffyID. Puesto que tenemos un ExpressionSet podemos hacer

```
ID = featureNames(eset)[sig]
```

Los primeros serían

```
head(ID)

## [1] "1431_at"      "1552286_at"    "1552463_at"
## [4] "1552628_a_at" "1552794_a_at" "1553132_a_at"
```

Podemos determinar los [símbolos](#) de nuestros genes. Dos posibles formas de hacerlo son

```
lookUp(ID, "hgu133plus2.db", "SYMBOL")
getSYMBOL(ID, "hgu133plus2.db")
```

Sus nombres con

```
lookUp(ID, "hgu133plus2.db", "GENENAME")
```

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")
```

¿O en [Gene Ontology](#)?

```
lookUp(ID, "hgu133plus2.db", "GO")
```

O bien sus identificadores en [Entrez](#).

```
lookUp(ID, "hgu133plus2.db", "ENTREZID")
```

Y bastante más información. Toda la información que podemos obtener se puede consultar en la ayuda del paquete [[26](#), hgu133plus2.db] o bien con

```
ls("package:hgu133plus2.db")
```

## 12.2 R2HTML

Veamos cómo hacerlo con el paquete [[85](#), R2HTML].

```
library(R2HTML)
```

En § [12.1](#) hemos visto cómo obtener a partir de las filas de la matriz de expresión distintos identificadores de los genes. Repetimos el código de dicha sección guardando los resultados.

```
ID = featureNames(eset)[sig]
Symbol = getSYMBOL(ID, "hgu133plus2.db")
Name = as.character(lookUp(ID, "hgu133plus2.db", "GENENAME"))
Ensembl = as.character(lookUp(ID, "hgu133plus2.db", "ENSEMBL"))
entrezid = as.character(lookUp(ID, "hgu133plus2.db", "ENTREZID"))
```

Lo siguiente es definir un `data.frame` donde reunimos toda la información. Incorporamos los p-valores y los p-valores ajustados.

```
tmp = data.frame(ID=ID, Symbol=Symbol, Name>Name, Ensembl=Ensembl,
                  entrezid = entrezid,pvalor = pvalor,pajustado = pajustado,
                  stringsAsFactors=F)
```

Podemos tener campos vacíos que hay que declarar como datos faltantes.

```
tmp[tmp=="NA"] = NA
HTML(tmp, "reports/prueba.html", append=F)
```

El informe que hemos generado lo podemos ver con el navegador.<sup>2</sup>

```
browseURL("reports/prueba.html")
```

Parece que un fichero HTML sin hipervínculos no parece muy útil. Se los añadimos.

```
Ensembl = ifelse(Ensembl=="NA", NA,
                  paste("<a href='http://useast.ensembl.org/Homo_sapiens/Gene/Summary?g='",
                        Ensembl, "'>", Ensembl, "</a>", sep=""))
tmp = data.frame(ID=ID, Symbol=Symbol, Name>Name, Ensembl=Ensembl,
                  stringsAsFactors=F)
entrezid = ifelse(entrezid=="NA", NA,
                  paste("<a href='http://www.ncbi.nlm.nih.gov/gene/?term='",
                        entrezid, "'>", entrezid, "</a>", sep=""))
tmp = data.frame(ID=ID, Symbol=Symbol, Name>Name, Ensembl=Ensembl,
                  entrezid = entrezid,pvalor = pvalor,pajustado = pajustado,
                  stringsAsFactors=F)
tmp[tmp=="NA"] <- NA
HTML(tmp, "reports/prueba.html", append=F)
```

Y vemos el nuevo fichero.

```
browseURL("reports/prueba.html")
```

## 12.3 ReportingTools

En esta sección vemos cómo generar un informe con [68, ReportingTools]. Cargamos el paquete.

```
library(ReportingTools)
```

Obtenemos distintos identificadores.

```
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,
                  paste("<a href='http://www.ncbi.nlm.nih.gov/gene/?term='",
                        entrezid, "'>", entrezid, "</a>", sep=""))
```

---

<sup>2</sup>Crucemos los dedos para que no sea Internet Explorer.

Possiblemente 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)
```

Utilizando `ReportingTools::HTMLReport` fijamos el nombre del fichero en que guardamos la información así como el directorio en donde estará dicho fichero.

```
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"
```

## 12.4 Ejercicios

### Ejercicio 26

Utilizando el ExpressionSet `gse1397` se pide:

1. ¿Cuál es el chip utilizado?
2. Seleccionar el gen en la fila 678. ¿Cuál es su identificador Affymetrix, AffyID? Determinar los identificadores en las siguientes bases de datos: Ensembl, Gene Ontology y NCBI.
3. Elegir al azar 100 genes con la función `sample`. Determinar sus AffyIDs y los correspondientes en Ensembl, Gene Ontology y NCBI.
  - (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 NCBI nos aparezca en vínculo para el acceso a la base de datos correspondiente.



# **Parte IV**

## **Reducción de dimensión y clasificación**



## Capítulo 13

# Componentes principales

### 13.1 Introducción

En este tema nos ocupamos de problemas de reducción de dimensión. ¿Qué significa reducir la dimensión? Responder a esta pregunta es obvio si nos fijamos en los datos que tenemos. Trabajando con expresión de genes tenemos tantas filas como genes y tantas columnas como muestras. En resumen miles de filas y decenas o centenares de columnas. En el tema En temas anteriores hemos visto como seleccionar filas, esto es, seleccionar genes es una tarea incluso previa. Hemos de quedarnos con genes que tengan una expresión diferencial si consideramos alguna característica fenotípica o bien con genes que tengan una expresión mínima o bien con genes que tengan un cierto nivel de variación. ¿Qué hacemos con las columnas? O de otro modo: ¿qué hacemos con las muestras? Quizás la respuesta natural sería: si tenemos miles de filas, ¿por qué preocuparse de unas decenas de filas? No es una buena respuesta. Realmente tener 50 o 100 columnas son muchas a la hora de visualizar resultados o bien de aplicar tratamientos estadísticos. En este tema tratamos el tema de cómo reducir el número de columnas o el número de filas.

### 13.2 Componentes principales

Para ilustrar los conceptos vamos a considerar unos datos sencillos. Tomamos los datos *golub* y nos fijamos en los genes que tienen que ver con “Cyclin” (tienen esta palabra en su nombre). Vamos a considerar las dos primeras muestras, esto es, las dos primeras columnas.

```
data(golub, package="multtest")
sel = grep("Cyclin", golub.gnames[,2])
golub.red = golub[sel, 1:2]
```

Los datos aparecen en la figura 13.1.

```
pacman::p_load("ggplot2")
df = data.frame(golub.red)
names(df) = c("sample1", "sample2")
png(paste0(dirTamiFigures, "PCA2.png"))
ggplot(df, aes(x=sample1, y=sample2)) + geom_point()
dev.off()
```

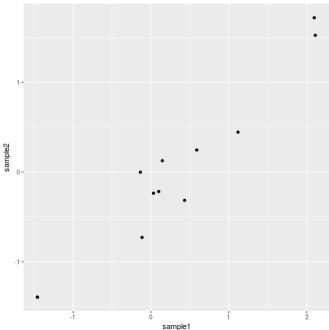


Figura 13.1: Expresión de las dos primeras muestras de los datos golub para aquellos genes que contienen la expresión Cyclin.

```
## pdf
## 2
```

Cada punto corresponde con uno de los genes seleccionados.

Para la fila  $i$  (para el gen  $i$ ) denotamos las expresiones observadas en las dos muestras como  $x_i = (x_{i1}, x_{i2})$ . Tenemos  $n$  filas y por lo tanto nuestros datos son  $x_i$  con  $i = 1, \dots, n$ .

Centramos los datos. Esto es, le restamos a cada columna la media de la columna. Para ello, primero calculamos las medias. El vector de medias lo vamos a denotar por  $\bar{x} = (\bar{x}_1, \bar{x}_2)$  donde

$$\bar{x}_j = \sum_{i=1}^n \frac{x_{ij}}{n}$$

es decir, cada componente es la media de las componentes. En resumen el primer valor es la expresión media en la primera muestra para todos los genes. Podemos calcular fácilmente el vector de medias. Una función específica es la siguiente.

```
medias = colMeans(golub.red)
```

También podemos usar la función `apply` que nos hace lo mismo.

```
medias = apply(golub.red, 2, mean)
```

Le restamos a cada columna su media.

```
golub.red = sweep(golub.red, 2, medias)
```

En la figura 13.2 reproducimos los datos centrados y mostramos los ejes de coordenadas en rojo.

```
df = data.frame(golub.red)
names(df) = c("sample1", "sample2")
p = ggplot(df, aes(x=sample1, y=sample2)) + geom_point()
p = p + geom_vline(xintercept=mean(golub.red[, 1]), color = "red")
p + geom_hline(yintercept=mean(golub.red[, 2]), color = "red")
plot(golub.red)
abline(v=mean(golub.red[, 1]), col="red")
abline(h=mean(golub.red[, 2]), col="red")
```

Hemos trasladado los datos de modo que las medias de cada variable valen cero ahora. Esto es lo que se conoce como centrar los datos. Hemos *centrado los datos*. Podemos comprobar que los nuevos datos tienen una media nula.

```
colMeans(golub.red)
```

```
## [1] -3.006854e-17  1.069746e-17
```

Nuestros datos (filas) corresponden a las expresiones correspondientes a los genes. Los datos originales tienen dimensión 2 (dos variables correspondientes a las dos muestras) y supongamos que pretendemos reducir la dimensión a solo una, esto es, representar cada gen mediante un único número. La idea de las componentes principales es considerar una combinación lineal de los valores originales. Es decir, se pretende elegir un vector (de dimensión dos)  $a_1 = (a_{11}, a_{12})$  de modo que

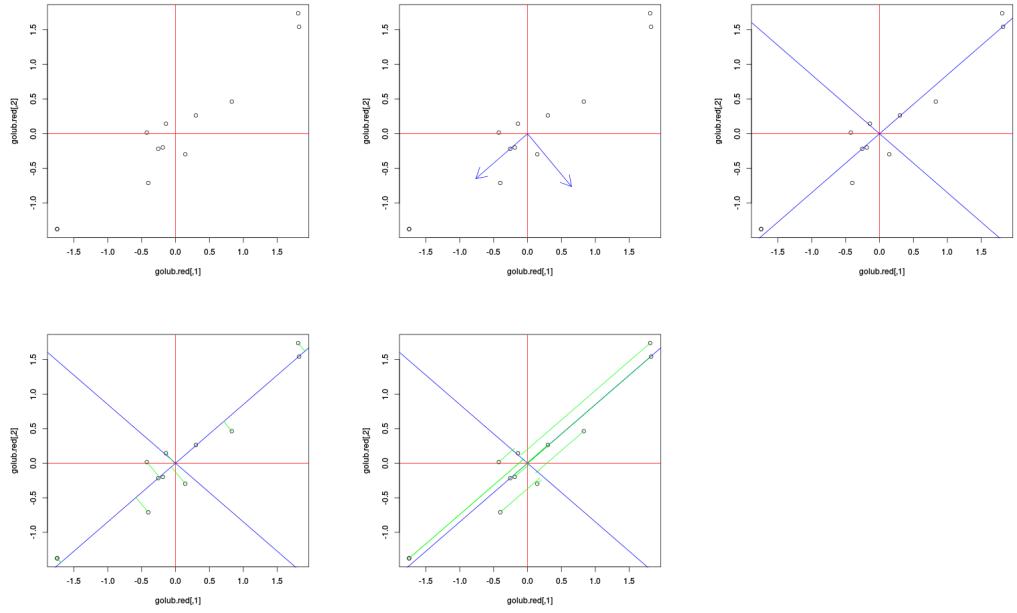


Figura 13.2: a) Centramos los datos golub.red y mostramos el nuevo sistema de coordenadas en rojo. b) Vectores que definen las líneas donde proyectamos. c) Las líneas sobre las que proyectamos. d) Datos golub.red mostrando las líneas sobre las que proyectamos (en azul) para obtener las dos componentes principales y las proyecciones sobre la *primera* componente en verde. e) Datos golub.red mostrando las líneas sobre las que proyectamos (en azul) para obtener las dos componentes principales y las proyecciones sobre la *segunda* componente.

en lugar de utilizar  $x_i$  consideremos (el resumen)  $u_i = a_{11}x_{i1} + a_{12}x_{i2}$ . ¿Qué  $a_1$  elegimos? La idea es lograr que los valores  $u_i$  tengan la mayor variabilidad que se pueda con objeto de no perder información. Mantener la variabilidad original indica que mantenemos la información que los datos originales tienen. En concreto se elige  $a_1$  de modo que maximizamos

$$\frac{1}{n} \sum_{i=1}^n (u_i - \bar{u})^2.$$

El vector  $a_1$  nos indica la dirección sobre la cual proyectamos los datos originales. Las proyecciones sobre  $a_1$ , los valores  $u_i$  son la mejor descripción univariante de los datos. La segunda mejor descripción que sea ortogonal a la anterior serían las proyecciones sobre la línea orthogonal a la primera que pasa por el origen de coordenadas. Obtengamos las componentes principales.

```
a.pca = prcomp(golub.red)
```

Los vectores directores de las líneas sobre las que proyectamos aparecen en la figura 13.2. Estos vectores son

```
a.pca$rotation
```

```
##          PC1         PC2
## [1,] -0.7619878  0.6475914
## [2,] -0.6475914 -0.7619878
```

Las líneas sobre las que proyectamos aparecen en azul en la figura 13.2 Hemos trasladado los datos de modo que las medias de cada variable valen cero ahora. Esto es lo que se conoce como centrar los datos. Hemos *centrado los datos*. Podemos comprobar que los nuevos datos tienen una media nula.

```
colMeans(golub.red)
## [1] -3.006854e-17  1.069746e-17
```

Nuestros datos (filas) corresponden a las expresiones correspondientes a los genes. Los datos originales tienen dimensión 2 (dos variables correspondientes a las dos muestras) y supongamos que pretendemos reducir la dimensión a solo una, esto es, representar cada gen mediante un único número. La idea de las componentes principales es considerar una combinación lineal de los valores originales. Es decir, se pretende elegir un vector (de dimensión dos)  $a_1 = (a_{11}, a_{12})$  de modo que en lugar de utilizar  $x_i$  consideremos (el resumen)  $u_i = a_{11}x_{i1} + a_{12}x_{i2}$ . ¿Qué  $a_1$  elegimos? La idea es lograr que los valores  $u_i$  tengan la mayor variabilidad que se pueda con objeto de no perder información. Mantener la variabilidad original indica que mantenemos la información que los datos originales tienen. En concreto se elige  $a_1$  de modo que maximizamos

$$\frac{1}{n} \sum_{i=1}^n (u_i - \bar{u})^2.$$

El vector  $a_1$  nos indica la dirección sobre la cual proyectamos los datos originales. Las proyecciones sobre  $a_1$ , los valores  $u_i$  son la mejor descripción univariante de los datos. La segunda mejor descripción que sea ortogonal a la anterior serían las proyecciones sobre la línea ortogonal a la primera que pasa por el origen de coordenadas. Obtengamos las componentes principales.

```
a.pca = prcomp(golub.red)
```

Los vectores directores de las líneas sobre las que proyectamos aparecen en la figura 13.2(b). Estos vectores son

```
a.pca$rotation
##           PC1          PC2
## [1,] -0.7619878  0.6475914
## [2,] -0.6475914 -0.7619878
```

Las líneas sobre las que proyectamos aparecen en azul en la figura 13.2(c). Y finalmente podemos ver las proyecciones. En verde mostramos las proyecciones sobre la primera componente (figura 13.2(d)). Y ahora consideraremos la proyección sobre la segunda componente. En la figura 13.2(e) la mostramos. Los valores de estas proyecciones los obtenemos con

```
predict(a.pca)
##           PC1          PC2
## [1,] -2.50309193 -1.541823e-01
```

```
## [2,] 0.01368602 -2.024163e-01
## [3,] -2.38702381 3.714339e-03
## [4,] 0.33489688 -6.847077e-05
## [5,] 0.76608286 2.806154e-01
## [6,] 0.27144878 2.899820e-02
## [7,] 0.31169639 -2.876394e-01
## [8,] 2.22052303 -8.232084e-02
## [9,] -0.93221244 1.836866e-01
## [10,] -0.39946389 -7.239549e-03
## [11,] 0.08293509 3.191733e-01
## [12,] 2.22052303 -8.232084e-02
```

Las desviaciones estándar de la primera y segunda componente principal son las siguientes

```
## [1] 1.4687760 0.1849567
```

Y las varianzas son los cuadrados de las desviaciones estándar.

```
a.pca$sdev^2
```

```
## [1] 2.15730290 0.03420899
```

¿Cómo de variables son nuestros datos? Podemos cuantificar el total de la variación de los datos sumando las varianzas de cada una de las dos coordenadas

```
var(golub.red[,1])
```

```
## [1] 1.266931
```

```
var(golub.red[,2])
```

```
## [1] 0.9245807
```

cuya suma es

```
var(golub.red[,1]) + var(golub.red[,2])
```

```
## [1] 2.191512
```

Las nuevas coordenadas tienen la misma varianza total.

```
sum(a.pca$sdev^2)
```

```
## [1] 2.191512
```

¿Y qué proporción de la varianza es atribuible a la primera componente? ¿Y a la segunda? Podemos dividir la varianza de cada componente por la suma total.

```
variacion.total = sum(a.pca$sdev^2)
a.pca$sdev^2 / variacion.total
```

```
## [1] 0.98439023 0.01560977
```

La primera componente explica un 98.44 % de la variación total.  
¿Para qué necesitamos utilizar dos números por gen si con uno tenemos esencialmente la misma información.

### 13.3 Componentes principales de los datos golub

Hemos visto las componentes principales con dos variables (en nuestro caso dos muestras) con efecto de poder ver el significado geométrico de las componentes principales. Vamos a trabajar con el banco de datos completo: todos los datos golub que tienen 38 muestras (27 de un tipo de leucemia y 11 de otro tipo).

Obtengamos las componentes principales.

```
golub.pca = prcomp(golub, scale=FALSE, center=TRUE)
```

El argumento `center=TRUE` centra los datos restando la media de la columna de modo que las variables tengan medias nulas. El argumento `scale=TRUE` hace que las variables originales sean divididas por su desviación estándar de modo que la varianza (y la desviación estándar) de las nuevas variables sea la unidad.

Diferentes criterios podemos aplicar a la hora de decidir con cuántas componentes nos quedamos.

1. Uno puede ser la proporción total explicada. Fijar un nivel mínimo y quedarnos con el número de componentes necesario para superar este valor mínimo.
2. El segundo puede ser que una componente no puede tener una desviación estándar menor que una de las variables originales. Si hemos escalado cada variable original dividiendo por su desviación estándar entonces la desviación estándar de cada componente ha de ser mayor que uno.
3. Otro criterio puede ser ver en qué momento se produce un descenso de la desviación estándar muy notable. Quedarnos con las componentes previas.

Un resumen de las componentes nos puede indicar con cuántas nos quedamos.

```
summary(golub.pca)

## Importance of components:
##                               PC1      PC2      PC3
## Standard deviation    5.0436 1.44073 1.11734
## Proportion of Variance 0.6694 0.05462 0.03285
## Cumulative Proportion  0.6694 0.72405 0.75691
##                               PC4      PC5      PC6
## Standard deviation    1.03505 0.85821 0.74399
## Proportion of Variance 0.02819 0.01938 0.01457
## Cumulative Proportion  0.78510 0.80448 0.81905
##                               PC7      PC8      PC9
## Standard deviation    0.72104 0.69232 0.63819
## Proportion of Variance 0.01368 0.01261 0.01072
## Cumulative Proportion  0.83273 0.84534 0.85606
##                               PC10     PC11     PC12
## Standard deviation    0.63630 0.56700 0.55263
## Proportion of Variance 0.01065 0.00846 0.00804
```

### 13.3. COMPONENTES PRINCIPALES DE LOS DATOS GOLUB209

```
## Cumulative Proportion  0.86672 0.87518 0.88321
##                               PC13     PC14     PC15
## Standard deviation      0.53868 0.52011 0.49568
## Proportion of Variance 0.00764 0.00712 0.00647
## Cumulative Proportion  0.89085 0.89797 0.90443
##                               PC16     PC17     PC18
## Standard deviation      0.48402 0.47719 0.47068
## Proportion of Variance 0.00617 0.00599 0.00583
## Cumulative Proportion  0.91060 0.91659 0.92242
##                               PC19     PC20     PC21
## Standard deviation      0.45421 0.43795 0.43410
## Proportion of Variance 0.00543 0.00505 0.00496
## Cumulative Proportion  0.92785 0.93290 0.93786
##                               PC22     PC23     PC24
## Standard deviation      0.42475 0.41582 0.40718
## Proportion of Variance 0.00475 0.00455 0.00436
## Cumulative Proportion  0.94260 0.94715 0.95152
##                               PC25     PC26     PC27
## Standard deviation      0.40066 0.3948  0.38731
## Proportion of Variance 0.00422 0.0041  0.00395
## Cumulative Proportion  0.95574 0.9598  0.96379
##                               PC28     PC29     PC30
## Standard deviation      0.38417 0.37882 0.37124
## Proportion of Variance 0.00388 0.00378 0.00363
## Cumulative Proportion  0.96767 0.97145 0.97508
##                               PC31     PC32     PC33
## Standard deviation      0.36957 0.3596  0.3593
## Proportion of Variance 0.00359 0.0034  0.0034
## Cumulative Proportion  0.97867 0.9821  0.9855
##                               PC34     PC35     PC36
## Standard deviation      0.35276 0.34218 0.33228
## Proportion of Variance 0.00327 0.00308 0.00291
## Cumulative Proportion  0.98875 0.99183 0.99473
##                               PC37     PC38
## Standard deviation      0.32572 0.30667
## Proportion of Variance 0.00279 0.00247
## Cumulative Proportion  0.99753 1.00000
```

En este caso podemos aplicar el criterio 1 o 3 ya que no hemos dividido por la desviación estándar.

Si fijamos, por ejemplo un 80% de la variación total a explicar entonces debemos quedarnos con las cinco primeras componentes.

Atendiendo al tercer criterio quizás (muy subjetivo) tampoco sea una mala solución quedarnos con las cinco primeras. Puede ser una buena elección y una solución intermedia. Los nuevos datos los obtenemos con la función *predict*.

```
a = predict(golub.pca)
```

Podemos ver todas las componentes para el primer gen (primera fila).

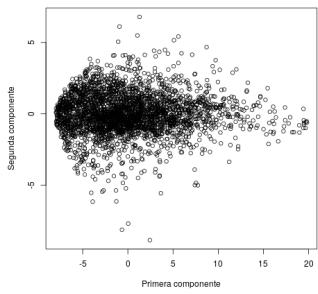
```
a[1,]

##          PC1          PC2          PC3
## -7.037559379 -1.611150783 -0.580507718
##          PC4          PC5          PC6
##  0.008740257  0.538496894  0.217863112
##          PC7          PC8          PC9
##  0.095229954 -0.918846037  0.512919401
##          PC10         PC11         PC12
##  0.863356337 -0.199100855 -0.661871987
##          PC13         PC14         PC15
##  0.098500958  1.167024082 -0.080881207
##          PC16         PC17         PC18
##  0.019311238  0.311828852 -0.734193835
##          PC19         PC20         PC21
##  0.484426444 -0.413971027  0.861064042
##          PC22         PC23         PC24
##  0.412112072 -0.169223915 -0.042496895
##          PC25         PC26         PC27
##  0.392156775 -0.810609198 -0.724090154
##          PC28         PC29         PC30
## -0.022842034 -0.267379502  0.223249663
##          PC31         PC32         PC33
##  0.004501899 -0.066925615 -0.420010671
##          PC34         PC35         PC36
##  0.043025063  0.325939735 -0.095872751
##          PC37         PC38
##  0.451061057  0.873973312
```

Y ahora nos quedamos con las primeras cinco columnas correspondientes con las cinco primeras componentes principales como hemos decidido previamente.

```
a = a[,1:5]
```

Podemos representar, como es habitual, las dos primeras componentes. Lo tenemos en la figura 13.3



```
png(paste0(dirTamiFigures,"PCA28.png"))
plot(a[,1],a[,2],xlab="Primera componente",ylab="Segunda componente")
dev.off()

## pdf
## 2
```

Es interesante observar los valores del vector asociado a la primera componente.

```
golub.pca$rotation[,1]

## [1] 0.1715179 0.1690829 0.1650131 0.1726783
## [5] 0.1659431 0.1668800 0.1686381 0.1602445
## [9] 0.1648769 0.1687936 0.1653992 0.1694389
## [13] 0.1629073 0.1661268 0.1647691 0.1720833
## [17] 0.1559293 0.1600159 0.1677201 0.1491867
```

Figura 13.3: Dos primeras componentes principales de los datos golub.

```
## [21] 0.1272725 0.1620961 0.1643597 0.1652554
## [25] 0.1659262 0.1690494 0.1539691 0.1689052
## [29] 0.1541333 0.1516988 0.1691436 0.1682306
## [33] 0.1452419 0.1675335 0.1638384 0.1508645
## [37] 0.1476137 0.1520465
```

Podemos ver que son coeficientes muy parecidos, todos positivos. Básicamente tenemos la media muestral de todos los niveles de expresión en las 38 muestras. La primera componente es básicamente la media sobre las 38 muestras. ¿Y la segunda componente?

```
golub.pca$rotation[,2]

## [1] 0.104190349 -0.036887376 0.069108679
## [4] 0.100701406 0.170952497 0.028349013
## [7] 0.032390592 0.000505933 0.093593873
## [10] 0.023532773 0.075375878 -0.089380731
## [13] 0.233399832 0.077938472 0.237951078
## [16] 0.184071755 0.078196661 0.041608386
## [19] 0.114629249 0.247148154 0.201580365
## [22] -0.014147623 0.037858911 0.210585781
## [25] -0.044465104 0.122286768 0.021439090
## [28] -0.189278987 -0.174593342 -0.243775839
## [31] -0.165316096 -0.150156242 -0.344034501
## [34] -0.157687744 -0.130649728 -0.277921375
## [37] -0.344828851 -0.222765782
```

Si observamos los coeficientes vemos que las primeros 27 valores son positivos y los 11 últimos son negativos. Además no hay una gran diferencia entre los 27 primeros y tampoco entre los 11 últimos. Básicamente (aunque no exactamente) estamos comparando, para cada gen, la media de los niveles de expresión sobre los datos ALL (leucemia linfoblástica aguda) con la media sobre los datos AML (leucemia mieloide aguda).

## 13.4 ¿Muestras o genes?

En la sección anterior hemos realizado un análisis de componentes principales *de las muestras*, esto es, la reducción de dimensión suponía resumir, para gen, su perfil de expresión con menos valores. Sin embargo, también podemos realizar un análisis de componentes principales de los genes. En este caso, las observaciones son las muestras y las variables serían las expresiones sobre los genes. Obviamente podemos realizarlo sobre el conjunto de todos los genes o bien después de aplicar alguna selección previa (bien un filtrado no específico, bien utilizando información sobre los propios genes).

**Ejemplo 13.1 (Datos golub)** *Pretendemos hacer un análisis de componentes principales donde las observaciones son las muestras y para cada muestra tenemos su perfil de expresión sobre todos los genes. Para ello hemos de considerar la matriz traspuesta de la matriz de expresión original con la función t(). A los datos que obtenemos le aplicamos la función prcomp.*

```
tgolub.pca = prcomp(t(golub), scale=FALSE, center=TRUE)
```

*Veamos el resumen del análisis que acabamos de realizar.*

```
summary(tgolub.pca)
```

```
## Importance of components:
##                               PC1      PC2      PC3
## Standard deviation    13.0934 10.17462 9.40357
## Proportion of Variance 0.1645  0.09934 0.08485
## Cumulative Proportion 0.1645  0.26385 0.34870
##                               PC4      PC5      PC6
## Standard deviation    7.9010  6.82616 6.62780
## Proportion of Variance 0.0599  0.04471 0.04215
## Cumulative Proportion 0.4086  0.45332 0.49547
##                               PC7      PC8      PC9
## Standard deviation    6.30435 5.83194 5.79413
## Proportion of Variance 0.03814 0.03264 0.03222
## Cumulative Proportion 0.53361 0.56625 0.59846
##                               PC10     PC11     PC12
## Standard deviation    5.15726 5.01893 4.90719
## Proportion of Variance 0.02552 0.02417 0.02311
## Cumulative Proportion 0.62398 0.64816 0.67126
##                               PC13     PC14     PC15
## Standard deviation    4.72354 4.50857 4.40036
## Proportion of Variance 0.02141 0.01951 0.01858
## Cumulative Proportion 0.69267 0.71218 0.73076
##                               PC16     PC17     PC18
## Standard deviation    4.34750 4.27398 4.12411
## Proportion of Variance 0.01814 0.01753 0.01632
## Cumulative Proportion 0.74890 0.76643 0.78275
##                               PC19     PC20     PC21
## Standard deviation    3.98196 3.94862 3.85795
## Proportion of Variance 0.01522 0.01496 0.01428
## Cumulative Proportion 0.79796 0.81292 0.82721
##                               PC22     PC23     PC24
## Standard deviation    3.77535 3.69767 3.64804
## Proportion of Variance 0.01368 0.01312 0.01277
## Cumulative Proportion 0.84088 0.85400 0.86677
##                               PC25     PC26     PC27
## Standard deviation    3.58436 3.51711 3.49260
## Proportion of Variance 0.01233 0.01187 0.01171
## Cumulative Proportion 0.87910 0.89097 0.90268
##                               PC28     PC29     PC30
## Standard deviation    3.44527 3.37281 3.35604
## Proportion of Variance 0.01139 0.01092 0.01081
## Cumulative Proportion 0.91407 0.92498 0.93579
##                               PC31     PC32     PC33
## Standard deviation    3.26862 3.26192 3.21319
## Proportion of Variance 0.01025 0.01021 0.00991
## Cumulative Proportion 0.94604 0.95625 0.96616
##                               PC34     PC35     PC36
## Standard deviation    3.10703 3.01784 2.95738
## Proportion of Variance 0.00926 0.00874 0.00839
```

```
## Cumulative Proportion  0.97543 0.98416 0.99256
##                               PC37      PC38
## Standard deviation     2.78502 5.815e-15
## Proportion of Variance 0.00744 0.000e+00
## Cumulative Proportion  1.00000 1.000e+00
```

*En principio, la dimensión de mis datos es de 3051 que es el número de genes con los que estamos trabajando. Sin embargo, vemos que solamente considera 38 posibles componentes principales. ¿Por qué? Pues porque es el rango de la matriz de covarianzas.<sup>1</sup>*

*Es interesante representar las componentes principales de las muestras y los genes. En las figuras 13.4 y 13.5 mostramos las dos primeras componentes utilizando la función plotPCA del paquete [R-affycoretools].*

```
library(affycoretools)
png(paste0(dirTamiFigures, "PCA34-a.png"))
affycoretools::plotPCA(golub, legend = FALSE)
dev.off()

## pdf
## 2

png(paste0(dirTamiFigures, "PCA34-b.png"))
affycoretools::plotPCA(t(golub), legend = FALSE)
dev.off()

## pdf
## 2
```

*Cuando realizamos un análisis de los genes sabemos que las observaciones están clasificadas (las muestras son de dos tipos de leucemia). La función plotPCA está preparada para mostrarnos si las componentes nos reproducen los grupos que sabemos que previamente tenemos (aquí no hemos realizado ningún análisis cluster previo). En la figura 13.6 mostramos la misma figura 13.4 pero diferenciando el tipo de muestra (según el tipo de leucemia).*

```
tipo = factor(golub.cl+1, levels = 1:2, labels = c("ALL", "AML"))
png(paste0(dirTamiFigures, "PCA37.png"))
affycoretools::plotPCA(golub, groups = tipo, groupnames = tipo,
dev.off()

## pdf
## 2
```

*Podemos ver cómo la primera componente principal diferencia bastante claramente los dos tipos de leucemia.*

## 13.5 ¿Tipificamos los datos?

Hemos trabajado hasta ahora con los datos centrados (restamos la media muestral a cada variable) pero no hemos tipificado, esto es, no

<sup>1</sup> Es un comentario incomprendible siguiendo el material anterior. Lo indico simplemente.

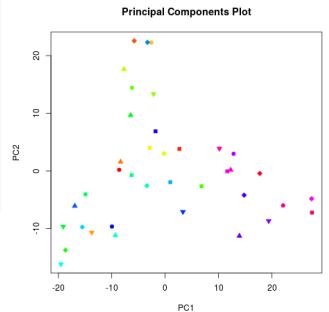


Figura 13.4: Dos primeras componentes principales de los datos golub utilizando como variables las expresiones en los genes.

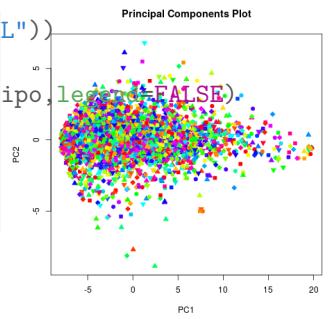
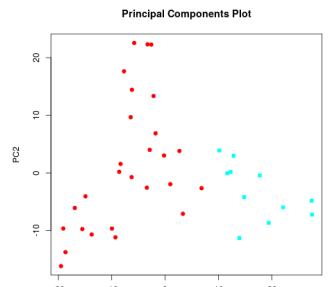


Figura 13.5: Dos primeras componentes principales de los datos golub utilizando como variables las expresiones en las muestras.



hemos dividido cada variable por su desviación estándar. Si los datos tienen una escala similar esta es la opción razonable. Para matrices de expresión esto es así, estamos midiendo lo mismo en distintas muestras y genes y, sobre el papel, la escala es la misma. Pero: ¿y si no es así? En este caso en lugar de trabajar con las variables originales las tipificamos y realizamos el análisis anterior con estos datos.<sup>197</sup>

<sup>197</sup> Realmente si no tipificamos y simplementamos centramos los datos estamos realizando un análisis de componentes principales sobre la matriz de covarianzas. Si tipificamos las variables entonces estamos realizando un análisis de componentes principales sobre la matriz de correlaciones. En § 27.2 definimos matriz de covarianzas y de correlaciones.

Reproducimos (y no mostramos) el análisis de los datos golub pero utilizando datos tipificados. Notemos que la opción fundamental es `scale = TRUE`.

```
golub.pca = prcomp(golub, scale=TRUE, center=TRUE)
summary(golub.pca)
tgolub.pca = prcomp(t(golub), scale=TRUE, center=TRUE)
summary(tgolub.pca)
```

## 13.6 Ejemplos

**Ejemplo 13.2 (GSE20986)** Cargamos los datos.

```
pacman::p_load(Biobase)
data(gse20986, package="tamidatadata")
```

La covariante fenotípica que nos indica el tejido de donde se obtuvo la muestra es

```
pData(gse20986) [, "tissue"]
## [1] iris      retina    retina    iris
## [5] retina    iris      choroides choroides
## [9] choroides huvec    huvec    huvec
## Levels: iris retina choroides huvec
```

En la figura 13.7 vemos representadas las componentes principales de los genes.

```
png(paste0(dirTamiFigures, "PCA44.png"))
affycoretools::plotPCA(gse20986, groups = pData(gse20986) [, "tissue"],
                      groupnames = c("iris", "retina", "choroides", "huvec"))
dev.off()
```

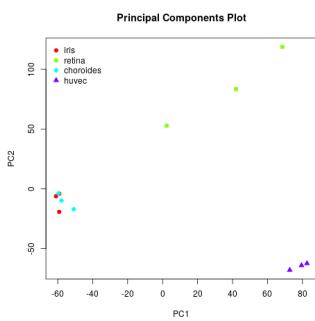


Figura 13.7: Datos GSE20986. Componentes principales de los genes. Se aprecia como el grupo huvec (tejido de vena umbilical) se diferencia claramente de los demás.

Llama la atención la clara diferenciación de las tres muestras correspondientes a huvec. Aparecen dispersas pero separadas de los demás las correspondientes a la retina. Finalmente las muestras de iris y coroides no muestran una diferenciación.

Vemos que son informativas pero: ¿qué proporción de la variación total explican? Realizamos unas componentes principales de las muestras. Notemos que tomamos la matriz de expresión y luego aplicamos la traspuesta ya que de lo contrario las componentes se estarían realizando sobre las muestras.

```
a.pca = prcomp(t(exprs(gse20986)))
summary(a.pca)

## Importance of components:
##                PC1       PC2       PC3
## Standard deviation 64.0229 58.2172 32.70628
## Proportion of Variance 0.3537 0.2924 0.09229
## Cumulative Proportion 0.3537 0.6461 0.73835
##                PC4       PC5       PC6
## Standard deviation 27.38208 24.78294 21.58365
## Proportion of Variance 0.06469 0.05299 0.04019
## Cumulative Proportion 0.80304 0.85604 0.89623
##                PC7       PC8       PC9
## Standard deviation 18.60356 17.9181 15.86741
## Proportion of Variance 0.02986 0.0277 0.02172
## Cumulative Proportion 0.92609 0.9538 0.97551
##                PC10      PC11
## Standard deviation 13.42905 10.17363
## Proportion of Variance 0.01556 0.00893
## Cumulative Proportion 0.99107 1.00000
##                PC12
## Standard deviation 7.927e-14
## Proportion of Variance 0.000e+00
## Cumulative Proportion 1.000e+00
```

Apreciamos que las dos primeras componentes nos explican un 64.61% de la variación total.

En la representación gráfica de la figura 13.7 hemos utilizado la función `affycoretools::plotPCA`. Quizás no es buena costumbre ligarnos tanto a un tipo de dato como el `ExpressionSet`. Podemos utilizar [72, `ggfortify`]. Como antes, hemos de trabajar con la matriz transpuesta y ponemos como una variable adicional de nuestro `data.frame` la variable que indica el tejido.

```
df0 = t(exprs(gse20986))
tissue = pData(gse20986) [, "tissue"]
df = data.frame(tissue, df0)
pacman::p_load(ggfortify)
png(paste0(dirTamiFigures, "PCA46b.png"))
autoplot(prcomp(df [-1]), data=df, colour="tissue")
dev.off()

## pdf
## 2
```

Vemos las componentes en la figura ??

**Ejemplo 13.3 (GSE1397)** Leemos los datos.

```
data(gse1397, package="tamidata")
```

Tenemos los metadatos fenotípicos con

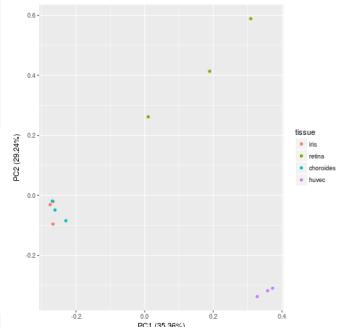


Figura 13.8: Componentes sobre los genes con `[ggfortify]`.

```
pData(gse1397)
```

Vemos que la variable que nos indica la alteración y el tejido son

```
pData(gse1397) [, 'type']

## [1] Euploid Euploid Euploid Euploid Euploid
## [6] Euploid Euploid TS21    TS21    TS21
## [11] TS21    TS21    TS21    TS21
## Levels: Euploid TS21
```

*y*

```
pData(gse1397) [, 'tissue']
```

```
## [1] Cerebrum   Cerebrum   Cerebrum   Cerebrum
## [5] Cerebellum Cerebellum Cerebellum Cerebrum
## [9] Cerebrum   Cerebrum   Cerebrum   Cerebellum
## [13] Cerebellum Cerebellum Cerebrum
## Levels: Cerebrum Cerebellum
```

En la figura 13.9 tenemos el resultado de un análisis de componentes principales considerando los genes como observaciones y diferenciando según el tipo de alteración.

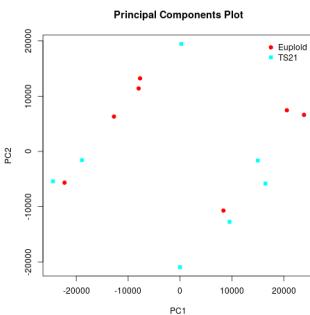


Figura 13.9: Dos primeras componentes principales de datos GSE1397 diferenciando el diagnóstico.

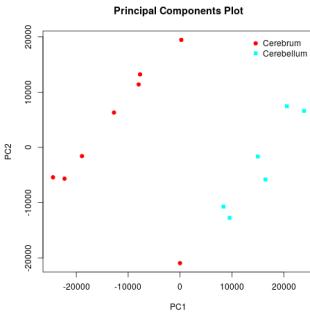


Figura 13.10: Dos primeras componentes principales de datos GSE1397 diferenciando el tejido.

```
png(paste0(dirTamiFigures, "PCA51.png"))
affycoretools::plotPCA(gse1397, groups = pData(gse1397) [, 'type'],
                      groupnames = levels(pData(gse1397) [, 'type']))
dev.off()

## pdf
## 2
```

Las dos primeras componentes vemos como no nos diferencian el tipo de alteración. Sin embargo, cuando consideramos el tejido de donde obtuvimos la muestra se aprecia una mejor diferenciación. De hecho la primera componente diferencia bastante bien los tejidos. Lo podemos ver en la figura 13.10.

```
library(affycoretools)
png(paste0(dirTamiFigures, "PCA53.png"))
affycoretools::plotPCA(gse1397, groups = pData(gse1397) [, 'tissue'],
                      groupnames = levels(pData(gse1397) [, 'tissue']))
dev.off()

## pdf
## 2
```

## 13.7 Componentes principales

En esta sección realizamos una presentación más formal del concepto de componente principal. Su nivel no corresponde con el resto del capítulo y se incluye como material complementario.

Cuando tomamos medidas sobre genes, personas, objetos, empresas, unidades experimentales de un modo genérico, se tiende a recoger el máximo de variables posible. En consecuencia tenemos dimensiones del vector de características  $X$  grandes.

Una opción consiste en sustituir la observación original, de dimensión  $d$ , por  $k$  combinaciones lineales de las mismas. Obviamente pretendemos que  $k$  sea mucho menor que  $d$ . El objetivo es elegir  $k$  de modo que expresen una proporción razonable de la dispersión o variación total cuantificada como la traza de la matriz de covarianza muestral,  $tr(S)$ ,

Sea  $X$  un vector aleatorio de dimensión  $d$  con vector de medias  $\mu$  y matriz de covarianzas  $\Sigma$ . Sea  $T = (t_1, t_2, \dots, t_d)$  (los  $t_i$  indican la  $i$ -ésima columna de la matriz) la matriz ortogonal tal que

$$T'\Sigma T = \Lambda = diag(\lambda_1, \dots, \lambda_d), \quad (13.1)$$

donde  $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d \geq 0$  son los valores propios de la matriz  $\Sigma$ . Sea

$$Y = T'(X - \mu). \quad (13.2)$$

Si denotamos la  $j$ -ésima componente de  $Y$  como  $Y_j$  entonces  $Y_j = t'_j(X - \mu)$  con  $j = 1, \dots, d$ . A la variable  $Y_j$  la llamamos la  $j$ -ésima **componente principal** de  $Y$ . La variable  $Z_j = \frac{Y_j}{\sqrt{\lambda_j}}$  es la  $j$ -ésima **componente principal estandarizada** de  $Y$ .

Estas componentes tienen algunas propiedades de gran interés. Notemos que el vector  $t_j$  tiene longitud unitaria y, por lo tanto,  $Y_j$  no es más que la proyección ortogonal de  $X - \mu$  en la dirección  $t_j$ .

**Proposición 13.1** 1. Las variables  $Y_j$  son incorreladas y  $var(Y_j) = \lambda_j$ .

2. Las variables  $Z_j$  son incorreladas y con varianza unitaria.

Demostración.

En cuanto al apartado primero tenemos que

$$var(Y) = var(T'(X - \mu)) = T'var(X - \mu)T = T'\Sigma T = \Lambda.$$

El segundo apartado es directo a partir del primero.

□

Se verifica el siguiente resultado.

**Teorema 13.1** Las componentes principales  $Y_j = t'_j(X - \mu)$  con  $j = 1, \dots, d$  tienen las siguientes propiedades:

1. Para cualquier vector  $a_1$  de longitud unitaria,  $var(a'_1 X)$  alcanza su valor máximo  $\lambda_1$  cuando  $a_1 = t_1$ .

2. Para cualquier vector  $a_j$  de longitud unitaria tal que  $a'_j t_i = 0$  para  $i = 1, \dots, j-1$ , se tiene que  $var(a'_j X)$  toma su valor máximo  $\lambda_j$  cuando  $a_j = t_j$ .

3.  $\sum_{j=1}^d var(Y_j) = \sum_{j=1}^d var(X_j) = traza(\Sigma)$ .

La versión muestral de las componentes principales la obtenemos sustituyendo en lo anterior  $\mu$  y  $\Sigma$  por  $\bar{X}$  y  $\hat{\Sigma}$  respectivamente. Es importante considerar el estimador de  $\Sigma$  que estamos utilizando (o bien el estimador insesgado donde dividimos por  $n - 1$  o bien el estimador en donde dividimos por  $n$ ).

Si denotamos por  $\hat{\lambda}_1 \geq \dots \geq \hat{\lambda}_d$  los valores propios ordenados de  $\hat{\Sigma}$  y la matriz  $\hat{T} = (\hat{t}_1, \dots, \hat{t}_d)$  es la matriz tal que cada columna es el correspondiente vector propio entonces tenemos las componentes principales muestrales dadas por  $y_j = \hat{T}'(x_i - \bar{x})$ . La nueva matriz de datos viene dada por

$$Y' = (y_1, \dots, y_n) = \hat{T}'(x_1 - \bar{x}, \dots, x_n - \bar{x}) \quad (13.3)$$

Finalmente, si las variables vienen dadas en unidades muy distintas puede ser conveniente sustituir la matriz de covarianzas (poblacional o muestral) por la correspondiente matriz de correlaciones. De hecho, una de los inconvenientes de las componentes principales como un modo de reducir la dimensión de los datos es precisamente que obtenemos resultados distintos si utilizamos las componentes principales obtenidas a partir de la matriz de covarianzas o bien las componentes principales obtenidas a partir de la matriz de correlaciones.

A partir de las  $d$  variables originales podemos obtener hasta  $d$  componentes principales. Sin embargo, hemos dicho que pretendemos reducir la dimensión del vector de datos. La pregunta a responder es: ¿con cuántas componentes nos quedamos?

Supongamos que estamos trabajando con la matriz de covarianzas  $\Sigma$ . Hemos de recordar que  $\text{var}(y_j) = \lambda_j$  y que  $\sum_{j=1}^d \text{var}(x_j) = \sum_{j=1}^d \text{var}(y_j) = \sum_{j=1}^d \lambda_j$ . En consecuencia se suelen considerar los siguientes cocientes

$$\frac{\sum_{j=1}^k \lambda_j}{\sum_{j=1}^d \lambda_j}, \text{ con } k = 1, \dots, d,$$

de modo que, cuando para un cierto valor de  $k$ , estamos próximos a la unidad nos quedamos con ese valor de  $k$ . En la versión muestral trabajaremos o bien con los valores propios de la matriz de covarianzas muestral o la matriz de correlaciones muestrales.

Una referencia muy interesante sobre componentes principales es [1].

.

# Capítulo 14

## Análisis cluster

### 14.1 Introducción

Este tema está dedicado al análisis cluster y su aplicación a datos de expresión de gen. Trataremos lo que en la literatura estadística recibe el nombre de *análisis cluster*<sup>1</sup> o, en mejor castellano, *análisis de conglomerados*. En la literatura de Inteligencia Artificial se utiliza la expresión *clasificación no supervisada*.<sup>2</sup>

¿Qué vamos a clasificar cuando trabajamos con datos de expresión de gen? Nuestra información es la matriz de expresión. Podemos considerar como observaciones a clasificar los genes. En este caso, la observación son los niveles de expresión observados en todas las muestras, el perfil de expresión del gen sobre las muestras analizadas.

También podemos considerar como observaciones a clasificar las muestras. En este caso cada muestra viene descrita por sus expresiones sobre todos los genes. Podemos trabajar con todos los genes o con un subconjunto de genes.

En uno u otro caso tenemos una muestra de observaciones multivariantes de dimensión  $d$ . ¿Qué pretendemos hacer con ellos? Encontrar grupos. Muy breve la respuesta pero: ¿qué son grupos? Imaginemos una imagen aérea fija de un patio de un colegio. En esta imagen los datos son las posiciones de los niños. ¿Se agrupan los niños formando grupos o todos están jugando con todos y los grupos son una consecuencia pasajera del juego (delanteros y defensas que aparecen agrupados en un ataque)?

Parece claro y simple el problema. Sí, lo parece. ¿Qué es un grupo? ¿Cómo defino un grupo? ¿Cuántos grupos distingo en los datos? Estamos viendo el efecto del ruido o realmente hay una estructura debajo que la vemos en un entorno con ruido.

¿Qué quiere decir encontrar grupos? Se trata de clasificar las observaciones en grupos de modo que las observaciones de un mismo grupo sean lo más similares que podamos y que los grupos entre sí sean muy distintos. El número de procedimientos que se han propuesto en la literatura es muy grande. La mayor parte de ellos no tienen un modelo probabilístico debajo, no son procedimientos *basados en modelo*. Son métodos que esencialmente utilizan el concepto de *proximidad*. Valoran de distintas formas lo próximos, lo cercanos que están los puntos,

---

<sup>1</sup> Por cierto que la palabra cluster no existe en castellano

<sup>2</sup> Por tradición, utilizaremos la expresión de análisis cluster. Es la que, personalmente, prefiero.

dentro de un mismo grupo y entre distintos grupos. Es pues, el primer punto a tratar: ¿cómo cuantificamos lo cerca o lejos que están los distintos puntos? En § 14.3 nos ocupamos de este punto. También será necesario, como veremos en el tema, valorar cuando dos conjuntos de puntos son más o menos parecidos, próximos, similares. En la misma sección nos ocupamos de ello. Supongamos que ya hemos clasificado en distintos grupos. ¿Hemos tenido éxito al hacerla? Cuando tenemos un análisis discriminante tenemos una muestra donde sabemos a qué grupo pertenece el individuo y dónde lo hemos clasificado. Esto nos permitía valorar si nuestro procedimiento clasifica bien o no. Aquí no vamos a tener esta referencia que nos da la muestra de entrenamiento. ¿Cómo valorarlo? Un concepto conocido por silueta y debido a Rousseeuw [82] nos va a servir para ello. No es ni tan simple ni tan satisfactorio como en análisis discriminante (como es de esperar si tenemos menos información para trabajar). Lo estudiamos en § 14.8.

Entre los muchos procedimientos de obtener los grupos a partir de los datos, los más utilizados son dos tipos: procedimientos jerárquicos y métodos de particionamiento. De los jerárquicos nos ocupamos en § 14.5. El método de las k-medias y el método de las k-mediodes (el castellano como siempre es muy sufrido pues no existe la palabra) son métodos de particionamiento y los tratamos en § 14.7.

Un texto antiguo pero de un interés enorme en este tema es [130].

## 14.2 Datos

En esta sección presentamos tres ejemplos sencillos de observaciones a agrupar y que utizamos como ilustración simple de los métodos que proponemos.

### 14.2.1 Un ejemplo artificial

Son unos datos muy conocidos, los datos Ruspini. Están en el paquete *cluster* [95].<sup>3</sup>

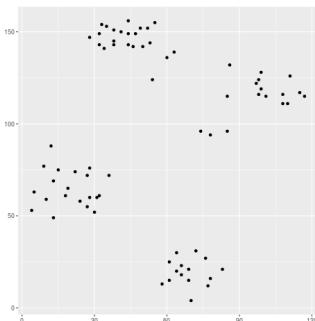
Cargamos el paquete y los datos.

```
pacman::p_load(cluster)
data(ruspini)
```

En la figura 14.1 mostramos estos datos.

```
pacman::p_load(ggplot2)
png(paste0(dirTamiFigures, "Cluster2.png"))
ggplot(ruspini, aes(x=x, y=y)) + geom_point()
dev.off()

## pdf
## 2
```



Son datos bivariantes. Visualmente vemos cómo se agrupan los puntos. Parece (muy) claro que podemos distinguir cuatro grupos.

---

<sup>3</sup>En este tema el paquete *cluster* es, sin duda, el fundamental.

Figura 14.1: Datos ruspini

### 14.2.2 Un ejemplo con muestras

Un ejemplo con los datos *golub*. Empezamos cargando los datos.

```
data(golub, package="multtest")
```

Previamente hemos visto que los valores de expresión de los genes “CCND3 Cyclin D3” y “Zyxin” permiten diferenciar entre ALL y AML. Localicemos las expresiones correspondientes a estos genes.

```
grep("CCND3 Cyclin D3", golub.gnames[, 2])
## [1] 1042

grep("Zyxin", golub.gnames[, 2])
## [1] 2124
```

Los datos aparecen en estas filas. Por lo tanto podemos construir la matriz de datos correspondiente.

```
cz.data = data.frame(golub[1042,], golub[2124,])
names(cz.data) = c("CCND3_Cyclin_D3", "Zyxin")
```

Este será un segundo ejemplo para analizar. Los datos aparecen en la figura 14.2.

```
png(paste0(dirTamiFigures, "Cluster6b.png"))
ggplot(cz.data, aes(x=CCND3_Cyclin_D3, y=Zyxin)) + geom_point()
dev.off()

## pdf
## 2
```

En este caso las observaciones corresponden a las muestras y las variables son los niveles de expresión de dos genes. ¿Hay grupos? Esto no son datos artificiales como los de Ruspini y ya no es tan claro.

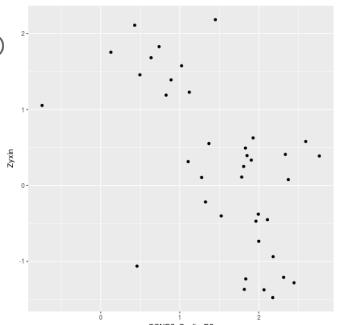


Figura 14.2: Datos cz.data

### 14.2.3 Un ejemplo con genes

Consideraremos (otra vez) los datos *golub*.

```
data(golub, package="multtest")
```

Para cada gen vamos a considerar la expresión media sobre los casos ALL

```
x = apply(golub[, golub.cl == 0], 1, mean)
```

y la media sobre los casos AML.

```
y = apply(golub[, golub.cl == 1], 1, mean)
```

Y vamos a limitarnos a un conjunto de genes. En concreto vamos a considerar los genes que se relacionan con el término biológico “Cyclin”. Determinamos qué filas ocupan.

```
sel = grep("Cyclin",golub.gnames[,2])
```

La función *grep* busca la palabra “Cyclin” en cada uno de los nombres que le pasamos. Y los nombres de estos genes son

```
golub.gnames[sel,2]

## [1] "CCND2 Cyclin D2"
## [2] "CDK2 Cyclin-dependent kinase 2"
## [3] "CCND3 Cyclin D3"
## [4] "CDKN1A Cyclin-dependent kinase inhibitor 1A (p21, Cip1)"
## [5] "CCNH Cyclin H"
## [6] "Cyclin-dependent kinase 4 (CDK4) gene"
## [7] "Cyclin G2 mRNA"
## [8] "Cyclin A1 mRNA"
## [9] "Cyclin-selective ubiquitin carrier protein mRNA"
## [10] "CDK6 Cyclin-dependent kinase 6"
## [11] "Cyclin G1 mRNA"
## [12] "CCNF Cyclin F"
```

Ahora nos limitamos a considerar las expresiones medias para los casos ALL y los casos AML para estos genes.

```
cyclin.data = data.frame(x[sel],y[sel])
names(cyclin.data) = c("meanALL","meanAML")
```

En la figura 14.3 mostramos los datos.

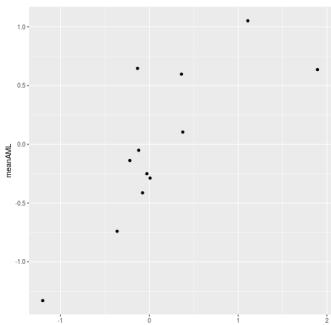


Figura 14.3: Datos cyclin.data.

```
png(paste0(dirTamiFigures,"Cluster14.png"))
ggplot(cyclin.data,aes(x=meanALL,y=meanAML))+geom_point()
dev.off()

## pdf
## 2
```

No parece, al menos visualmente, que tengamos muchos grupos. De eso se trata. De saber si hay grupos y encontrarlos.

## 14.3 Disimilaridades

En lo que sigue denotaremos los datos a agrupar con  $x_i$  con  $i = 1, \dots, n$  siendo  $x_i \in \mathbb{R}^d$ . Todos los ejemplos que hemos visto en la sección anterior eran (con objeto de poder representarlos) datos bidimensionales,  $d = 2$ .

### 14.3.1 Distancias euclídea y de Manhattan

Empezamos tratando el problema de cuantificar el grado de proximidad, de similaridad entre dos puntos en el espacio de dimensión  $d$ . Tradicionalmente este tema en Matemáticas se ha formalizado a través del concepto de *distancia* o *métrica*. Una métrica es una función que a cada par de puntos  $x, y \in \mathbb{R}^d$  le asocia un valor positivo de modo que cuando mayor es más *distantes* son, más alejados están. Como siempre la formalización matemática de un concepto intuitivo ha de ser prudente

y pedir que se verifiquen ciertos axiomas que resulten razonables y generalmente admisibles. Las distancias más utilizadas en análisis cluster son la distancia euclídea y la distancia de Manhattan. Para dos vectores  $x$  e  $y$  (en  $\mathbb{R}^d$ ) entonces la distancia euclídea se define como

$$d_E(x, y) = \sqrt{\sum_{k=1}^d (x_k - y_k)^2}, \quad (14.1)$$

con  $x, y \in \mathbb{R}^d$ . La distancia de Manhattan viene dada por

$$d_M(x, y) = \sum_{k=1}^d |x_k - y_k|. \quad (14.2)$$

Las distancias euclídea y de Manhattan son adecuadas cuando trabajamos con variables continuas y que además estén en una misma escala. Son dos ejemplos de distancias o métricas.<sup>4</sup> En lo que sigue en lugar de distancia utilizaremos el término *medida de disimilaridad* indicando que no necesariamente ha de ser distancia. Es una visión menos restrictiva a la hora de definir que dos genes están cerca.

En lo que sigue agrupamos bien genes bien muestras. Si esta clasificación no supervisada la basamos en una distancia lo que buscamos es encontrar grupos de genes que tienen perfiles de expresión similares. En definitiva que se comportan de un modo parecido en el experimento para todas las muestras. Si agrupamos muestras lo que buscamos son muestras que, para los genes considerados, tienen unas expresiones similares. Por ejemplo, si las muestras corresponden a un cierto tipo de cáncer entonces los grupos que podemos encontrar serían subtipos de cáncer. En resumen genes o muestras los consideramos similares cuando sus expresiones lo son. Además interpretamos la similaridad como una métrica. El concepto de distancia o métrica (del cual hemos visto dos ejemplos) es limitado en muchas aplicaciones. En ocasiones podemos interpretar que dos genes están *próximos* cuando sus expresiones están *relacionadas* de alguna forma, en resumen, cuando están *corregulados*. En este caso podemos usar para definir la disimilaridad a partir de una cuantificación de la dependencia entre los genes.

### 14.3.2 Disimilaridades utilizando coeficientes de correlación

Consideremos dos genes con perfiles  $x = (x_1, \dots, x_n)$  e  $y = (y_1, \dots, y_n)$ . En primer lugar pretendemos cuantificar el grado de dependencia entre los pares  $(x_i, y_i)$  para  $i = 1, \dots, n$ . Si nos limitamos a la posible dependencia lineal entre estos pares entonces el coeficiente de correlación de Pearson es la medida genéricamente aceptada. En § 27.1 tenemos su

---

<sup>4</sup>En concreto la función  $d$  definida en el espacio producto  $\mathbb{R}^d \times \mathbb{R}^d$  se dice que es una métrica si verifica:

**No negativa**  $d(x, y) \geq 0$ .

**Un punto dista 0 de sí mismo**  $d(x, x) = 0$ .

**Simetría**  $d(x, y) = d(y, x)$ .

**Desigualdad triangular**  $d(x, z) \leq d(x, y) + d(y, z)$ , para todo  $x, y, z \in \mathbb{R}^d$ .

definición y algunas propiedades básicas. Se define como

$$r_{x,y} = \frac{\sum_{i=1}^n (x_i - \bar{x}_n)(y_i - \bar{y}_n)}{\sqrt{\sum_{i=1}^n (x_i - \bar{x}_n)^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y}_n)^2}}.$$

Lo fundamental es que varía entre -1 y 1. Cuando toma valores próximos a cero indica independencia entre los pares y cuando toma valor próximo a 1 o a -1 indica una dependencia lineal extrema. De hecho si vale 1 entonces existen constantes  $a$  y  $b$  tales que  $y_1 = ax_1 + b$  con  $a > 0$ . Si vale -1 entonces existen constantes  $a$  y  $b$  tales que  $y_1 = ax_1 + b$  con  $a < 0$ .

Supongamos que cuando evaluamos si dos genes están *próximos* o están *alejados* lo que estamos intentando expresar es que están coregulados. Esto indicaría que una mayor expresión de uno en una muestra se corresponde con una mayor expresión en la misma muestra. Si, además, la asociación es de tipo lineal esto significaría que el perfil de expresión de gen se podría obtener como una función lineal de la expresión del otro gen. En resumen su coeficiente de correlación de Pearson ([§ 27.1](#)) debiera de ser próximo a uno. También sería razonable entender como próximos aquellos genes donde cuando uno tiene un valor alto de expresión el otro lo tiene bajo. Es decir, finalmente tienen una fuerte relación lineal pero de tipo inverso.

A partir del coeficiente de correlación podemos definir distintas medidas de disimilaridad. Algunas opciones que se han propuesto son las siguientes:

1.  $d(x, y) = 1 - r_{x,y}$ ,
2.  $d(x, y) = (1 - r_{x,y})/2$ ,
3.  $d(x, y) = 1 - |r_{x,y}|$ ,
4.  $d(x, y) = \sqrt{1 - r_{x,y}^2}$ .

<sup>5</sup> De un modo similar podemos definir disimilaridades sustituyendo el coeficiente de correlación de Pearson con el coeficiente de correlación de Spearman.

### 14.3.3 Matriz de disimilaridades

Tenemos una serie de conjunto de puntos a agrupar:  $\{x_1, \dots, x_n\}$ . Una vez elegida una disimilaridad construiremos una matriz que en la fila  $i$ , columna  $j$  tiene la disimilaridad entre el punto  $x_i$  y el punto  $x_j$ . Denotaremos indistintamente  $d(x_i, x_j)$  o  $d_{ij}$ . Por tanto, la matriz de disimilaridades es

$$D = [d(x_i, x_j)]_{i,j=1,\dots,n} = [d_{ij}]_{i,j=1,\dots,n}.$$

¿Cómo calculamos las matrices de disimilaridades que acabamos de proponer? La función `dist` de [\[113\]](#) es una buena opción cuando utilizamos las distancias euclídea o de Manhattan.

---

<sup>5</sup>En <http://research.stowers-institute.org/efg/R/Visualization/cor-cluster/> podemos encontrar un breve estudio comparativo del funcionamiento de estas disimilaridades en clustering jerárquico.

Un detalle es importante. Si la disimilaridad que estamos considerando es simétrica, esto es, si la disimilaridad  $d$  verifica

$$d(x_i, x_j) = d(x_j, x_i),$$

entonces la matriz de disimilaridades

$$D = [d(x_i, x_j)]_{i,j=1,\dots,n}$$

es simétrica y estamos almacenando información redundante. Además hemos de considerar que

$$d(x_i, x_i) = 0,$$

es decir, la disimilaridad de un perfil de expresión consigo mismo es nula y por lo tanto, en la matriz de disimilaridades, la diagonal de la matriz es nula. Sabemos que trabajamos habitualmente con un número de genes muy grande. La matriz de disimilaridades total sería de tamaño  $N \times N$  y por ello, es más que conveniente eliminar redundancias. Veamos cómo esto lo hace de un modo automático la función `dist`.

Vamos a calcular la matriz de distancias con la función `dist` para los datos `cz.data` propuestos en § 14.2.2.

```
cz.dist.euclidea = dist(cz.data, method="euclidean")
```

¿De qué clase es `cz.dist.euclidea`.

```
class(cz.dist.euclidea)
## [1] "dist"
```

La distancia de Manhattan la podemos obtener con

```
cz.dist.manhattan = dist(cz.data, method="manhattan")
```

```
data(golub, package="multtest")
golub.cor1 = 1-cor(t(golub))
golub.cor2 = (1-cor(t(golub)))/2
golub.cor3 = 1-abs(cor(t(golub)))
golub.cor4 = sqrt(1-cor(t(golub))^2)
```

En este caso `cz.dist` sí es una matriz completa (con todas sus filas y columnas). La vamos a transformar a un objeto de clase `dist`.

```
golub.cor1 = as.dist(golub.cor1)
golub.cor2 = as.dist(golub.cor2)
golub.cor3 = as.dist(golub.cor3)
golub.cor4 = as.dist(golub.cor4)
```

Con cada una de estas matrices podemos cuantificar la disimilaridad que hay entre los elementos originales de la muestra. Algunos de los procedimientos de agrupamiento que vamos a considerar en lo que sigue no necesitan conocer los datos originales. Pueden aplicarse con solo conocer esta matriz de disimilaridades. Otros no. Otros utilizan los datos a lo largo de las distintas etapas de aplicación del procedimiento.

## 14.4 Disimilaridades entre grupos de observaciones

En algunos procedimientos de agrupamiento (en particular, los jerárquicos) vamos a necesitar calcular disimilaridades entre conjuntos disjuntos de las observaciones originales. Estas disimilaridades las podemos calcular a partir de las disimilaridades originales entre puntos. Supongamos que tenemos un banco de datos con  $n$  individuos cuyos índices son  $\{1, \dots, n\}$ . Sean  $A$  y  $B$  dos subconjuntos disjuntos del conjunto de índices de la muestra  $\{1, \dots, n\}$ , esto es, dos subconjuntos de observaciones disjuntas. ¿Cómo podemos definir una disimilaridad entre  $A$  y  $B$  partiendo de las disimilaridades entre los datos individuales? Se han propuesto muchos procedimientos. Si denotamos la disimilaridad entre  $A$  y  $B$  como  $d(A, B)$  entonces las disimilaridades más habitualmente utilizadas son las siguientes:

**Enlace simple** La disimilaridad entre los dos grupos es el mínimo de las disimilaridades entre las observaciones de uno y de otro. Tomamos la disimilaridad de los objetos que más se parecen en uno y otro grupo.

$$d(A, B) = \min_{a \in A, b \in B} d(a, b)$$

**Enlace completo** Ahora tomamos como disimilaridad entre los grupos como el máximo de las disimilaridades, en definitiva, la disimilaridad entre los objetos más alejados o más distintos.

$$d(A, B) = \max_{a \in A, b \in B} d(a, b)$$

**Promedio** La disimilaridad es el promedio de las disimilaridades entre todos los posibles pares.

$$d(A, B) = \frac{1}{|A| \times |B|} \sum_{a \in A, b \in B} d(a, b)$$

donde  $|A|$  es el cardinal del conjunto  $A$ .

Es importante notar que solamente necesitamos conocer las disimilaridades entre los individuos para poder calcular las disimilaridades entre grupos de individuos.

En la siguiente sección nos vamos a ocupar de los métodos jerárquicos en los cuales es fundamental el procedimiento que elegimos para calcular distancias entre grupos.

## 14.5 Cluster jerárquico

La idea de estos procedimientos es construir una jerarquía de particiones del conjunto de índices.

Sea  $\{1, \dots, n\}$  el conjunto de índices que indexan las distintas observaciones. Supongamos que  $\{C_1, \dots, C_r\}$  es una partición de este conjunto de índices:

- $C_i \subset \{1, \dots, n\}$ ; son disjuntos dos a dos,  $C_i \cap C_j = \emptyset$  si  $i \neq j$  con  $i, j = 1, \dots, n$  y

$$\bullet \cup_{i=1}^r C_i = \{1, \dots, n\}.$$

Dada una partición del conjunto de índices podemos calcular la matriz  $r \times r$  que en la posición  $(i, j)$  tiene la disimilaridad entre el conjunto  $C_i$  y  $C_j$ ,  $d(C_i, C_j)$  según alguno de los procedimientos antes indicados.

Veamos los procedimientos jerárquicos aglomerativos. En estos procedimientos vamos a iniciar el agrupamiento con la partición:  $C_i = \{i\}$  con  $i = 1, \dots, n$ , es decir, cada grupo es un individuo. En cada iteración vamos agrupando el par de conjuntos (elementos de la partición que tengamos en esa iteración) que estén *más próximos* según la disimilaridad entre grupos que estemos utilizando. El proceso continúa hasta que tengamos un único grupo.

Un esquema algorítmico del procedimiento indicado puede ser el siguiente:

**Paso 0** Tenemos grupos unitarios formados por cada una de las observaciones. Tenemos pues una partición inicial  $C_i = \{i\}$  con  $i = 1, \dots, n$ . En un principio, cada dato es un grupo.

**Paso 1** Calculamos las disimilaridades entre los elementos de la partición. Para ello utilizamos cualquiera de los procedimientos antes indicados.

**Paso 2** Agrupamos los dos conjuntos de la partición más próximos y dejamos los demás conjuntos igual. Tenemos ahora  $C_i$  con  $i = 1, \dots, k$ .

**Paso 3** Si tenemos un solo conjunto en la partición paramos el procedimiento.

**Paso 4** Volvemos al paso 1.

Hay una representación gráfica muy utilizada para describir los resultados de un cluster jerárquico aglomerativo como el que acabamos de describir. Esta representación tiene el nombre de **dendograma**. En el dendograma se va mostrando a qué valor de la medida de disimilaridad se produce la unión de los grupos y simultáneamente qué grupos se están uniendo para esa disimilaridad. También nos permite una valoración rápida de cuántos grupos puede haber en el banco de datos. Simplemente trazando una linea horizontal a la altura en que tengamos el número de grupos que *pensamos* que puede haber.

**Ejemplo 14.1 (ruspini)** Consideremos los datos ruspini. Aplicaremos un cluster jerárquico aglomerativo utilizando como disimilaridad entre grupos el promedio de las disimilaridades y como medida de disimilaridad la distancia euclídea.

```
pacman::p_load(cluster)
ruspini.ag = agnes(ruspini, metric = "euclidean", method="average")
```

Una opción alternativa con base::hclust al código anterior es

```
dd = dist(ruspini, method="euclidian")
ruspini.ag = hclust(dd, method="average")
```

Representamos el dendograma en la figura 14.4.

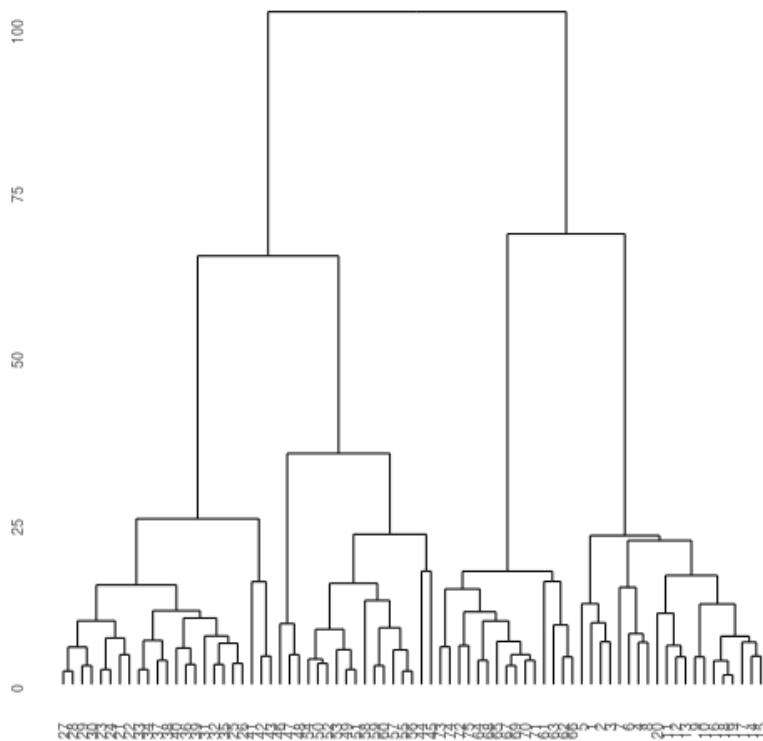


Figura 14.4: Dendograma para un cluster jerárquico aplicado a los datos ruspini con métrica euclídea y el promedio para la disimilaridad entre grupos.

```
png(paste0(dirTamiFigures, "Cluster23.png"))
ggdendro::ggdendrogram(ruspini.ag)
dev.off()

## pdf
## 2
```

*Supongamos que decidimos quedarnos con cuatro grupos. Las clasificaciones de los datos son las siguientes donde la etiqueta que se asigna a cada grupo es arbitraria.*

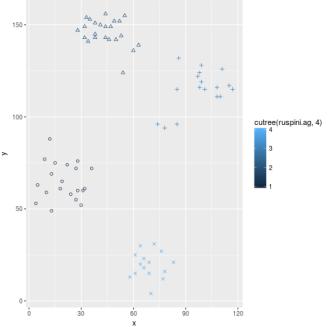
```
cutree(ruspini.ag, 4)

##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
##  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1  1
## 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
##  1  1  1  1  2  2  2  2  2  2  2  2  2  2  2  2
## 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
##  2  2  2  2  2  2  2  2  2  2  2  3  3  3  3  3
## 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64
##  3  3  3  3  3  3  3  3  3  3  3  3  4  4  4  4
## 65 66 67 68 69 70 71 72 73 74 75
##  4  4  4  4  4  4  4  4  4  4  4
```

*Y ahora podemos representar los datos de modo que ponemos en un mismo color los datos que hemos clasificado en un grupo. Lo mostramos en la figura 14.5.*

```
png(paste0(dirTamiFigures, "Cluster26.png"))
p = ggplot(ruspinib, aes(x=x, y=y, color=cutree(ruspinib.ag,4)))
p + geom_point(shape=cutree(ruspinib.ag,4))
dev.off()

## pdf
## 2
```



**Ejemplo 14.2 (ez.data)** Apliquemos un cluster jerárquico aglomerativo utilizando como disimilitud entre grupos el enlace simple y como distancia la de Manhattan.

```
##cz.ag = agnes(cz.data,metric = "euclidean",method="single")
dd = dist(cz.data,method="euclidian")
cz.ag = hclust(dd,method="single")
```

Figura 14.5: Clasificación obtenida para los datos ruspini utilizando un cluster jerárquico con distancia euclídea, promedio entre grupos y cuatro grupos.

*La figura 14.6 muestra el dendograma que obtenemos.*

```
pacman::p_load(ggdendro)
png(paste0(dirTamiFigures, "Cluster29.png"))
ggdendro::ggdendrogram(cz.ag , rotate = FALSE, size = 2)
dev.off()

## pdf
## 2
```

*Supongamos que nos quedamos cinco grupos. Las clasificaciones son:*

```
cutree(cz.ag,5)  
## [1] 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1  
## [24] 1 1 1 1 3 4 3 3 3 3 3 3 5 3 3 3 3
```

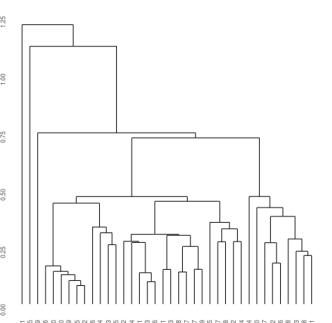


Figura 14.6: Dendograma para datos `ez.data` utilizando distancia euclídea y enlace simple.

*En la figura 14.7 tenemos los resultados de la clasificación.*

```
png(paste0(dirTamiFigures, "Cluster32.png"))
p = ggplot(cz.data, aes(x=CCND3_Cyclin_D3, y=Zyxin, color=cutr
p + geom_point(shape=cutree(cz.ag,5))
dev.off()

## pdf
## ?
```

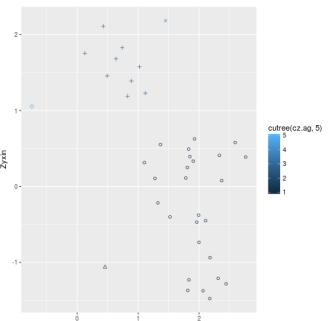


Figura 14.7: Clasificación obtenida para los datos ez.data utilizando un cluster jerárquico cuando cortamos el arbol en cinco grupos. Se utilizó distancia euclídea y enlace simple.

## 14.6 Distancia cofenética

Una vez hemos construido un clustering jerárquico podemos asociarle una distancia conocida como *distancia cofenética*. Dadas dos observaciones (en nuestro caso dos perfiles de expresión) consideraremos el

momento en que ambas observaciones han sido agrupadas en un mismo grupo. La distancia cofenética entre estas observaciones es la distancia entre los dos grupos a los que previamente pertenecían estas observaciones (una en cada uno de ellos) en el momento en que se unen. Se calcula con la función `cophenetic` del paquete [113, stats].

**Ejemplo 14.3 (Distancia cofenética con `cyclin.ag`)** *Continuamos con el jerárquico aglomerativo aplicado a los datos `cyclin.data`. Repetimos el análisis cluster calculando previamente la matriz de disimilaridades. Empezamos calculando la matriz de distancias.*

```
cyclin.manhattan = daisy(cyclin.data, metric = "manhattan")
```

*Y ahora el cluster jerárquico (observemos que no le damos los datos sino que le pasamos directamente la matriz de disimilaridades).*

```
##cyclin.ag = agnes(cyclin.manhattan, diss = TRUE)
cyclin.ag = hclust(cyclin.manhattan, method="average")
```

*Calculamos la distancia cofenética asociada al cluster jerárquico que acabamos de realizar.*

```
cyclin.cofe = cophenetic(cyclin.ag)
```

*Y determinamos el coeficiente de correlación de Pearson.*

```
cor(cyclin.manhattan,cyclin.cofe)
```

```
## [1] 0.8501727
```

*El coeficiente de correlación observado 0.8501727 es un valor grande. Tenemos pues una buena descripción del modo en que los datos se agrupan.*

Las distancias cofenéticas tienen sus problemas. Es una manera *natural* de medir disimilaridad entre los datos a partir del modo en que estos van agrupándose. Sin embargo, notemos que muchos pares de observaciones tienen la misma disimilaridad. Esto es lógico, cuando unimos dos grupos grandes la distancia cofenética entre cada observación de un grupo y del otro grupo tienen la misma distancia cofenética. Se suele calcular el coeficiente de correlación entre la matriz de disimilaridad original y la distancia cofenética. Si el valor es grande indica que el dendograma es una buena descripción de los datos.

Un estudio mucho más amplio de lo visto en esta sección puede verse en [130, capítulo 5].

## 14.7 Métodos de particionamiento

Suponemos ahora que tenemos una idea de cuántos grupos hay. Posiblemente hemos realizado un análisis jerárquico previo con todos los datos o, si eran muchos, con una selección aleatoria de los datos. Tenemos pues una idea de cuántos grupos tendremos que considerar. Obviamente podemos luego evaluar los resultados modificando el número de grupos. En principio, vamos a suponer que fijamos el número de grupos a considerar. Suponemos pues que *sabemos* el número de grupos y lo denotamos por  $k$ .

### 14.7.1 Método de las k-medias

El primer procedimiento que vamos a ver es el método de las *k-medias* (que por alguna extraña razón en la literatura de Inteligencia Artificial se le llama de las *c-medias* lo que demuestra que cada persona copia a sus amigos o, simplemente, conocidos). Supongamos que tenemos  $C_1, \dots, C_k$  una partición de  $\{1, \dots, n\}$ . Un modo bastante natural de valorar la calidad del agrupamiento que la partición nos indica sería simplemente considerar la siguiente función.

$$\sum_{i=1}^k \sum_{j \in C_i} d_E(x_j, \bar{x}_{C_i})^2, \quad (14.3)$$

donde  $d_E$  denota aquí la distancia euclídea y

$$\bar{x}_{C_i} = \frac{1}{|C_i|} \sum_{j \in C_i} x_j, \quad (14.4)$$

es el vector de medias del grupo cuyos índices están en  $C_i$ . Una partición será tanto mejor cuanto menor sea el valor de la función dada en 14.3. El procedimiento de agrupamiento de las k-medias simplemente se basa en elegir como partición de los datos aquella que nos da el *mínimo* de la función objetivo considerada en ecuación 14.3. Notemos que en muchos textos se hablan del algoritmo de las k-medias y se identifica con un procedimiento concreto para encontrar el mínimo de la función. Aquí entendemos el procedimiento como la minimización de la función objetivo. De hecho, R ofrece hasta cuatro posibles procedimientos de los muchos que cabe proponer. Hay que diferenciar claramente el procedimiento del método de aplicación del mismo, del método de obtención de dicho mínimo.

Es importante darnos cuenta de que el procedimiento que acabamos de ver está basado en la utilización de la distancia euclídea y en que, dado un grupo, podemos calcular el vector de medias y esto solo lo podemos hacer si todas las variables son cuantitativas.

Vamos a aplicar el método de las k-medias en los tres ejemplos utilizando el número de grupos que hemos utilizado previamente.

**Ejemplo 14.4 (ruspini)** *Aplicamos el k-medias.*

```
ruspini.km = kmeans(ruspini, 4)
```

*La clasificación viene dada por*

```
ruspini.km$cluster
##  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
## 2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
## 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32
## 2  2  2  2  4  4  4  4  4  4  4  4  4  4  4  4
## 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48
## 4  4  4  4  1  1  4  4  1  1  1  3  3  3  3  3
## 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64
## 3  3  3  3  3  3  3  3  3  3  3  3  2  2  2  2
## 65 66 67 68 69 70 71 72 73 74 75
## 2  2  2  2  2  2  2  2  2  2  2  2  2  2  2  2
```

*La clasificación obtenida se corresponde con la que vemos en la figura 14.5.*

**Ejemplo 14.5 (Método de las k-medias y datos ez.data)** *Empezamos con el k-medias.*

```
cz.km = kmeans(cz.data, 2)
```

*Las clasificaciones son*

```
cz.km$cluster
## [1] 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2 2
## [24] 2 2 2 2 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
```

*Y representamos los resultados.*

```
pacman:::p_load(factoextra)
png(paste0(dirTamiFigures, "Cluster50.png"))
fviz_cluster(cz.km, cz.data, ellipse.type = "norm")
dev.off()
## pdf
## 2
```

*Supongamos que probamos con tres grupos.*

```
cz.km = kmeans(cz.data, 3)
png(paste0(dirTamiFigures, "Cluster52.png"))
fviz_cluster(cz.km, cz.data, ellipse.type = "norm")
dev.off()
## pdf
## 2
```

## 14.7.2 Particionamiento alrededor de los medioides

¿Y si no podemos calcular el vector de medias? ¿Y si no tiene sentido calcular el vector de medias? ¿Cómo promediar dos configuraciones de puntos distintas? ¿Cómo promediamos dos formas distintas descritas numéricamente? Cuando el concepto de promedio aritmético no tiene sentido podemos generalizar el procedimiento anterior y hablar de *k-medioides*.

La idea ahora es sustituir esos centros calculados como vectores de medias de los individuos de un mismo grupo por individuos bien centrados, por individuos típicos, que sustituyan a las medias.

Supongamos que tomamos  $k$  individuos de la muestra que denotamos por  $m_i$  con  $i = 1, \dots, k$ . Particionamos la muestra en  $k$  grupos de modo que el grupo  $C_i$  está formado por los individuos más próximos a  $m_i$  que a cualquier otro  $m_j$  con  $j \neq i$ ,

$$C_i = \{l : d(x_l, x_i) = \min_{j \neq i} d(x_l, x_j)\}.$$

Consideraremos la siguiente cantidad:

$$\sum_{i=1}^k \sum_{j \in C_i} d(x_j, x_{m_i}). \quad (14.5)$$

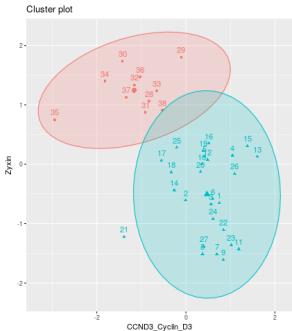


Figura 14.8: Clasificación obtenida utilizando el método de las k-medias aplicada a los datos ez.data.

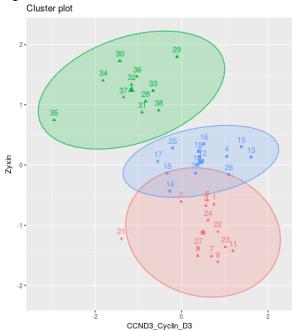


Figura 14.9: Clasificación obtenida utilizando el método de las k-medias aplicada a los datos ez.data con tres grupos.

En el método de particionamiento alrededor de los mediodes nos planteamos encontrar las observaciones  $m_1, \dots, m_k$  que minimizan el valor dado en 14.5.

**Ejemplo 14.6 (ruspini)** *Aplicamos PAM.*

```
ruspini.pam = pam(ruspini, 4)
```

*La clasificación obtenida corresponde con la que mostramos en la figura 14.5. Con los datos ruspini cualquiera de los métodos que hemos utilizado cuando decidimos quedarnos con 4 grupos nos da los mismos resultados de clasificación.*

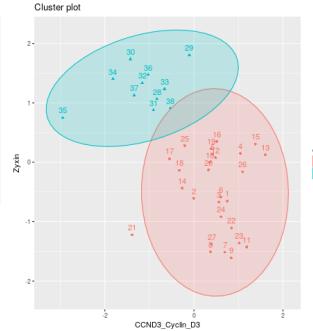
**Ejemplo 14.7 (ez.data)** *Empezamos con dos grupos.*

```
cz.pam = pam(cz.data, 2)
```

*Y representamos los resultados de la clasificación en la figura 14.10*

```
png(paste0(dirTamiFigures, "Cluster58.png"))
fviz_cluster(cz.pam, cz.data, ellipse.type = "norm")
dev.off()

## pdf
## 2
```



## 14.8 Silueta

Veamos cómo se construye la silueta. Para la observación  $i$  y el grupo  $C$  consideramos

$$\bar{d}(i, C) = \frac{1}{|C|} \sum_{j \in C} d(x_i, x_j),$$

la disimilaridad media  $i$  con los elementos del grupo  $C$ . Para cada observación  $i$ , sea  $A$  el cluster al cual lo ha asignado el procedimiento cluster que empleamos y calculamos  $a(i)$  la disimilaridad media de  $i$  con todos los demás individuos del grupo  $A$ ,  $a(i) = \bar{d}(i, A)$ . Obviamente estamos asumiendo que  $A$  contiene al menos otro objeto. Consideremos  $\bar{d}(i, C)$  para todos los grupos  $C \neq A$  y seleccionemos el que tiene el mínimo valor:

$$b(i) = \min_{C \neq A} \bar{d}(i, C).$$

El grupo  $B$  donde se alcanza este mínimo, es decir,  $\bar{d}(i, B) = b(i)$  se le llama vecino del objeto  $i$ .<sup>6</sup> Definimos  $s(i)$  como

$$s(i) = 1 - \frac{a(i)}{b(i)} \text{ si } a(i) < b(i), \quad (14.6)$$

$$= 0 \text{ si } a(i) = b(i), \quad (14.7)$$

$$= \frac{b(i)}{a(i)} - 1 \text{ si } a(i) > b(i). \quad (14.8)$$

Figura 14.10: Clasificación de los datos ez.data con método PAM y dos grupos.

<sup>6</sup>No parece un nombre inadecuado.

Tabla 14.1: Silueta media y estructura en un conjunto de datos

SC	Interpretación
0.71 – 1.00	Fuerte estructura
0.51 – 0.70	Estructura razonable
0.26 – 0.50	Estructura débil. Probar otros métodos
$\leq 0.25$	No se encuentra estructura

Esto se puede expresar en una única ecuación como

$$s(i) = \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}.$$

En el caso en que el grupo  $A$  contenga un único objeto no está muy claro cómo definir  $a(i)$ . Tomaremos  $s(i) = 0$  que es una elección arbitraria. Se comprueba con facilidad que  $-1 \leq s(i) \leq 1$  para cualquier objeto  $i$ .

Para interpretar el significado de  $s(i)$  es bueno ver los valores extremos. Si  $s(i)$  es próximo a uno significa que  $a(i)$  es mucho menor que  $b(i)$  o lo que es lo mismo, que el objeto  $i$  está bien clasificado pues la disimilaridad con los de su propio grupo es mucho menor que la disimilaridad con los del grupo más próximo que no es el suyo. Un valor próximo a cero significa que  $a(i)$  y  $b(i)$  son similares y no tenemos muy claro si clasificarlo en  $A$  o en  $B$ . Finalmente un valor de  $s(i)$  próximo a  $-1$  significa que  $a(i)$  es claramente mayor que  $b(i)$ . Su disimilaridad media con  $B$  es menor que la que tiene con  $A$ . Estaría mejor clasificado en  $B$  que en  $A$ . No está bien clasificado.

Los valores de  $s(i)$  aparecerán representados para cada cluster en orden decreciente. Para cada objeto se representa una barra horizontal con longitud proporcional al valor  $s(i)$ . Una buena separación entre grupos o cluster viene indicada por unos valores positivos grandes de  $s(i)$ . Además de la representación gráfica se proporciona un análisis descriptivo. En concreto la media de los valores de la silueta dentro de cada cluster y la media de la silueta para todo el conjunto de datos. La clasificación será tanto mejor cuanto mayor sean estos valores medios. De hecho, se puede decidir el número de grupos en función del valor medio de las silueta sobre toda la muestra. Vamos probando distintos números de grupos y nos quedamos con el número que nos da la silueta media máxima.

¿Cuándo podemos decir que hay estructura de grupos en los datos que estamos analizando? Experiencias con datos sugieren la tabla 14.1.

**Ejemplo 14.8 (ruspini)** Veamos el resumen de la silueta.

```
ruspini.pam = pam(ruspini, 4)
summary(silhouette(ruspini.pam))

## Silhouette of 75 units in 4 clusters from pam(x = ruspini, k = 4) :
## Cluster sizes and average silhouette widths:
##      20      23      17      15
## 0.7262347 0.7548344 0.6691154 0.8042285
## Individual silhouette widths:
##   Min. 1st Qu. Median Mean 3rd Qu. Max.
## 0.4196 0.7145 0.7642 0.7377 0.7984 0.8549
```

## 14.9. ANÁLISIS CLUSTER Y DATOS DE EXPRESIÓN DE GEN235

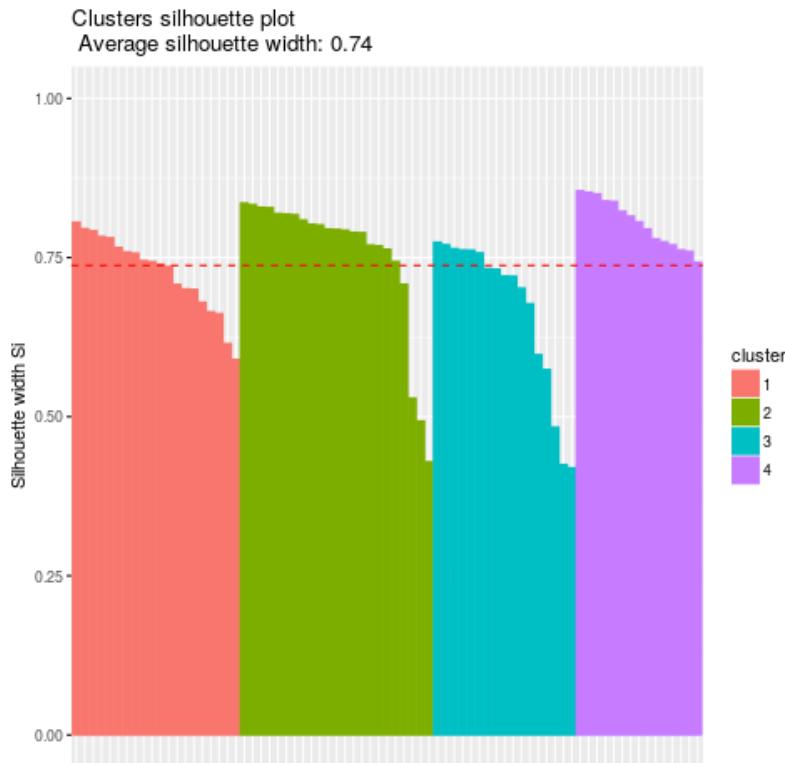


Figura 14.11: Silueta de la clasificación de los datos ruspini con método PAM y cuatro grupos.

También podemos representarla gráficamente en la figura 14.11.

```
png(paste0(dirTamiFigures, "Cluster63.png"))
fviz_silhouette(ruspini.pam)

##   cluster size ave.sil.width
## 1       1   20      0.73
## 2       2   23      0.75
## 3       3   17      0.67
## 4       4   15      0.80

dev.off()

## pdf
## 2
```

## 14.9 Análisis cluster y datos de expresión de gen

198

En las secciones previas hemos visto algunos métodos de clasificación así como de valoración de hay estructura cluster en un conjunto de datos dados. Estos métodos se aplican (sin modificación) en contextos

<sup>198</sup> Muchos comentarios e ideas de esta sección y del resto del capítulo las he tomado de la excelente presentación [41].

muy diversos. Sin embargo, ¿qué interés particular tiene su aplicación para datos de expresión de gen? Lo primero a considerar es que nos podemos plantear o bien la clasificación de las filas de la matriz de expresión o bien la clasificación de las columnas. Es decir, tiene sentido biológico clasificar genes o bien las muestras.

Si clasificamos las filas lo que tenemos en cada fila es el perfil de expresión del gen en todas las muestras del estudio. En este caso, puede ser que los grupos que obtengamos nos puedan servir posteriormente (quizás cruzando información con alguna de las bases de datos de anotación de genes) para realizar un enriquecimiento del conjunto.

Si clasificamos las columnas, las muestras, entonces podemos estar, por ejemplo, buscando subclases de células tumorales. Esto es, tenemos muestras que, a priori, son indistinguibles y posiblemente tengamos alguna subclase que no podíamos prever.

Los resultados de la clasificación de genes y muestras puede utilizarse como control de calidad. Sobre todo en las muestras tenemos un diseño experimental previo (habitualmente, el diseño grupo control frente a grupo de tratamiento).

## 14.10 Ejemplos

En esta sección incluimos análisis cluster completos de algunos bancos de datos. Utilizamos herramientas no solamente de este tema sino también de los anteriores.

### 14.10.1 Un análisis de los datos ALL

Son los datos del paquete [87, ALL] (§ 5.4). Cargamos los datos.

```
library("ALL")
data("ALL")
```

Podemos comprobar que es un ExpressionSet.

```
class(ALL)

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

Se realiza el preprocessado de los datos que indicamos en § 5.4.

```
bcell = grep("^B", as.character(ALL$BT))
types = c("NEG", "BCR/ABL")
moltyp = which(as.character(ALL$mol.biol) %in% types)
all = ALL[, intersect(bcell, moltyp)]
tipos = ALL$mol.biol[intersect(bcell, moltyp)]
tipos = factor(tipos) ## Eliminamos categorías vacías
```

Vemos que `all` es un ExpressionSet.

```
class(all)

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

Vamos a realizar un filtrado independiente de los genes. En concreto los criterios estadísticos serán que el nivel medio de expresión sea mayor que un cierto valor mínimo tanto en un tipo de biología molecular como en el otro.

```
alltemp0 = genefilter::nsFilter(all, var.func = mean, var.cutoff = 0.7)
alltemp1 = genefilter::nsFilter(alltemp0$eset, var.func = IQR, var.cutoff = 0.7)
all1 = alltemp1$eset

library(cluster)
all1.d = dist(exprs(all1), method = "manhattan")
all1.km = kmeans(exprs(all1), 2)
table(all1.km$cluster)

##
##    1    2
## 501 272

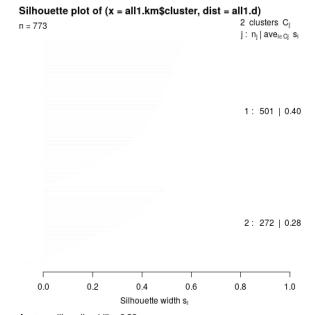
summary(silhouette(all1.km$cluster, all1.d))

## Silhouette of 773 units in 2 clusters from silhouette.default(x = all1.km$cluster, d =
##   Cluster sizes and average silhouette widths:
##      501      272
## 0.4033134 0.2847019
## Individual silhouette widths:
##      Min. 1st Qu. Median Mean 3rd Qu.
## -0.09864 0.24309 0.41445 0.36158 0.49128
##      Max.
## 0.58111
```

Y mostramos la silueta en figura 14.12.

```
png(paste0(dirTamiFigures, "Cluster71.png"))
plot(silhouette(all1.km$cluster, all1.d))
dev.off()

## pdf
## 2
```



Vemos un cluster jerárquico aglomerativo de los genes (figura ??).

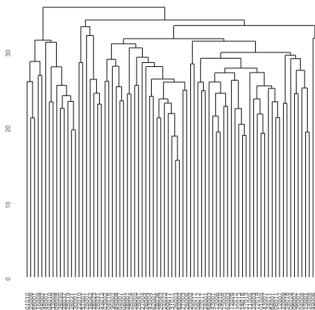
```
all1.ag = hclust(all1.d, method="average")
```

O un análisis jerárquico de las muestras en la figura ?? . Observemos que aplicamos la función cluster::agnes a la matriz traspuesta de las expresiones.<sup>7</sup>

```
dd = dist(t(exprs(all1)), method="euclidean")
muestras.ag = hclust(dd, method="average")
png(paste0(dirTamiFigures, "Cluster75.png"))
ggdendrogram::ggdendrogram(muestras.ag)
dev.off()
```

Figura 14.12

<sup>7</sup> La matriz traspuesta resulta de intercambiar filas por columnas en una matriz. La función  $t(A)$  nos devuelve la traspuesta de la matriz A.



```
## pdf
## 2
```

Recordemos que las muestras correspondían a dos grupos que vienen definidos por la covariable `mol.biol` que nos daba la biología molecular. Veamos si hay coincidencia entre estos tipos y los grupos que hemos obtenido.

```
tipos.ag = cutree(muestras.ag, 2)
```

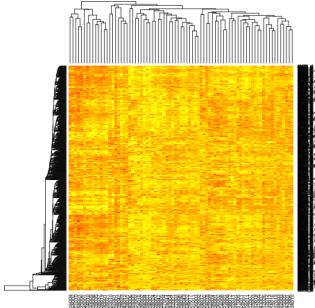
Y construimos la tabla de contingencia.

```
table(tipos,tipos.ag)
```

```
##               tipos.ag
## tipos      1   2
##   BCR/ABL 33   4
##   NEG      32  10
```

Mediante un `heatmap` podemos ver conjuntamente los dos dendogramas y una representación gráfica de los niveles de expresión.

```
png(paste0(dirTamiFigures, "Cluster79.png"))
heatmap(exprs(all1), hclustfun=agnes)
dev.off()
```



```
## pdf
## 2
```

Reflexionar qué nos da este dibujo. ¿Para qué nos puede servir un dibujo como este? Básicamente podemos intentar ver si hay *grupos formados por filas (genes) y columnas (muestras)*.

Supongamos que nos planteamos un PAM y un k-medias con dos grupos para las muestras. Vamos a clasificar con cada uno de los métodos e intentar ver si hay relación con la variable `tipos` que contiene la biología molecular.

```
all1.pam = pam(t(exprs(all1)), 2)
table(tipos, all1.pam$cluster)
```

```
##
## tipos      1   2
##   BCR/ABL 26  11
##   NEG      24  18
```

```
all1.km = kmeans(t(exprs(all1)), 2)
table(tipos, all1.km$cluster)
```

```
##
## tipos      1   2
##   BCR/ABL 10  27
##   NEG      21  21
```

No parece que tengamos ninguna relación clara. Como siempre hay que seguir trabajando los datos a ver si se encuentra algo.

Figura 14.13: Heatmap

### 14.10.2 Análisis cluster de GSE20986

Utilizamos los datos preprocesados. Los cargamos.

```
data(gse20986, package="tamidat")
gse = gse20986

library(genefilter)
gse0 = nsFilter(gse, var.func = mean, var.cutoff = 0.7)
gse1 = nsFilter(gse0$eset, var.func = IQR, var.cutoff = 0.7)
gse = gse1$eset
```

¿Con cuántas filas nos hemos quedado?

```
dim(exprs(gse))
## [1] 1818    12
```

Vamos a clasificar las muestras y compararemos con la clasificación original de las mismas. Utilizamos distancia Manhattan y promedio con el método PAM.

Primero calculamos la matriz de distancias.

```
d0 = dist(t(exprs(gse)), method = "manhattan")
```

Y aplicamos PAM.

```
library(cluster)
gse.pam = pam(d0, diss = TRUE, k=4)
```

La clasificación original es

```
tipo0 = rep(1:4, rep(3, 4))
```

```
table(tipo0, gse.pam$clustering)

##
## tipo0 1 2 3 4
##      1 1 1 1 0
##      2 2 1 0 0
##      3 3 0 0 0
##      4 0 0 0 3
```

Lo que sale es muy interesante y tiene una interpretación.

## 14.11 Otras aproximaciones y problemas

Sin duda, un problema de gran interés es lo que se conoce como biclustering o co-clustering. En este caso intentamos grupos de filas (genes) y columnas (muestras) que muestren unos valores de expresión similares. Es un problema importante pero no lo tratamos (de momento) en estas notas. Una referencia fundamental de biclustering es [94].

La clasificación es tanto mejor cuanta más información se utiliza. En particular un trabajo de interés para clasificación de genes incorporando información de Gene Ontology es [143].

Cuando se clasifican muestras el mayor problema es la enorme dimensión de los vectores que clasificamos ( $N$  en nuestro caso). Una posibilidad es seleccionar genes y no utilizarla la expresión para todos ellos. En [2] se propone un método utilizando información de Gene Ontology.

## 14.12 Ejercicios

### Ejercicio 27

Utilizando los datos *ALL* se pide:

1. Filtrar las muestras (columnas) con biología molecular “BCR/ABL” y “NEG”.
2. Quedarse con los genes (filas) que verifiquen las siguientes condiciones:
  - (a) La media de los niveles de expresión sobre las biologías moleculares indicadas sea superior a 5.62.
  - (b) La desviación estándar de las expresiones del gen ha de ser superior a 1.85.
3. Aplicar un análisis cluster jerárquico a los genes. En concreto:
  - (a) Hay que representar el dendograma.
  - (b) Decidir por inspección visual qué número de grupos parece razonable. Supongamos que denotamos por  $k$  el número de grupos.
  - (c) Representar la silueta y obtener su anchura media.
4. Aplicar un análisis cluster jerárquico a las muestras. En concreto:
  - (a) Hay que representar el dendograma.
  - (b) Decidir por inspección visual qué número de grupos parece razonable. Supongamos que denotamos por  $k$  el número de grupos.
  - (c) Representar la silueta y obtener su anchura media.
5. En este apartado vamos a aplicar un k-medias. En concreto vamos a ir clasificando según este método y nos quedaremos con el número de grupos que tengan una anchura media de la silueta mayor.
6. Repetir el apartado anterior utilizando el método PAM. Discutir el número de grupos.
7. Comparar las clasificaciones obtenidas en los dos apartados anteriores.

# **Parte V**

## **Análisis de grupos de genes**



# Capítulo 15

## Grupos de genes

### 15.1 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 (§ 13). 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** Podemos utilizar las tres ontologías consideradas en **Gene Ontology**. El grupo de genes viene definido por todos aquellos genes que tienen este término.

**KEGG** Tenemos las rutas de señalización para muchos organismos.

**MSigDB** En **MSigDB**<sup>199</sup> consideran hasta ocho formas distintas de construir estos grupos.

<sup>199</sup> The Molecular Signatures Database

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.

Como vemos 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.<sup>200</sup>

<sup>200</sup> No son grupos obtenidos por un análisis cluster previo.

<sup>201</sup> Siempre nuestro interés principal.

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.<sup>201</sup>

Vamos a definir conjuntos de genes o colecciones de conjuntos de genes utilizando el paquete [98, GSEABase].

```
library(GSEABase)
```

## 15.2 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 ([§ 5.11](#)).

```
data(gse20986, package="tamidata")
eset = gse20986
```

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(eset[345:405, ], setName = "Burjasot"))
```

Los genes tienen sus identificadores obtenidos de su anotación (mos- tramos 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.<sup>202</sup> Podemos tener más información con

<sup>202</sup> Ver [§ 4](#).

```
details(egs)

## setName: Burjasot
## geneIds: 1552739_s_at, 1552740_at, ..., 1552822_at (total: 61)
## geneIdType: Annotation (hgu133plus2)
## collectionType: ExpressionSet
## setIdentifier: orugacanora:29480:Wed May 9 17:05:18 2018:26335
## 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 ExpressionSet 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 ExpressionSet 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 `eset`.

```
gsc = GeneSetCollection(eset, setType = GOCollection())
```

Podemos ver un resumen de los conjuntos.

```
gsc

## GeneSetCollection
##   names: GO:0000002, GO:0000011, ..., GO:2001070 (17312 total)
##   unique identifiers: 1555591_at, 201917_s_at, ..., 1552675_at (37100 total)
##   types in collection:
##     geneIdType: AnnotationIdentifier (1 total)
##     collectionType: GOCollection (1 total)
```

Por ejemplo, el correspondiente a GO:0000122 sería

```
gsc[["GO:0000122"]]

## setName: GO:0000122
## geneIds: 1487_at, 1552338_at, ..., AFFX-HUMISGF3A/M97935_MB_at (total: 1936)
## geneIdType: Annotation (hgu133plus2)
## collectionType: GO
##   ids: GO:0000122 (1 total)
##   evidenceCode: EXP IDA IPI IMP IGI IEP ISS ISO ISA ISM IGC IBA IBD IKR IRD RCA TAS
##   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"  "63875"
## [11] "84275"  "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.

```
gscfilt = gsc[sapply(geneIds(gsc), length) > 10]
```

Son los siguientes grupos.

```
geneIds(gscfilt)
```

### 15.3 Grupos con levadura

Veamos cómo construir grupos de genes utilizando los términos GO en la *Saccharomyces cerevisiae*. Empezamos cargando el fichero de anotación.

```
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())
```

Y lo guardamos.

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, subsettingGeneSet, fn0 = featureNames(gse6647))
gsc2 = GeneSetCollection(gsc1)
```

## 15.4 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.<sup>1</sup>

```
library(GSEABase)
```

```
gse1397.gsc = GeneSetCollection(gse1397, setType=GOCollection())
names(gse1397.gsc) = unlist(lapply(gse1397.gsc, setName))
```

¿Cuántos grupos tenemos definidos?

---

<sup>1</sup>Se lleva un tiempo así que paciencia.

```

gsc = gse1397.gsc
length(gsc)

## [1] 16445

```

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, ..., 222216_s_at (total: 18)
## geneIdType: Annotation (hgu133a)
## collectionType: GO
##   ids: GO:0000002 (1 total)
##   evidenceCode: EXP IDA IPI IMP IGI IEP ISS ISO ISA ISM IGC IBA IBD IKR IRD RCA TAS
##   ontology: CC MF BP
## details: use 'details(object)'

```

Y sus identificadores (AffyId) 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"   "212607_at"   "212609_s_at"
## [13] "214306_at"   "214684_at"   "214821_at"
## [16] "217497_at"   "219393_s_at" "222216_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 212607_at 212609_s_at
## "4976"      "4976"      "10000"     "10000"
## 214306_at 214684_at 214821_at 217497_at
## "4976"      "4205"      "291"       "1890"
## 219393_s_at 222216_s_at
## "10000"     "63875"

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"
##      212607_at1      212607_at2
## "ENSG00000117020" "ENSG00000275199"
##      212609_s_at1      212609_s_at2
## "ENSG00000117020" "ENSG00000275199"
##      214306_at      214684_at
## "ENSG00000198836" "ENSG00000068305"
##      214821_at      217497_at
## "ENSG00000151729" "ENSG00000025708"
##      219393_s_at1      219393_s_at2
## "ENSG00000117020" "ENSG00000275199"
##      222216_s_at
## "ENSG00000158042"

```

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
## 2999  2047  1601  1263   983   711

```

Podemos ver que de tamaño uno tenemos 2336 grupos. Nos quedamos con aquellos grupos que, al menos, tienen 50 genes.

```
gse1397.gscfilt = gsc[which(sapply(geneIds(gsc),length) > 50)]
```

## 15.5 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)
gsc1 = gscSc[gruposGrandes]
```

Podemos ver el primer grupo.

```
gsc1[[1]]
```

```
## setName: GO:0000001
## geneIds: YNL304W, YHR194W, ..., YNR035C (total: 29)
## geneIdType:
## collectionType: GO
##   ids: GO:0000001 (1 total)
##   evidenceCode: EXP IDA IPI IMP IGI IEP ISS ISO ISA ISM IGC IBA IBD IKR IRD RCA TAS
##   ontology: CC MF BP
## details: use 'details(object)'
```

## 15.6 Grupos GO para humanos

En esta sección construimos los grupos de genes según Gene Ontology para humanos. El código es el siguiente:<sup>203</sup>

<sup>203</sup> Para otros organismos sería similar reemplazando el paquete [28] por el correspondiente al organismo que nos interese.

```
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"))
```

## 15.7 Grupos para Arabidopsis thaliana

Utilizamos el paquete ??]R-ath1121501.db.

```
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"))
```

## 15.8 Ejercicios

**Ej. 28** — Construir los grupos basados en Gene Ontology y en KEGG para el ratón (*Mus musculus*).

**Ej. 29** — Construir los grupos basados en Gene Ontology y en KEGG para las cepas de la *E. coli* que puedas.

**Ej. 30** — Consultar las anotaciones OrgDb para decidir de qué organismos podemos construir los grupos.

## Capítulo 16

# Test de Fisher unilateral

### 16.1 Introducción

Por alguna razón (que desconozco) a este procedimiento se le llama en numerosas publicaciones de perfil biológico *test hipergeométrico*.<sup>1</sup> 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.<sup>204</sup> 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<sup>2</sup> o un no (no es significativo, no hay expresión diferencial o no hay asociación gen-fenotipo).

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 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

<sup>204</sup> 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.

<sup>1</sup> La razón de este nombre sí la entiendo (la distribución del estadístico de contraste es la distribución hipergeométrica) lo que no entiendo es el cambio de nombre ya que la denominación test de Fisher está más que consolidada y debiera de utilizarse.

<sup>2</sup> A gusto del consumidor.

Tabla 16.1:  $S_0$  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.

	$S_1$	$S_1^c = G \setminus S_1$	
$S_0$	$n_{11}$	$n_{12}$	$n_{1\cdot}$
$S_0^c = G \setminus S_0$	$n_{21}$	$n_{22}$	$n_{2\cdot}$
	$n_{\cdot 1}$	$n_{\cdot 2}$	$N$

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.<sup>205</sup> 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 16.1. 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.

<sup>205</sup> § 15.

En la tabla 16.1 consideramos datos 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 16.1? 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

$$p = P(N_{11} \geq n_{11}) = \sum_{t=n_{11}}^{\min\{n_{1\cdot}, n_{\cdot 1}\}} \frac{\binom{n_{1\cdot}}{t} \binom{n_{2\cdot}}{n_{\cdot 1} - t}}{\binom{N}{n_{\cdot 1}}}.$$

## 16.2 fisher.test

Supongamos que hemos observado la tabla 16.2 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.

Tabla 16.2:  $S_0$  ( $S$ ) indica el grupo de genes significativos (grupo pre-definido) y  $S_0^c$  ( $S^c$ ) 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

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. Su explicación la encontramos en [??. 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.](#)

## 16.3 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? Consideraremos los datos

Tabla 16.3:  $S_0$  (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	$196 + k$	$346 + k$

de la tabla 16.3. ¿Qué describe la tabla? Supongamos que tenemos dos conjuntos de genes dados,  $S_0$  y  $S_1$ . Dado este par de 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 16.3 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 16.3 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 ??) y el cociente de odds crece (figura ??). En definitiva la interpretación tanto del p-valor como del cociente de odds depende del universo de genes que estamos considerando.

```
## pdf
## 2
## pdf
## 2
```

En la figura 16.1(a) (respectivamente en la figura 16.1(b)) tenemos el p-valor del test de Fisher unilateral correspondiente a la tabla 16.3 (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*.

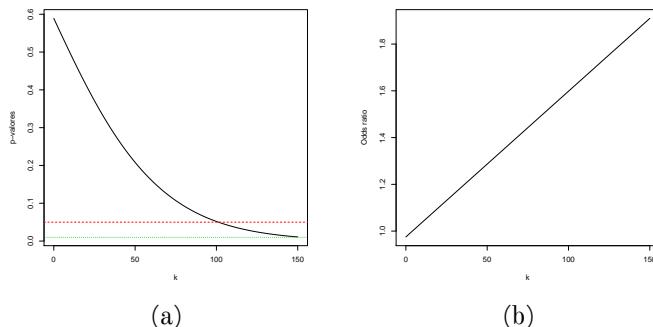


Figura 16.1: a) p-valores del test de Fisher unilateral. b) Cocientes de odds.

## 16.4 Utilizando Category y GOstats

En esta sección utilizamos los paquetes [56, GOstats] (ver también [47]) y [55, Category].

### 16.4.1 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?<sup>206</sup>

<sup>206</sup> Es una base de datos asociada a un chip.

```
annotation(eset)
## [1] "hgu133a"
```

Cargamos el paquete con la base de datos correspondiente.

```
library(hgu133a.db)
```

¿Tenemos un solo identificador GO para cada gen?

```
G1.entreizd = unlist(mget(featureNames(eset), hgu133aENTREZID))
anyDuplicated(G1.entreizd)
## [1] 6
```

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)
```

207 En <http://geneontology.org/> te-

nemos una aplicación en línea que nos realiza un análisis similar.

Y realizamos los tests.<sup>207</sup>

```
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))

##      GOBPID      Pvalue OddsRatio   ExpCount
## 1 GO:0042776 1.411212e-05 711.0000 0.006318006
## 2 GO:0015985 2.332207e-05 541.4762 0.008073008
## 3 GO:0015986 2.332207e-05 541.4762 0.008073008
## 4 GO:0042407 2.764815e-05 494.3043 0.008775009
## 5 GO:0007007 4.861515e-05 366.4839 0.011583012
## 6 GO:0006754 7.919224e-05 283.8000 0.014742015
##      Count Size
## 1       2   18
## 2       2   23
## 3       2   23
## 4       2   25
## 5       2   33
## 6       2   42
##
##                                         Term
## 1 mitochondrial ATP synthesis coupled proton transport
## 2 energy coupled proton transport, down electrochemical gradient
## 3                               ATP synthesis coupled proton transport
```

```
## 4                      cristae formation
## 5          inner mitochondrial membrane organization
## 6          ATP biosynthetic process
```

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()
```

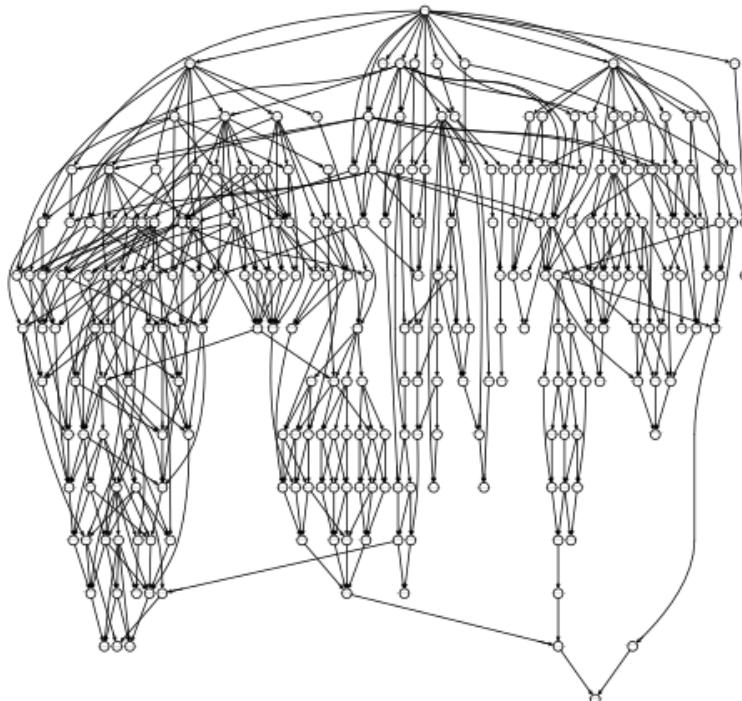


Figura 16.2: Grafo **Gene Ontology** con las grupos significativos.

### 16.4.2 GSE20986

Leemos los datos normalizados y lo renombramos.

```
data(gse20986, package = "tamidata")
gse = gse20986
```

En lo que sigue necesitamos la anotación de este ExpressionSet.

```
Biobase::annotation(gse)
## [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<sup>3</sup> podría ser aplicar un filtrado no específico y que el universo de genes sean los restantes.

```
library(genefilter)
gse.filt = genefilter::nsFilter(gse, var.func = IQR, var.cutoff = 0.6,
                                require.GOBP = TRUE)$eset
```

¿Cuántos genes nos quedan?

```
dim(gse.filt)
## Features Samples
##       6134      12
```

¿Tenemos un solo identificador GO para cada gen?

```
G1.entreizd = unlist(mget(featureNames(gse.filt), hgu133plus2ENTREZID))
anyDuplicated(G1.entreizd)

## [1] 0
```

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.entreizd = unlist(mget(featureNames(gse), hgu133plus2ENTREZID))
anyDuplicated(G2.entreizd)

## [1] 6
```

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(gse), 1, IQR)
uniqGenes = findLargest(featureNames(gse), gse.iqr, "hgu133plus2")
gse.filt2 = gse[uniqGenes, ]
```

Y ahora construimos el universo de genes y comprobamos que no hay duplicidades.

---

<sup>3</sup>Y nada más que esto razonable y tan razonable como muchas otras.

```
G2.entrezid = unlist(mget(featureNames(gse.filt2), hgu133plus2ENTREZID))
anyDuplicated(G2.entrezid)

## [1] 0
```

Aplicamos el procedimiento de expresión diferencial utilizado en el ejemplo 7.2.

```
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.

```
library(GO.db);library(Category);library(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.

```
f1 = tempfile()
htmlReport(overRepresented, file = f1)
browseURL(f1)
```

### 16.4.3 ALL

Esta sección se reproduce el análisis propuesto en [http://www.bioconductor.org/help/course-materials/2009/SSCMay09/gsea/HyperG\\_Lecture.pdf](http://www.bioconductor.org/help/course-materials/2009/SSCMay09/gsea/HyperG_Lecture.pdf). Utilizamos los datos [87, ALL]. En § 5.4 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`.<sup>4</sup>

```
bcrnegfilt = nsFilter(bcrneg, var.cutoff = 0.5, require.GOBP = TRUE)$eset
```

Veamos qué nos queda.

```
dim(bcrnegfilt)
## Features Samples
##      4074       79
```

Vamos a generar una lista de genes significativos de un modo muy básico<sup>5</sup>. 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(bcrnegfilt)[, "mol.biol"]
fac0 = factor(fac0) ## Quitamos categorías vacías
rtt = rowttests(bcrnegfilt, 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(bcrnegfilt)
```

Guardamos estos identificadores en `ids`.

```
ids = featureNames(bcrnegfilt)
```

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. Prime-  
ro cargamos el paquete [24, GO.db].

```
library(GO.db)
```

Nos fijamos por ejemplo en el término siguiente

---

<sup>4</sup>En concreto se pide su anotación en la ontología *Biological Process*.

<sup>5</sup>Y tampoco muy recomendable.

Tabla 16.4: Tabla de contingencia para el término GO:0006468.

		InGO	
		FALSE	TRUE
Selected	FALSE	3101	162
	TRUE	658	30

```
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 16.4 muestra los conteos observados.

Cargamos los paquetes [55, Category] y [56, GOstats].

```
library(Category)
library(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.

```
f1 = tempfile()
htmlReport(overRepresented, file = f1)
browseURL(f1)
```

## 16.5 Ejercicios

### Ejercicio 31

Utilizamos los datos tamidata::gse20986. En el problema 21 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  $S_{coroides}$ .

- 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).



# Capítulo 17

## Análisis de conjuntos de genes

### 17.1 Introduction

1

El problema abordado en este capítulo corresponde a lo que se conoce como análisis de conjunto de genes.<sup>208</sup> Estudiamos si hay relación entre conjuntos de genes **previamente definidos** y fenotipo.<sup>209</sup> 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.

<sup>208</sup> En la literatura se refieren a este problema como *gene set analysis*, *gene set enrichment analysis*, *set based enrichment analysis*.

<sup>209</sup> Donde fenotipo se entiende en un sentido amplio como en todo el texto.

### 17.2 Sobre la distribución de la matriz de expresión

Denotaremos (como siempre) la matriz de expresión (aleatoria) como

$$X = [X_{ij}]_{i=1, \dots, N; j=1, \dots, n}$$

<sup>1</sup> 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  $X_{ij}$  es la expresión aleatoria del  $i$ -ésimo gen en la  $j$ -ésima muestra. Las expresiones *observadas* serán

$$x = [x_{ij}]_{i=1,\dots,N; j=1,\dots,n}$$

Notemos que  $X_{ij}$  es una variable aleatoria mientras que  $x_{ij}$  es un valor observado. Las columnas de la matriz de expresión  $X$  son independientes. Son vectores aleatorios independientes<sup>210</sup> 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 corregulaciones entre ellos. Además tampoco tienen porqué comportarse de un modo (aleatorio) común.

<sup>210</sup> Corriendo un tupido velo por toda la parte de preprocesado de la información que hemos visto en § 4 para datos de microarrays. Es claro que el preprocesado de la información introduce dependencias entre valores observados para distintas muestras pero las ignoramos.

### 17.3 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.<sup>211</sup> Si denotamos por  $G = \{1, \dots, N\}$  el conjunto total de genes considerado (o universo de genes) entonces los conjuntos de genes serán  $S_1, \dots, S_K$ . 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  $n_{S_k}$  o, si no hay confusión, simplemente  $n_k$ . Los distintos conjuntos  $S_k$  **no** son una partición de  $G$ : no son necesariamente disjuntos ( $S_i \cap S_j \neq \emptyset$  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?*<sup>212</sup> Es una pregunta muy vaga. Se requiere una formulación más precisa. Caben distintas interpretaciones de la pregunta. En [136] formulan las siguientes dos hipótesis nulas que concretan de dos modos distintos la cuestión previa. Reproducimos las hipótesis nulas.<sup>213</sup>

**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, no. 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). 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 [GoemanBuhlmann2007]. 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:

<sup>211</sup> § 15.

<sup>212</sup> Esto es, la misma pregunta que nos formulábamos para cada gen pero ahora referido a un conjunto dado de genes.

<sup>213</sup>

**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.

**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  $G^c$ .

**Hipótesis nula autocontenido  $H_0^{auto}$ :** Ningún gen en  $S$  está diferencialmente expresado.

<sup>2</sup>

Muchos procedimientos estadísticos propuestos contrastar estas hipótesis no formula 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?

## 17.4 Ejemplos

En esta sección comentamos los ejemplos que utilizamos.

### 17.4.1 Un ejemplo simulado de Efron y Tibshirani

En [44] se proponen un par de ejemplos con datos simulados. Son los ejemplos 17.1 y 17.2.

**Ejemplo 17.1** 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 misma distribución. La distribución común es una normal estándar ( $X_{ij} \sim N(0, 1)$ ). 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.

---

<sup>2</sup>Es interesante leer las hipótesis nula tal como las formulan:

**Competitive null hypothesis  $H_0^{comp}$ :** The genes in  $G$  are at most as often differentially expressed as the genes in  $G^c$ .

**Self-contained null hypothesis  $H_0^{self}$ :** No genes in  $G$  are differentially expressed.

```
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 17.2 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)
png(paste0(dirTamiFigures, "GeneSetAnalysis5b.png"))
ggplot(df, aes(x=et1.tt)) + geom_density()
dev.off()
```

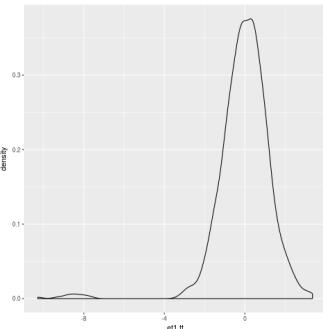


Figura 17.1: Densidad estimada de et1.tt.

**Ejemplo 17.2** Este ejemplo se define exactamente como el ejemplo 17.1 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 17.1. Definimos los grupos para su uso posterior.

```
gen.name = function(i) paste("g", ((i-1)*20 +1):(i*20), sep="")
gsc.et = lapply(as.list(1:50), gen.name)
names(gsc.et) = paste("set", as.character(1:50), sep="")
gsnames.et = paste("set", as.character(1:50), sep="")
genenames.et = paste("g", 1:1000, sep="")
```

```
et2.tt = genefilter::rowttests(et2,y.et)$statistic
```

En figura ?? tenemos una estimación de la densidad.

```
pacman::p_load("ggplot2")
df = data.frame(et2.tt)
png(paste0(dirTamiFigures, "GeneSetAnalysis11c.png"))
ggplot(df, aes(x=et2.tt))+geom_density()
dev.off()

## pdf
## 2
```

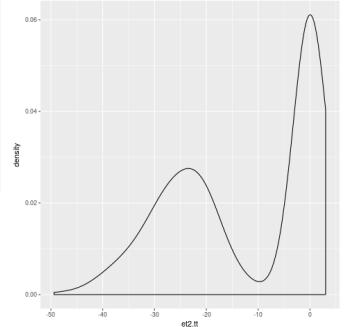


Figura 17.2: Densidad estimada de et1.tt.

## 17.5 Quantificando 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 muestrales  $n_1$  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 (§ 7.8).

En § 9.1 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).

## 17.6 Enriqueciendo el conjunto de genes

El vector  $t = (t_1, \dots, t_N)$  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

$$t(S) = \sum_{i \in S} \frac{t_i}{n_S}, \quad (17.1)$$

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.

## 17.7 Distribuciones condicionadas a los datos

Sea  $x$  la matriz 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 vector  $t = (t_1, \dots, t_N)$ .

### 17.7.1 Distribución de permutación para un gen

Para un gen tenemos su perfil de expresión  $u = (u_1, \dots, u_n)$  y tenemos el vector  $y$  asociado a las muestras.

Sea  $\pi$  denota una permutación aleatoria de  $(1, \dots, n)$  de forma que los  $\pi(i)$  será el índice que ocupa la posición  $i$ -ésima en la permutación  $\pi$ . En particular denotaremos por  $\pi_0$  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  $y$ )  $y_\pi = (y_{\pi(1)}, \dots, y_{\pi(n)})$ . Obviamente  $y = y_{\pi_0}$ .

La cantidad que describe la asociación entre  $u$  y  $y_\pi$  es  $t_\pi$ . Utilizando esta notación el valor observado de la asociación gen-fenotipo para  $u$  e  $y$  será  $t_{\pi_0}$ . Si consideramos  $B$  permutaciones aleatorias entonces tendremos los valores  $t_{\pi_1}, \dots, t_{\pi_B}$  para las  $B$  permutaciones aleatorias.

Si no hay asociación gen-fenotipo entonces el valor de  $t_{\pi_0}$  debe de ser *como* los valores  $t_{\pi_1}, \dots, t_{\pi_B}$ . De hecho, cualquier ordenación de  $t_{\pi_0}, t_{\pi_1}, \dots, t_{\pi_B}$  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  $t_{\pi_b}$  (con  $b = 1, \dots, B$ ) son mayores que  $t_{\pi_0}$ ? 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  $t_{\pi_b}$ s mayores que  $t_{\pi_0}$ , es decir,

$$p_r = \frac{|\{b : t_{\pi_b} > t_{\pi_0}\}|}{B}. \quad (17.2)$$

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

$$p_r = \frac{|\{b : |t_{\pi_b}| > |t_{\pi_0}|\}|}{B}, \quad (17.3)$$

ya que tendríamos un test bilateral.

**Ejemplo 17.3** Consideremos el ejemplo 17.1, 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  $t_{\pi_0}$  es 8.77. Y el p-valor será la proporción de los que su valor absoluto es mayor que el valor absoluto de  $t_{\pi_0}$ ), es decir,

```
sum(tb > t0) / B
```

```
## [1] 0
```

Tenemos un p-valor nulo. No parece que todas las ordenaciones de los valores  $t_{\pi_0}, t_{\pi_1}, \dots, t_{\pi_B}$  sean equiprobables.

La distribución nula que hemos considerado en esta sección corresponde con la hipótesis nula autocontenido 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 autocontenido esta sería la distribución nula natural.<sup>214</sup>

### 17.7.2 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  $t(S)$  (ahora las muestras tienen su ordenación original). A la distribución de probabilidad de  $t(S)$  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  $b = 1, \dots, B$  y los valores del estadístico de enriquecimiento observados son  $t(S_0)$  (para el grupo original) y  $t(S_b)$  con  $b = 1, \dots, B$  para los seleccionados al azar. El p-valor se define análogamente como

$$p = \frac{|\{b : t(S_b) > t(S_0)\}|}{B}. \quad (17.4)$$

si solamente rechazamos para valores grandes (positivos). En el caso bilateral tendremos

$$p = \frac{|\{b : |t(S_b)| > |t(S_0)|\}|}{B}. \quad (17.5)$$

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.<sup>215</sup>

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).

<sup>214</sup> Podemos hablar de aleatorización por columnas entendiendo que las distintas muestras corresponden a las distintas columnas en la matriz de expresión.

<sup>215</sup> 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. También se habla de muestreo de genes o *gene sampling*.

## 17.8 Limma::genSetTest

En el paquete [129, limma] tenemos la función limma::geneSetTest. En este caso la medida de enriquecimiento que se utiliza es la media muestra de los estadísticos calculados para cada gen. Simplemente se permuta por filas. Contrastamos pues la hipótesis competitiva.

**Ejemplo 17.4 (genSetTest y ejemplo 17.1)** Consideremos los datos del ejemplo 17.1 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.

## 17.9 Limma::wilcoxGST

Siguiendo con el paquete [129, 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.<sup>216</sup> 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.

<sup>216</sup> ??.

**Ejemplo 17.5 (Datos de ejemplo 17.1)** 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
```

## 17.10 Limma::CAMERA

El método fue propuesto en [154].<sup>217</sup> 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 procedimientos 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.

Se asume que la expresión está cuantificada en escala logarítmica (base 2 como es habitual en este contexto).

$$E[X_{ij}] = \mu_{ij} = \sum_{k=1}^p \alpha_{ik} y_{jk} \quad (17.6)$$

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  $\sigma_i^2$ .<sup>218</sup> Se asume que las expresiones para distintas muestras son independientes pero no entre distintos genes. En particular, denotamos el coeficiente de correlación de Pearson<sup>219</sup> entre las variables aleatorias  $X_{i_1j}$  y  $X_{i_2j}$  como  $cor(X_{i_1j}, X_{i_2j}) = \rho_{i_1, i_2}$ . Obviamente no asumimos que estos coeficientes sean nulos.<sup>220</sup>

Un contraste, para el gen  $i$ -ésimo viene dado por

$$\beta_i = \sum_{k=1}^p c_j \alpha_{ik}.$$

<sup>217</sup> El nombre es un acrónimo Correlation Adjusted MEan RAnk.

<sup>218</sup> Asumible si hemos aplicado previamente algún procedimiento de normalización.

<sup>219</sup> § 27.1.

<sup>220</sup> No asumimos incorrelación. Recordemos que independencia implica incorrelación pero no viceversa.

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

<sup>221</sup> No necesariamente con distribución normal.

## 17.11 GSA

<sup>222</sup> En el paquete [43, GSA] se implementa el método propuesto en [44].

<sup>222</sup> Se utiliza la distribución de permutación a la que aplican una reestandarización. La medida de enriquecimiento que proponen por defecto <sup>3</sup> 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

$$t(S) = \max\{\bar{t}_S^+, \bar{t}_S^-\}. \quad (17.7)$$

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 17.6** [44, página 119] *La medida de enriquecimiento 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 [137, samr] que aparecen en la sección § 9.1.1. 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 17.7 (Ejemplo 17.1 con GSA)** *Empezamos analizando el ejemplo 17.1.*

```
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)
```

*Podemos ver los estadísticos enriquecidos para los grupos*

<sup>3</sup> Aunque lleva el promedio de los  $t_i$ 's y el promedio de  $|t_i|$  como otras opciones.

```
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.   Max.
## -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 17.3 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 17.4 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 17.1).*

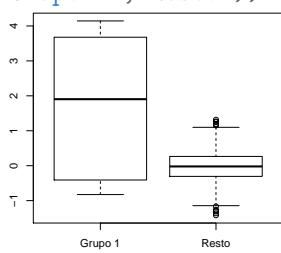


Figura 17.3: Valores  $t_i$  para el primer grupo (izquierda) y los demás. El primer grupo tiene diez genes claramente diferenciados mientras que los demás no lo están. De ahí la gran variabilidad que muestra el gráfico.

```
GSA.plot(et1.gsa)
```

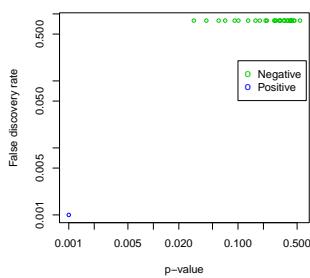


Figura 17.4: Grupos significativos con datos et1 en función de FDR.

Vemos cómo solamente admitimos un grupo con un valor bajo de FDR.

**Ejemplo 17.8 (Ejemplo 17.2 con GSA)** En el ejemplo 17.2 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 gnames.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 17.5 muestra los grupos significativos en función de la FDR para los datos et2 (ejemplo 17.2).

```
GSA.plot(et2.gsa)
```

Es claro que ningún grupo se diferencia de los demás.

**Ejemplo 17.9 (Datos GSE1397 y GSA)** Leemos los datos.

```
data(gse1397, package="tamidat")
data(gse1397.gsc, package="tamidat")
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 [43, 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 17.6 podemos ver la representación de la tasa de falsos positivos como función del p-valor.

```
GSA.plot(gse1397.gsa)
```

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.

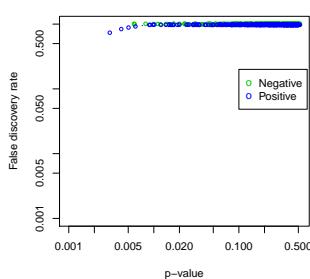


Figura 17.6: Grupos significativos con datos gse1397.

```
(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:0000012 GO:0000018
## "GO:0000027" "GO:0000028" "GO:0000160"
## GO:0000019 GO:0000022 GO:0000023
## "GO:0000271" "GO:0000462" "GO:0000466"
```

*y análogamente*

```
head(names(gsc[ind.hi]))

## GO:0000002 GO:0000012 GO:0000018
## "GO:0000012" "GO:0000186" "GO:0000187"
## GO:0000019 GO:0000022 GO:0000023
## "GO:0000302" "GO:0000305" "GO:0001507"
```

*Y ahora viene el trabajo del especialista para ver hasta qué punto lo que sale tiene sentido o no.*

## 17.12 GSEA: Gene set enrichment analysis

223

La referencia básica es [133] que presenta una versión modificada del procedimiento originalmente propuesto en [97]. Es el método más popular para realizar un enriquecimiento (agregado). Se basa en el uso del estadístico de Kolmogorov-Smirnov para dos muestras.<sup>224</sup> De hecho, utiliza una versión modificada de este estadístico que lo hace más sensible a la detección de interacción entre grupo de genes y fenotipo.

### Método

Empezamos ordenando el universo de genes de acuerdo al grado de asociación gen-fenotipo utilizando algunos de los estadísticos propuestos en § 17.5. Tendremos la lista de genes ordenada,  $L$ .

**Entradas** 1. La matriz de expresión  $X$  con  $N$  filas y  $n$  columnas.

<sup>223</sup> Este método no está implementado en un paquete R. Los autores proporcionan código en R pero no en forma de paquete. Existe un interfaz Java para utilizarlo fuera de R/Bioconductor que no es nuestra opción de trabajo.

<sup>224</sup> En la sección § 25.6 se explica el método.

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 genotípico,  $t_i$ .

2. Ordenamos el universo de genes de acuerdo a las medidas de asociación del paso 1. Denotamos los índices ordenados con  $r_1, \dots, r_N$ , es decir,

$$t_{r_1} \geq \dots \geq t_{r_N}.$$

3. Calculamos para cada  $i$  con  $i = 1, \dots, N$

$$h_S(i) = \frac{1}{\sum_{r_i \in S} |t_{r_i}|^p} \sum_{j \leq i; r_j \in S} |t_j|^p, \quad (17.8)$$

Calculamos también

$$m_S(i) = \sum_{j \leq i; r_j \notin S} \frac{1}{N - n_S}, \quad (17.9)$$

El valor de  $e_S$  es la máxima desviación de cero de  $h_S(i) - m_S(i)$ , es decir,

$$e_S = \max_{1 \leq i \leq N} |h_S(i) - m_S(i)|. \quad (17.10)$$

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 (ver sección § 25.6).

**Estimación del p-valor** 1. Consideramos una asignación aleatoria del fenotípico a las muestras (una permutación aleatoria de las muestras manteniendo fijo el fenotípico), 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  $e_1, \dots, e_B$  son los valores de  $e_S$  para las asignaciones aleatorias de fenotípico a muestras entonces el p-valor viene dado por

$$p = \begin{cases} \frac{2 \cdot |\{e_i: e_i \geq e_0\}|}{B}, & \text{for } e_0 > 0, \\ \frac{2 \cdot |\{e_i: e_i \leq e_0\}|}{B} & \text{for } e_0 < 0. \end{cases}$$

**Contrastes múltiples** Suponemos que consideramos ahora una colección de conjuntos de genes de interés:  $S_1, \dots, S_K$ .

1. Calculamos  $e_{S_k}$  para  $k = 1, \dots, K$ .
2. Para cada  $S_k$  y 1000 permutaciones fijas  $\pi_i$  con  $i = 1, \dots, 1000$  (las mismas para todos los conjuntos de genes) del fenotípico, reordenamos los genes y calculamos el enriquecimiento  $e_{S_k, \pi_i}$ .

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. Consideraremos 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

$$q(\tilde{e}) = \frac{|\{k,i : e_{S_k,\pi_i} \leq \tilde{e}\}| / |\{k,i : e_{S_k,\pi_i} \leq 0\}|}{|\{k : e_{S_k} \leq \tilde{e}\}| / |\{k : e_{S_k} \leq 0\}|}$$

## 17.13 Estudiando interacciones entre genes individuales y grupos de genes

Suponemos que tenemos bien definido un grupo de genes  $S$  y pretendemos estudiar la posible asociación entre un gen  $g$  no incluido en  $S$  y el grupo  $S$ . Se trata de estudiar posibles asociaciones no conocidas previamente entre la actividad del gen individual y la actividad conjunta del grupo  $S$ .

En [22] consideran el caso en que las muestras están clasificadas en dos grupos. Se propone el siguiente método. Consideramos la matriz de datos  $[x_{ij}]_{i \in S, j \in \{1, \dots, n\}}$ . Se aplican una componentes principales a esta matriz de datos y nos quedamos con la primera componente principal. Por cada muestra tendremos la primera componente del grupo  $S$  y la expresión del gen de interés. Se calcula el coeficiente de correlación de los pares de valores indicados en un grupo y en otro (trabajamos en el caso en que las muestras las clasificamos en dos grupos y se el módulo de la diferencia de los coeficientes de correlación. Obviamente un valor muy grande indica que el coeficiente de correlación en cada uno de los grupos considerados es muy diferente y por tanto el tipo de asociación entre el gen y el grupo de genes es distinto.<sup>225</sup>

<sup>225</sup> Indican que ha implementado el método en el paquete R GPCscore pero no se encuentra en ningún repositorio.

## 17.14 Un método simple pero efectivo

Es un método propuesto en [77]. Insiste mucho en este trabajo en la innecesaria complejidad del método GSEA comentado en sección § 17.12. Su interés está en la detección de grupos de genes que se asocian con el fenotipo (consideran la situación en que comparamos dos grupos) donde algunos de los genes están sobre expresados (up-regulated) y otros infra expresados (down-regulated).

Como siempre  $t_i$  es la medida de asociación marginal gen-fenotipo (puede ser el t-estadístico pero no necesariamente). Si consideramos el grupo de genes  $S$  con  $n_S$  elementos. Asumiendo (lo que no es cierto) que los distintos  $t_i$  son valores observados de variables independientes y con varianza igual a 1, entonces aproximadamente (aplicando teorema central del límite) se tiene que

$$E^{(1)} = \sqrt{n_S} \bar{t}_S \sim N(0, 1), \quad (17.11)$$

siendo  $\bar{t}_S = \sum_{i \in S} t_i / n_S$ . Aunque no hay mayores justificaciones de esta distribución aproximada los autores indican que, sobre los datos, es asumible.<sup>4</sup> Utilizando esta distribución podemos tener fácilmente p-valores que no están basados en la distribución de permutación sino en la distribución normal estándar.

El estadístico considerado en 17.11 detecta cambios en la media pero no cambios de escala. Si un grupo de genes tiene la mitad que están sobre expresados y la otra mitad infra expresados entonces no detectaríamos cambios en la media pero la variabilidad de los valores  $t_i$  sí que se modificaría claramente. Proponen utilizar el siguiente estadístico con la siguiente distribución aproximada (con  $n_S \geq 20$  proponen los autores)

$$E^{(2)} = \frac{\sum_{i \in S} (t_i - \bar{t}_S)^2 - (n_S - 1)}{\sqrt{2(n_S - 1)}} \sim N(0, 1) \quad (17.12)$$

<sup>5</sup> Tendremos un p-valor asociado a cada grupo de genes. Una vez tenemos todos los grupos tenemos un problema de comparaciones múltiples que resolvemos utilizando lo visto en el capítulo § 8. En particular, los autores utilizan los q-valores de Storey. Esencialmente el trabajo se centra mucho en la comparación con el método GSEA destacando su simplicidad frente al otro y que los resultados obtenidos son esencialmente los mismos. Algunos comentarios propios (discutibles):

1. En mi opinión hay un uso de resultados asintóticos que no siempre son asumibles.
2. En cualquier caso, el tamaño de los grupos de genes ha de ser suficientemente grande para que se verifiquen.
3. Es una buena opción a probar.

---

<sup>4</sup> Si la cosa funciona: ¿por qué no?

<sup>5</sup> Bajo la hipótesis de que los  $t_i$  son independientes, con una distribución común con varianza unitaria entonces el estadístico tiene una distribución aproximadamente normal estándar. En el trabajo original dividen por  $2(n_S - 1)$  y no la raíz cuadrada. Entiendo que debe ser una errata.

## 17.15 Un método basado en poblaciones finitas

Veamos el método propuesto en [105]. Consideremos el conjunto de genes  $S$  con  $m$  genes.<sup>6</sup> Tenemos  $t_i$  la medida de asociación gen-fenotipo para el  $i$ -ésimo gen. Consideremos la media muestral como estadístico de enriquecimiento dado por

$$\bar{t}_S = \sum_{i \in S} \frac{t_i}{n_S}.$$

En el resto de aproximaciones lo que se considera aleatorio es la medida de asociación  $t_i$  y *fijo* el conjunto de genes  $S$ . En este trabajo se adopta otro punto de vista. Tenemos  $G = \{1, \dots, N\}$  un universo de genes y nuestro conjunto de genes  $S$  es un *subconjunto aleatorio* de  $G$ . Por tanto nos planteamos cuál es la distribución de probabilidad de  $\bar{t}_S$  cuando  $S$  es un subconjunto aleatorio de tamaño  $n_S$  de  $G$  o, dicho de otro modo, extraemos al azar y sin reemplazamiento  $n_S$  genes del total de  $N$  que componen nuestro universo. Los autores llaman a esta distribución de probabilidad el modelo de conjunto aleatorio (random-set model). Notemos que se plantea la distribución de probabilidad de  $\bar{t}_S$  **condicionada a los valores  $t_i$** . Este es el punto básico, los  $t_i$  están dados y *no se consideran aleatorios sino fijos, dados previamente*. Utilizando resultados de muestreo en poblaciones finitas se tiene que la media de  $\bar{t}_S$  es

$$\mu = E(\bar{t}_S) = \sum_{i \in G} \frac{t_i}{N}, \quad (17.13)$$

es decir, la media de todos los  $t_i$  observados sobre todo el universo de genes. La varianza es

$$\sigma^2 = \text{var}(\bar{t}_S) = \frac{1}{m} \frac{N-m}{N-1} \left[ \sum_{i \in G} \frac{t_i^2}{N} - \left( \sum_{i \in G} \frac{t_i}{N} \right)^2 \right]. \quad (17.14)$$

En este procedimiento no recurrimos a ninguna distribución de aleatorización del estadístico. Estamos utilizando una distribución asintótica en el sentido en que se asume un tamaño de  $S$  suficientemente grande para que (aplicando el teorema central del límite) podamos considerar que  $\bar{t}_S$  tiene una distribución aproximadamente normal. Como estadístico de enriquecimiento utilizan la versión estandarizada de  $\bar{t}_S$ , es decir, utilizan

$$Z = \frac{\bar{t}_S - \mu}{\sigma} \quad (17.15)$$

con  $\mu$  y  $\sigma$  dados en las ecuaciones 17.13 y 17.14. Bajo la hipótesis nula de que no hay ningún gen diferencialmente expresado en el conjunto de genes  $S$  entonces  $Z$  tiene aproximadamente una distribución normal estándar.

$$Z \sim N(0, 1). \quad (17.16)$$

## 17.16 Análisis de grupos de genes por muestra

---

<sup>6</sup>Estamos denotando el cardinal de  $S$  por  $n_S$ . Aquí denotamos por  $m$  porque queremos destacar que es un número fijo y dado una vez tenemos el grupo de genes  $S$ .

¿Qué es lo que se pretende en este tipo de procedimientos? Partiendo de una matriz de expresión y una colección de grupos de genes se obtiene una cuantificación **para cada muestra y grupo de genes** de lo diferente que es la expresión en esa muestra del grupo de genes considerado respecto del resto de grupos de genes. De alguna forma lo que estamos atendiendo es a la hipótesis competitiva pero para cada muestra. Pasamos a otra matriz en la que en filas tenemos los grupos de genes considerados mientras que el número de muestras se conserva.

<sup>227</sup> Expresión diferencial marginal en este texto.

Trasladamos el problema de expresión diferencial a nivel de gen<sup>227</sup> a un problema de expresión diferencial a nivel de grupos de genes. Y el valor que asociamos a cada grupo cuantifica lo diferente que es, en la muestra considerada, la expresión de los genes del grupo de la expresión de los genes que no pertenecen a este grupo.

Con esta nueva matriz podemos aplicar análisis de expresión diferencial donde en lugar de genes trabajamos con un grupo de genes. De hecho, con una colección de grupos de genes. Esto, obviamente, no tiene ninguna importancia desde el punto de vista estadístico. Podemos aplicar todo lo visto para expresión diferencial marginal.<sup>228</sup>

<sup>228</sup> El adjetivo marginal ahorra se refiere a grupos de genes.

### 17.16.1 GSVA

Este método fue propuesto en [70] y está implementado en el paquete [67, GSVA].

Suponemos  $x = [x_{ij}]_{i=1, \dots, N; j=1, \dots, n}$  la matriz de expresión obtenida después de normalización previa. Tenemos los grupos de genes  $\{S_1, \dots, S_K\}$  entendidos como subconjuntos de las filas de  $x$  ( $S_k \subset \{1, \dots, N\}$ ). El perfil de expresión del gen (en fila)  $i$  será  $x_i = (x_{i1}, \dots, x_{in})$ . El método trabaja con datos de expresión de microarrays ( $\log_2$ ) o bien con los conteos en datos de RNA-seq.

**Estimación kernel de las funciones de distribución** Utilizando el perfil de expresión  $x_i$  proponen realizar una estimación kernel de la función de distribución. Para datos continuos (microarrays) utilizan

$$\hat{F}_{h_i}(x_{ij}) = \frac{1}{n} \sum_{k=1}^n \int_{-\infty}^{\frac{x_{ij}-x_{ik}}{h_i}} \frac{1}{\sqrt{2\pi}} e^{-t^2/2} dt. \quad (17.17)$$

El ancho de banda  $h_i$  lo fijan en  $h_i = \frac{s_i}{4}$  siendo  $s_i$  la desviación estándar de  $x_i$ .

Para datos de conteo (RNA-seq) utilizan un kernel de Poisson. En concreto el estimador kernel de la función de distribución propuesto es

$$\hat{F}_r(x_{ij}) = \frac{1}{n} \sum_{k=1}^n \sum_{y=0}^{x_{ij}} e^{-(x_{ik}+r)} \frac{x_{ik}+r}{y!}, \quad (17.18)$$

tomando  $r = 0.5$  para hacer que la moda del kernel Poisson coincida con la propia observación. Denotamos por  $z_{ij}$  el valor  $\hat{F}_{h_i}(x_{ij})$  o  $\hat{F}_r(x_{ij})$  dependiendo de que trabajemos con microarrays o con conteos.

**Órdenes simétricos** Para cada muestra  $j$  ordenamos los datos  $z_{ij}$  con  $j$  constante. Los órdenes (no los valores ordenados) los denotamos por  $r_{ij}$ , esto es,  $z_{r_1,j} \leq \dots \leq z_{r_N,j}$ . Finalmente los valores  $r_{ij}$  son

a su vez transformados con el fin de conseguir que sean simétrico alrededor del origen. En concreto tomamos:  $\tilde{r}_{ij} = |\frac{N}{2} - r_{ij}|$ .

**Agregación de puntuaciones por muestra** Fijamos  $j$  (una muestra dada) y un grupo de genes  $S_k$ . y se define el estadístico

$$\nu_{jk}(s) = \frac{\sum_{i=1}^s |\tilde{r}_{ij}|^\tau 1_{S_k}(r_{ij})}{\sum_{i=1}^N |\tilde{r}_{ij}|^\tau 1_{S_k}(r_{ij})} - \frac{\sum_{i=1}^s 1_{S_k^c}(r_{ij})}{N - |S_k|}, \quad (17.19)$$

donde  $1_{S_k}(r_{ij}) = 1$  si  $r_{ij} \in S_k$  y 0 en otro caso.  $S_k^c$  es el complementario de  $S_k$  y  $|S_k|$  el cardinal de  $S_k$ .

El valor calculado depende del grupo de genes  $S_k$  y de la muestra  $j$ .

**Resumen** En el apartado anterior tenemos una función de  $s$  que hemos de resumir. La primera propuesta es tomar

$$E_{jk}^{max} = \nu_{jk}(s*). \quad (17.20)$$

con

$$\nu_{jk}(s*) = \max\{\nu_{jk}(s) : s = 1, \dots, N\}. \quad (17.21)$$

Una segunda opción para resumir es considerar

$$E_{jk}^{diff} = \max_{s=1, \dots, N} \{0, \nu_{jk}(s)\} - \min_{s=1, \dots, N} \{0, \nu_{jk}(s)\}. \quad (17.22)$$

Los autores proponen los estadísticos en 17.20 y 17.22.

## 17.17 Otros paquetes y bibliografía complementaria

Un estudio comparativo de distintos métodos de análisis de grupos de genes aparece en [88].

No hemos utilizado en este tema pero son de interés los paquetes [5, topGO] y [50, PGSEA].

La bibliografía sobre este tema es (muy) grande. Como ejemplo, podemos citar como bibliografía complementaria [122, 138, 32, 163, 93, 153, 96, 154, 103, 160, 37, 53].

Una revisión sobre distintos procedimientos para análisis de expresión diferencial basada en conjuntos con datos de RNASeq lo tenemos en [114].



## Capítulo 18

# Enriquecimiento de grafos

En § 16 y § 17 se ha estudiado la expresión diferencial de grupos de genes en una aproximación que tiene dos pasos. Un primer paso que realiza un análisis de expresión diferencial marginal (o gen a gen). Una vez tenemos este grupo pasamos a un segundo paso en el que enriquecemos la señal que cada uno de los genes nos proporciona. No se ha utilizado el conocimiento previo que podemos tener sobre la interacción de los distintos genes. Por ejemplo, su pertenencia a una determinada ruta de señalización. Los genes se han tratado como un conjunto<sup>229</sup> sin considerar relaciones (interacciones) entre ellos que se suelen modelizar utilizando grafos o redes. De esto va este capítulo. De la incorporación de esta información a la hora de estudiar expresión diferencial quizás no de grupos de genes sino de rutas de señalización o sistemas biológicos en los que estos genes intervienen. En la literatura a este problema se le denomina de distintas formas: gene network enrichment analysis, graph-structured test.

Referencias de interés son [88]

Algunas bases de datos en donde tenemos grafos de interacciones entre genes son:

YeastNet <http://www.inetbio.org/yeastnet/><sup>230</sup>

FunCoup <http://funcoup.sbc.su.se/>.

<sup>229</sup> En el sentido matemático de la expresión.

<sup>230</sup> Probabilistic Functional Gene Network of *Saccharomyces cerevisiae*.

### 18.1 Grafos

Un grafo no es más que una colección de vértices y aristas. Siendo un poco más formales el grafo  $\mathbb{G}$  sería el par  $\mathbb{G} = (\mathbb{V}, \mathbb{E})$  donde  $\mathbb{V}$  es el conjunto de vértices mientras que  $\mathbb{E}$  es el conjunto de aristas. Podemos denotar  $\mathbb{G} = \{1, \dots, N\}$ . Una arista será un par  $(i, j)$ . Podemos tener grafos dirigidos, no dirigidos o mixtos.

Los vértices del grafo corresponden con genes (proteínas, grupos de genes).

Una arista dirigida indica que el gen en el vértice origen regula al gen en el vértice destino de la arista.

### 18.2 NEAT

En esta sección consideraremos el procedimiento propuesto en [124] e implementado en [125, neat]. Consideraremos un grafo cuyos vértices

corresponden con genes.

Tenemos un grafo dado describiendo las interacciones entre genes y pretendemos si existe algún tipo de asociación (sobre o infra asociación) entre un par de grupos de genes,  $A$  y  $B$ . El grupo  $A$  podría corresponder a un grupo diferencialmente expresado (obtenido por ejemplo mediante un análisis de expresión diferencial marginal). El conjunto  $B$  puede corresponder a un grupo previamente definido en [Gene Ontology](#) o en [KEGG](#).

Supongamos en un primer momento un grafo dirigido. Denotamos:

$o_A$  El número de aristas salientes de algún vértice de  $A$ .

$i_A$  El número de aristas entrantes a algún vértice de  $A$ .

$n_{AB}$  El número de aristas que salen de un vértice de  $A$  y llegan a algún vértice de  $B$ .

Los autores consideran el siguiente modelo. En ausencia de relación entre los dos grupos de genes tendríamos una urna con  $i_V$  bolas (aristas que llegan a algún vértice en el grafo). De ellas consideramos como bolas rojas las que llegan a un vértice de  $B$  y negras todas las demás. Por tanto en la urna tenemos  $i_B$  bolas rojas e  $i_V - i_B$  bolas negras. Extraemos al azar  $o_A$  bolas de la urna. Suponiendo que ninguna de estas aristas tienen una mayor probabilidad de llegar a otro vértice que no sea de  $B$  entonces el vértice al que llegan podemos considerar que será una extracción aleatoria de la urna considerada. Sea  $N_{AB}$  la variable aleatoria que nos da el total de bolas rojas que obtenemos en las  $o_A$  bolas extraídas. En otras palabras,  $N_{AB}$  cuenta cuántas de las aristas que salen de  $A$  llegan a  $B$ . Bajo la hipótesis nula de que no hay asociación entre ambos conjuntos de genes este número sería una extracción aleatoria del total de aristas y seguiría una distribución hipergeométrica con  $o_A$  extracciones  $i_B$  e  $i_V - i_B$  bolas negras. Esta sería la distribución nula si no hay asociación de ningún tipo entre  $A$  y  $B$ . La media de la hipergeométrica es  $o_A \frac{i_B}{i_V}$ . En [124] se propone contrastar la hipótesis de que la media real de la variable aleatoria  $N_{AB}$  es  $o_A \frac{i_B}{i_V}$ , es decir, contrastar  $H_0 : EN_{AB} = o_A \frac{i_B}{i_V}$  frente a  $H_1 : EN_{AB} \neq o_A \frac{i_B}{i_V}$ . Teniendo en cuenta que  $N_{AB}$  tiene una distribución discreta el p-valor sería  $p = 2 \min\{P(N_{AB} < n_{AB}), P(N_{AB} > n_{AB})\} + P(N_{AB} = n_{AB})$ .

Consideremos el caso de un grafo no dirigido. Sea  $d_A$  la suma de las aristas que indicen con cualquier vértice de  $A$ . Ahora  $N_{AB}$  (respectivamente  $n_{AB}$ ) sería el valor aleatorio (observado) de aristas que unen un vértice de  $A$  con algún vértice de  $B$ . Bajo la hipótesis nula de no asociación entonces  $N_{AB}$  tiene una distribución nula hipérgeométrica con  $d_A$  extracciones,  $d_B$  bolas rojas y  $d_V - d_B$  negras. Por los demás todo es igual.

Cuando tengamos un grafo mixto lo que se hace es las aristas no dirigidas considerarlas como dos aristas dirigidas uniendo los dos vértices en los dos sentidos posibles.

### 18.3 DEGraph

El procedimiento fue propuesto en [79] y está implementado en el paquete [78, DEGraph].

## Método

Se trata de proponer tests multivariantes para estudiar expresión diferencial (básicamente, cambios en el vector de medias) y que sean coherentes con una estructura de grafo dada.

Consideremos un grafo con  $p$  genes. Este grafo es  $\mathbb{G} = (\mathbb{V}, \mathbb{E})$  donde  $\mathbb{V}$  es el conjunto de vértices mientras que  $\mathbb{E}$  es el conjunto de aristas. Consideramos el vector  $\boldsymbol{\delta} \in \mathbb{R}^p$  que nos indicará la diferencia en la expresión media entre dos condiciones.

Sea  $E_{\mathbb{G}}$  una función de energía definida sobre el grafo. ¿Cómo definirla? Veamos distintas posibilidades. Por ejemplo, consideremos la laplaciana del grafo  $\mathbb{L}$ . Si  $\mathbb{G}$  es un grafo no dirigido con matriz de adyacencia  $A^{231}$  y matriz de grados de incidencia  $D = \text{diag}(A\mathbf{1}_p)^{232}$ . El elemento  $i$ -ésimo de la diagonal de  $D$ ,  $D_{ii} = d_i$  es el grado de incidencia del vértice  $i$ . Se define la matriz laplaciana de  $\mathbb{G}$  como  $\mathbb{L} = D - A$  y su versión normalizada como  $\mathbb{L}_{\text{norm}} = I - D^{-1/2}AD^{-1/2}$ .

Supongamos que consideramos la función de energía definida sobre un vector  $\boldsymbol{\delta} \in \mathbb{R}^d$  como

$$E_{\mathbb{G}}(\boldsymbol{\delta}) = \boldsymbol{\delta}' \mathbb{Q}_{\mathbb{G}} \boldsymbol{\delta}, \quad (18.1)$$

donde  $\mathbb{Q}_{\mathbb{G}}$  es  $\mathbb{L}$  entonces

$$E_{\mathbb{G}}(\boldsymbol{\delta}) = \sum_{i,j \in \mathbb{V}} (\delta_i - \delta_j)^2 \quad (18.2)$$

mientras que si utilizamos la versión normalizada tendremos

$$E_{\mathbb{G}}(\boldsymbol{\delta}) = \sum_{i,j \in \mathbb{V}} \left( \frac{\delta_i}{\sqrt{d_i}} - \frac{\delta_j}{\sqrt{d_j}} \right)^2 \quad (18.3)$$

De un modo genérico, si consideremos una matriz semidefinida positiva  $\mathbb{Q}_{\mathbb{G}}$  entonces podemos considerar su descomposición en valores singulares

$$\mathbb{Q}_{\mathbb{G}} = U \Lambda U', \quad (18.4)$$

siendo  $U$  una matriz ortogonal,  $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_p)$  los valores propios correspondientes a los vectores propios que forman las columnas de  $U$ .

Pretendemos reducir la dimensión del vector  $\boldsymbol{\delta}$  a un espacio de dimensión  $k$  mucho menor que  $p$  de forma que se tengan las funciones de menor energía. De un modo preciso, pretendemos encontrar los vectores  $u_i \in \mathbb{R}^p$  con  $i \leq k$  tales que

$$u_i = \underset{v \in \mathbb{R}^p}{\text{argmin}} E_{\mathbb{G}}(v), \quad (18.5)$$

y cada  $u_i$  es ortogonal a todos los anteriores  $u_j$  con  $j < i$ . Estos vectores corresponden a los vectores propios asociados a los  $k$  valores propios más pequeños.<sup>233</sup>

Se consideran grafos dirigidos  $\mathbb{G} = (\mathbb{V}, \mathbb{E})$  donde el conjunto de aristas  $\mathbb{E}$  es un par ordenado de vértices. La matriz de adyacencia  $A$  puede no ser simétrica y  $a_{ij} \neq 0$  si y solamente si  $(i, j) \in \mathbb{E}$  o, de otro modo, una arista va del vértice  $v_i$  al vértice  $v_j$ . Se utiliza la función de energía

$$E_{\mathbb{G}}(\boldsymbol{\delta}) = \sum_{i: d_i^- \neq 0}^p \left( \delta_i - \frac{1}{d_i^-} \sum_{(j,i) \in \mathbb{E}} a_{ij} \delta_j \right)^2, \quad (18.6)$$

<sup>231</sup>  $A = [a_{ij}]_{i,j=1,\dots,p}$  donde  $a_{ij} = 1$  si  $(i, j) \in \mathbb{E}$  y cero en otro caso.

<sup>232</sup>  $\text{diag}(x)$  denota la matriz diagonal con el vector  $x$  en la diagonal y  $\mathbf{1}_p$  es el vector  $p \times 1$  con el valor 1 en cada posición.

<sup>233</sup> Vemos que estamos utilizando los mismos resultados que en componentes principales § 13.

donde  $d_i^- = \sum_{j=1}^p |a_{ji}|$ . Se comprueba que

$$E_{\mathbb{G}}(\boldsymbol{\delta}) = \boldsymbol{\delta}' M_{\mathbb{G}} \boldsymbol{\delta}, \quad (18.7)$$

siendo  $M_{\mathbb{G}} = (\tilde{I} - D_-^{-1} A')' (\tilde{I} - D_-^{-1} A')$ ,  $D_- = \text{diag}(d_1^-, \dots, d_p^-)$  y  $\tilde{I} = \text{diag}(I(d_1^- \neq 0), \dots, I(d_p^- \neq 0))$ <sup>234</sup>. Una vez tenemos esta forma cuadrática podemos hacer su descomposición en valores singulares,  $M_{\mathbb{G}} = U' \Lambda U$ . En lo que sigue si  $x \in \mathbb{R}^p$  entonces se denota  $\tilde{x} = U' x$ .

El procedimiento propuesto consiste en lo siguiente: en lugar de trabajar con los datos originales  $x$  se trabaja con las primeras  $k$  columnas de la matriz  $U$  correspondientes a los  $k$  vectores propios cuyos valores propios son los  $k$  menores. Y se aplica sobre estos datos reducidos un test de la  $T^2$  de Hotelling.<sup>235</sup> Varios comentarios son importantes. Obviamente no podemos aplicar el test de la  $T^2$  de Hotelling por un problema de dimensión. Los autores muestran además la ganancia de potencia en el test de comparación aplicando esta reducción de dimensión. ¿Cómo queda el procedimiento? Denotamos:

1.  $\bar{x}_i$  con  $i = 1, 2$  los vectores de medias en ambas condiciones.

2.  $S$  la matriz de covarianzas estimada conjuntamente<sup>236</sup>

Si aplicáramos el test de  $T^2$  de Hotelling entonces el estadístico del contraste sería

$$T^2 = \frac{n_1 n_2}{n_1 + n_2} (\bar{x}_1 - \bar{x}_2)' S^{-1} (\bar{x}_1 - \bar{x}_2). \quad (18.8)$$

Si en lugar de trabajar en la base original transformamos a una base ortonormal formada por las columnas de  $U$  entonces el estadístico no cambiaría. Si denotamos el estadístico en la nueva base  $\tilde{T}^2$  se tiene que

$$T^2 = \tilde{T}^2 = \frac{n_1 n_2}{n_1 + n_2} (\bar{x}_1 - \bar{x}_2)' U (U' S U)^{-1} U' (\bar{x}_1 - \bar{x}_2). \quad (18.9)$$

Realmente no se trabaja con la nueva base completa sino con  $k$  componentes ( $k \leq p$ ). Denotamos por  $U_{[k]}$  la matriz formada por las primeras  $k$  componentes de  $U$  (correspondientes a los  $k$  menores valores propios) y por  $I_{[k]}$  una matriz con todo ceros salvo los  $k$  primeros elementos de la diagonal principal iguales a uno. Considerando la proyección sobre el espacio generado por las primeras  $k$  componentes el estadístico de Hotelling en el nuevo espacio sería

$$\begin{aligned} \tilde{T}_k^2 &= \frac{n_1 n_2}{n_1 + n_2} (\bar{x}_1 - \bar{x}_2)' U_{[k]} (U_{[k]}' S U_{[k]})^{-1} U_{[k]}' (\bar{x}_1 - \bar{x}_2) = \\ &= \frac{n_1 n_2}{n_1 + n_2} (\bar{x}_1 - \bar{x}_2)' U I_{[k]} U' (U I_{[k]} U' S U I_{[k]} U')^+ U I_{[k]} U' (\bar{x}_1 - \bar{x}_2). \end{aligned} \quad (18.10)$$

donde  $A^+$  es la inversa generalizada de la matriz  $A$ . El estadístico dado en 18.10 es el utilizado en el contraste.

### Ejemplo

```
library(DEGraph)
```

# **Parte VI**

## **Agregación**



## Capítulo 19

# Listas de características ordenadas

237

En temas anteriores hemos visto cómo cuando realizamos un análisis de cada característica marginalmente obtenemos un estadístico y un p-valor asociado. Por ejemplo, en la situación más simple de comparación de dos grupos de muestras (dos condiciones) podemos utilizar un t-test (bilateral). De hecho ordenamos las características según el módulo del estadístico es mayor o bien si el p-valor es más pequeño. Ambos criterios producen la misma ordenación.<sup>238</sup>

En lo anterior estas listas las hemos usado sólamente a fin de determinar en qué posición cortar de modo que por encima del punto de corte declaramos que un test es significativo y por debajo que no lo es. ¿Este es el único uso que podemos realizar de esta ordenación? No, claro. Supongamos que obtenemos 1000 características significativas con un FDR de 0.05. ¿De qué nos sirve? ¿El investigador va a seguir estudiando cada uno de estos genes? No es probable que tenga dinero para ello.<sup>239</sup> Lo más probable es que elija seguir estudiando aquellos genes que aparecen en la parte superior de la lista que hemos construido. Por tanto, la posición en la lista tiene gran importancia.

¿Cómo de estable es la ordenación que hemos obtenido? Cuando se habla de estable hay que definir frente a qué se pretende estabilidad. Una primera fuente de variabilidad a la hora de obtener la lista es utilizar distintos estadísticos (y por lo tanto distintos p-valores) cuando contrastamos la expresión diferencial marginal. Por tanto, estabilidad frente al procedimiento estadístico utilizado. Dentro de este apartado se puede incluir el hecho que los datos hay que preprocesarlos (correcciones de fondo, normalización y resumen en microarrays por ejemplo) y no hay, por supuesto, un único método.

Una segunda fuente de variabilidad son modificaciones en los datos que manejamos, pequeñas modificaciones de los mismos no debieran afectar (demasiado) a la lista obtenida.

En consecuencia, la lista ordenada es variable y hemos de cuantificar la estabilidad de la ordenación. Esto se trata en § 19.1. Un segundo aspecto es cómo agregar distintas listas con objeto de obtener una lista agregada más estable, más fiable. Lo vemos en § 19.2.

<sup>237</sup> Este capítulo está basado fundamentalmente en [20] y su implementación en el paquete [127, GeneSelector]. De hecho, no es más que un resumen. Una buena referencia es [33].

<sup>238</sup> Si es con el estadístico ordenamos de mayor a menor y si es con el p-valor la ordenación se hace de menor a mayor.

<sup>239</sup> Bueno, a lo mejor en Estados Unidos o Alemania. Eso dicen.

## 19.1 Estabilidad de la lista ordenada

Tenemos la matriz de expresión  $x = [x_{ij}]_{i=1,\dots,N; j=1,\dots,n}$  y el vector que describe las muestras  $y = (y_1, \dots, y_n)'$ . Una lista ordenada es una permutación de  $(1, \dots, N)$  que podemos denotar  $\pi(i)$ . El gen  $i$  ocupa la posición  $\pi(i)$  en la lista ordenada. Podemos también denotar por  $(r_1, \dots, r_N)$  la lista ordenada de modo que

$$r_j = i \iff \pi(i) = j.$$

Por ejemplo, si  $\pi(1898) = 1$  significa que el gen cuyo perfil de expresión está en la fila 1989 está en la primera posición de la lista ordenada. Tendremos también que  $r_1 = 1899$ . No vamos a considerar ordenaciones con empates. Normalmente se suele mostrar la lista de los  $k$  primeros genes en la lista<sup>240</sup> donde  $k$  suele ser 20 o 50.

<sup>240</sup> Top-k list.

### 19.1.1 Diferentes criterios de ordenación

Si nos centramos en el caso de dos grupos tenemos muchísimos procedimientos que nos permiten cuantificar la asociación entre la expresión del gen y el fenotipo.

Cargamos el paquete [127, GeneSelector].

```
library(GeneSelector)
```

Utilizamos los datos tamidata::gse1397.

```
library(tamidata)
data(gse1397)
x = exprs(gse1397)
y = pData(gse1397) [, "type"]
```

Obtengamos distintos estadísticos. Empezamos con el fold-change

```
fc = RankingFC(x, y)
```

Calculamos los t-estadísticos.

```
tstat = RankingTstat(x, y)
```

Los t-estadísticos moderados utilizando el modelo limma [128]

```
limma = RankingLimma(x, y, proportion=0.01)
```

Los t-estadísticos de Fox-Dimmic [49].

```
foxdimmic = RankingFoxDimmic(x, y, m=100)
```

El t-estadístico de contracción<sup>241</sup> con

```
shrinkt = RankingShrinkageT(x, y)
```

Nos fijamos para comentar en detalle en una de estas ordenaciones. Veamos qué nos han devuelto.

<sup>241</sup> Shrinkage t-statistic.

```
class(tstat)

## [1] "GeneRanking"
## attr(,"package")
## [1] "GeneSelector"
```

Que tiene los siguientes slots.

```
getSlots("GeneRanking")

##           x      y statistic ranking
##   "matrix"   "factor"  "numeric"  "numeric"
##       pval      type     method
##   "vector" "character" "character"
```

Por ejemplo, en `tstat@x` tenemos la matriz de expresión y `tstat@y` nos proporciona la variable fenotípica que en este caso simplemente nos da el grupo al que pertenece la muestra (en código numérico).

```
tstat@y

##  [1] 1 1 1 1 1 1 1 2 2 2 2 2 2 2
## Levels: 1 2
```

La lista la tenemos en `tstat@ranking`. Veamos los primeros.

```
head(tstat@ranking)

## 1007_s_at    1053_at    117_at    121_at 1255_g_at
## 12194        12433      2870      8757    15880
## 1294_at
## 234
```

Nos indica la posición en la matriz de expresión y también tiene como etiqueta el identificador Affy (AffyID). También podemos ver los t-estadísticos y los correspondientes p-valores

```
head(tstat@statistic)

## 1007_s_at    1053_at    117_at    121_at
## -0.6183648  0.6022344  1.5635785 -0.8738516
## 1255_g_at    1294_at
## 0.3815208  2.9096918

head(tstat@pval)

## 1007_s_at    1053_at    117_at    121_at
## 0.54789263  0.55822637  0.14389203  0.39934953
## 1255_g_at    1294_at
## 0.70948639  0.01308856
```

y recuperar la información sobre el test que hemos utilizado que en este caso es un t-test para muestras independientes.

```
tstat$type
## [1] "unpaired"

tstat$method
## [1] "ordinaryT"
```

La función GeneSelector::toplist nos proporciona un resumen útil con la lista en la primera columna, el estadístico en la segunda y el p-valor en la tercera. Podemos ver las 10 primeras posiciones con

```
toplist(tstat,top=10)

##      index statistic      pval
## 1     346 -10.517552 2.070655e-07
## 2    1170 -10.344123 2.481752e-07
## 3    1853  -8.762265 1.463473e-06
## 4    6303    7.648232 5.939862e-06
## 5    7889   -7.190209 1.101446e-05
## 6   10450    6.365838 3.579137e-05
## 7   10132   -6.020134 6.027785e-05
## 8   17751   -6.013987 6.084819e-05
## 9   16324   -5.798600 8.491862e-05
## 10  12401    5.637777 1.093774e-04
```

Hasta ahora hemos descrito los resultados utilizando el t-test. Hemos utilizado otros tres. Mostremos de un modo conjunto los resultados.

```
fc.top = toplist(fc,top=10,show=FALSE)
tstat.top = toplist(tstat,top=10,show=FALSE)
limma.top = toplist(limma,top=10,show=FALSE)
foxdimmic.top = toplist(foxdimmic,top=10,show=FALSE)
shrinkt.top = toplist(shrinkt,top=10,show=FALSE)

global.top = cbind(fc.top[,"index"],tstat.top[,"index"],limma.top[,"index"],
                   foxdimmic.top[,"index"],shrinkt.top[,"index"])
colnames(global.top) = c("fc","tstat","limma","foxdimmic","shrinkt")
```

Y obtenemos.

```
global.top

##          fc tstat limma foxdimmic shrinkt
##  [1,]  9149   346   346      6303    346
##  [2,] 11148  1170   1170      6468   1170
##  [3,]  4375  1853   1853      328    1853
##  [4,]  8951  6303   6303      651    6303
##  [5,]   888  7889   7889      8566  10132
##  [6,]  3946 10450  10132      9149  17751
##  [7,] 11145 10132  17751     11148  16324
##  [8,] 11150 17751  16324      9620 14969
##  [9,] 11573 16324 14969      5817    650
## [10,] 11659 12401  9555      966   2277
```

¿Tenemos genes que aparecen en las cinco listas consideradas? Podemos observar que no hay ninguno. La función GeneSelector::GeneSelector nos hace el trabajo de comprobar si hay genes presentes en la parte superior según todos los criterios.

```
lista = list(fc,tstat,limma,foxdimmic,shrinkt)
genesel = GeneSelector(lista,maxrank=10)
show(genesel)

## GeneSelector run with gene rankings from the following statistics:
## Foldchange
## ordinaryT
## Limma
## FoxDimmicT
## ShrinkageT
## Number of genes below threshold rank 10 in all statistics:0
```

Sin embargo, excluyendo el primer criterio (que no es precisamente el mejor) obtenemos un resultado distinto.

```
listareducida = list(tstat,limma,foxdimmic,shrinkt)
genesel = GeneSelector(listareducida,maxrank=10)
show(genesel)

## GeneSelector run with gene rankings from the following statistics:
## ordinaryT
## Limma
## FoxDimmicT
## ShrinkageT
## Number of genes below threshold rank 10 in all statistics:1
```

Podemos ver qué gen es.

```
which(slot(genesel,"selected") == 1)
## [1] 6303
```

Repetiendo para los 50 primeros de la lista reducida.

```
genesel = GeneSelector(listareducida,maxrank=50)
show(genesel)

## GeneSelector run with gene rankings from the following statistics:
## ordinaryT
## Limma
## FoxDimmicT
## ShrinkageT
## Number of genes below threshold rank 50 in all statistics:2
```

Y son los siguientes.

```
which(slot(genesel,"selected") == 1)
## [1] 651 6303
```

Y una tentación a evitar. Cuando un investigador ha invertido tiempo y dinero (y necesita publicar desesperadamente) necesita obtener resultados “significativos”. Posiblemente tenía la hipótesis de que algún o algunos genes debieran de estar en la parte superior de la lista. La cosa es fácil. ¿Por qué no elegir estadísticos que lo hagan entre los disponibles o incluso proponiendo uno nuevo? En definitiva elegir los criterios utilizando información previa (a veces, sin mala intención). Obviamente esto sesga los resultados o simplemente los invalida.<sup>242</sup>

<sup>242</sup> Y el comentario es válido para muchos otros procedimientos estadísticos. Siempre hay un evidente sesgo hacia la publicación de los resultados significativos ignorando los que no lo son.

<sup>243</sup> De las pocas que habitualmente tendremos.

<sup>244</sup> Que nos da el número de subconjuntos distintos de tamaño  $k$  que podemos formar con un total de  $n$  elementos.

<sup>245</sup> El método bootstrap consiste en muestrear sobre los propios datos. O dicho de un modo más técnico muestrear de la distribución empírica de las muestras.

<sup>246</sup> Lo que se conoce como ruido blanco.

### 19.1.2 Estabilidad frente a cambios en los datos

¿Qué ocurre con nuestra lista si quitamos alguna muestra?<sup>243</sup> ¿O si cambian un poco los valores? En definitiva, esperamos que pequeñas modificaciones en los datos no alteren en demasiado nuestra ordenación de las características.

Veamos diferentes opciones para modificar los datos que lleva implementados [127, GeneSelector].

La primera opción puede ser elegir un número de muestras  $k$  ( $< n$  el número total de muestras) y *quitarlas* de nuestros datos. El número total de posibilidades será el número combinatorio  $\binom{n}{k}$ .<sup>244</sup> Este procedimiento es conocido como *Jackknife*. Un caso particular sería cuando  $k = 1$  y vamos dejando una muestra fuera cada vez. Sería el procedimiento *leaving-one-out*. Cuando aplicamos este procedimiento podemos considerar las  $\binom{n}{k}$  o bien una muestra elegida al azar si son muchas.

Una segunda opción es utilizar un método bootstrap. En este caso lo que hacemos es elegir de las  $n$  muestras disponibles un total de  $n$  muestras con reemplazamiento. Obviamente alguna de las muestras se puede repetir.<sup>245</sup>

Una tercera opción puede ser añadir ruido aleatorio a los propios datos de expresión. En microarrays podemos añadir ruido normal con media nula y un valor pequeño de la varianza que tiene que tener en cuenta el orden de los datos.<sup>246</sup>

Una cuarta opción es modificar la covariable  $y$ . Supongamos que tenemos un factor experimental (un número finito de grupos) y lo que hacemos es cambiar la etiqueta que identifica el grupo sobre una pequeña cantidad de muestras. En resumen cambiamos el grupo de un pequeño grupo de muestras.

Fijamos la semilla del generador para obtener los mismos resultados.

```
set.seed(1979)
```

Aplicamos jackknife con dejando fuera cada vez una de las muestras originales.<sup>247</sup> Observemos que se impone una restricción a esto y es que cada clase ha de tener un tamaño de, al menos, 6 muestras.

```
leave1out = GenerateFoldMatrix(y = as.numeric(y), replicates = 50,
                                k = 1,minclassize=6)
show(leave1out)

## number of removed samples per replicate: 1
## number of replicates: 14
## constraints: minimum classize for each class: 6
```

<sup>247</sup> En definitiva, un leaving-one-out.

Y la salida la podemos pasar a la función GeneSelector::RepeatRanking.

```
leave1out.tstat = RepeatRanking(tstat, leave1out, scheme="subsampling")
```

Un dibujo de interés es comparar para cada gen su posición original y las que tiene con los nuevos datos perturbados o modificados. Es la figura ??.

```
png(paste0(dirTamiFigures, "ListasOrdenadas25.png"))
plot(leave1out.tstat)
dev.off()
```

También podemos modificar los datos de modo que vamos cambiando la etiqueta de una muestra cada vez.<sup>248</sup>

```
change1.tstat = RepeatRanking(tstat, leave1out, scheme="labelexchange")
```

También generamos muestras bootstrap imponiendo la misma restricción sobre el tamaño de cada clase.

```
boot = GenerateBootMatrix(y = as.numeric(y), replicates = 50,
                           maxties = 3, minclassize=6)
boot.tstat = RepeatRanking(tstat, boot)
```

Podemos añadir ruido a las intensidades observadas.

```
noise.tstat = RepeatRanking(tstat, varlist=list(geneewise=TRUE, factor=1/10))
```

Fijémonos ahora a los resultados obtenidos con las muestras generadas mediante bootstrap.<sup>249</sup>

<sup>248</sup> Lo que significa que asignamos esa muestra al otro grupo.

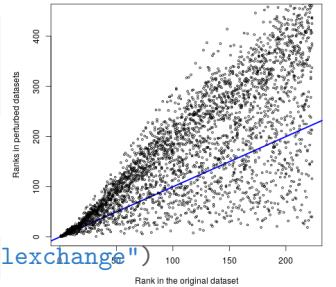
<sup>249</sup> Resultados no mostrados.

```
toplist(boot.tstat, show = FALSE)
```

Vemos que nos muestra la tabla original con los diez primeros genes, el t-estadístico y el p-valor. La siguiente tabla de frecuencias muestra los genes que han aparecido al menos una vez entre los 10 primeros y la frecuencia de veces que aparece en cada una de las 10 primeras posiciones.

En la figura 19.1 mostramos las ordenaciones originales en abscisas frente a las observadas cuando modificamos los datos según los distintos procedimientos. Es más que evidente la tremenda variabilidad que obtenemos en las ordenaciones.

```
par(mfrow=c(2,2))
plot(leave1out.tstat, col="blue",
      pch=". ", cex=2.5, main = "jackknife")
plot(change1.tstat, col="blue",
      pch=". ", cex=2.5, main = "intercambio etiqueta")
plot(boot.tstat, col="blue",
      pch=". ", cex=2.5, main = "bootstrap")
plot(noise.tstat, frac=1/10,
      col="blue", pch=". ", cex=2.5, main = "ruido")
```



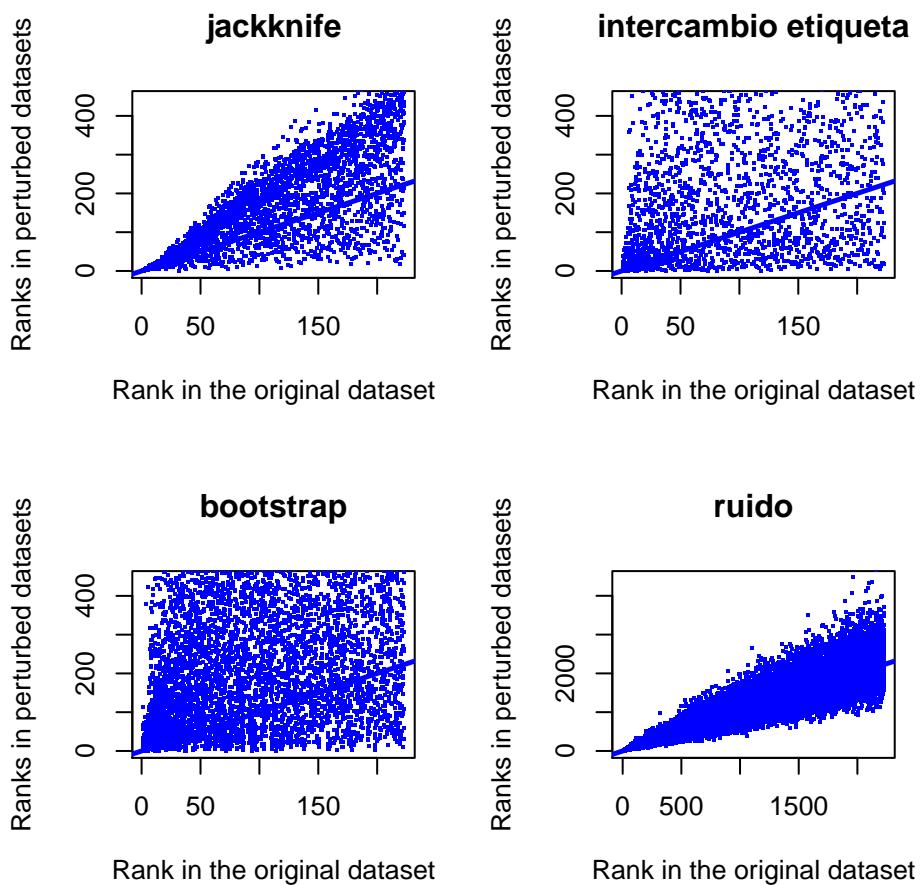


Figura 19.1: Orden original frente a los obtenidos modificando los datos.

## 19.2 Agregación de listas

Hemos obtenido cinco listas distintas correspondientes a distintos estadísticos. Vamos a plantearnos la obtención de una sola lista que las agregue. Lo primero<sup>250</sup> es construir un heatmap. Los datos a clasificar son las distintas ordenaciones. Se realiza una doble clasificación jerárquica con la distancias euclídea<sup>251</sup> y el enlace completo como disimilitud entre grupos.<sup>252</sup>

En la figura 19.2 tenemos el dibujo. Se aprecia cómo el criterio del fold change produce una lista claramente distinta a las demás. También es claro que el t-estadístico y el t-estadístico modera de Limma producen ordenaciones similares. En cuanto a los genes se necesita un estudio mucho más profundo para concluir algo.<sup>253</sup>

```
merged = MergeMethods(lista)
```

```
HeatmapRankings(merged)
```

Hemos de agregar las ordenaciones. La opción más básica sería promediar, por gen, los órdenes obtenidos.

```
AggMean = AggregateSimple(merged, measure = "mean")
```

<sup>250</sup> Porque es bonito aunque tengo mis serias dudas de que sirva para mucho.

<sup>251</sup> Notemos que son órdenes y no está demasiada justificada esta elección.

<sup>252</sup> § 14.

<sup>253</sup> El dibujo es bonito y por eso lo he puesto.

```
merged = MergeMethods(lista)
HeatmapRankings(merged)
```

Figura 19.2: Heatmap para las ordenaciones obtenidas con distintos estadísticos.



# **Parte VII**

# **Probabilidad y Estadística**



# Capítulo 20

## Estadística descriptiva

### 20.1 Datos

Vamos a trabajar con datos de expresión obtenidos mediante micro-arrays para un gen determinado. Tomamos los datos en la fila 3109 de tamidata:gse44456.

```
library(BioBase)
data(gse44456, package="tamidata")
y = exprs(gse44456)[3109,]
```

Es el gen

```
featureNames(gse44456)[3109]
## [1] "7895629"
```

Los valores observados son

```
head(y)
##      GSM1085665_HE32H001.CEL
##      8.339249
##      GSM1085666_HE32H003.CEL
##      7.907181
##      GSM1085667_HE32H004.CEL
##      7.816187
##      GSM1085668_HE32H005.CEL
##      8.232150
##      GSM1085669_HE32H006_2_.CEL
##      7.869529
##      GSM1085670_HE32H007_2_.CEL
##      8.023928
```

Nos fijamos en el grupo a que corresponde cada muestra, si era alcohólico o no.

```
(z = pData(gse44456)[, "case"])
## [1] control    alcoholic control    alcoholic
## [5] control    alcoholic alcoholic control
```

```
## [9] control alcoholic control control
## [13] alcoholic alcoholic control alcoholic
## [17] alcoholic alcoholic alcoholic alcoholic
## [21] alcoholic control alcoholic control
## [25] alcoholic control alcoholic control
## [29] control control alcoholic control
## [33] alcoholic alcoholic control alcoholic
## [37] control control control
## Levels: control alcoholic
```

Vamos a construir un `data.frame` con las expresiones de este gen y la covariable que nos indica la cepa.

```
df = data.frame(x = z, y=y)
```

## 20.2 Introducción

Tenemos unos datos observados en distintas muestras. De un modo genérico denotaremos por  $x_1$  el primer valor observado, por  $x_2$  el segundo valor observado y así sucesivamente. En el lenguaje estadístico a esto se le llama una **muestra**. Por ello, diremos que tenemos una muestra  $x_1, \dots, x_n$  de  $n$  datos. Se entiende que estos datos se han tomado en unas condiciones similares. ¿Cómo son estos datos? Pretendemos describirlos de un modo sencillo. Esta descripción será de dos tipos: una descripción numérica, describimos muchos números con unos pocos números que tengan un sentido claro; y una descripción gráfica. Describimos los números con gráficos que destaque sus propiedades básicas. Al conjunto de técnicas que nos dan descripciones numéricas y gráficas de un conjunto de datos reciben el nombre de **Estadística descriptiva** y, con frecuencia, simplemente **descriptiva**. Se habla de la *descriptiva de los datos*. Veremos que son ideas sencillas pero de un uso constante. Cuando se describen los datos las preguntas básicas que hemos de tener en la cabeza pueden ser:

1. ¿De qué orden son?
2. ¿Cómo de dispersos están?
3. ¿Hay datos anormales que estén muy alejados de los demás?

En la primera pregunta intentamos localizar los valores: ¿estamos alrededor de 2?, ¿o bien alrededor de 20?, ¿o alrededor de 200000? pretendemos localizar la muestra, dar un valor representativo de todos ellos. En la segunda pregunta nos preguntamos si los datos se agrupan si están próximos entre sí. Por último, nos planteamos si tenemos datos que son anormales. Obviamente lo que es anormal depende de cómo son los otros.

## 20.3 Descriptivas numéricas

Empezamos con las descripciones numéricas. De un modo genérico se pretende describir un conjunto de datos numéricos mediante unos

pocos números (elegidos, eso sí, con gracia). En particular vamos a considerar *medidas de localización* y *medidas de dispersión*. Las medidas de localización intentan responder la pregunta que veíamos antes de: ¿de qué orden son los datos? Las medidas de dispersión intentan responder a la segunda pregunta, ¿cómo de dispersos son los datos? ¿Cómo de variables son los datos?

Como medidas de localización veremos la media y medianas muestrales fundamentalmente y como medidas de dispersión básicas veremos la varianza y desviación típica muestrales.

### 20.3.1 Media muestral

La medida de localización más utilizada es la media aritmética o *media muestral* (que es el nombre habitualmente usado en Estadística) que se define como

$$\bar{x} = \sum_{i=1}^n \frac{x_i}{n}. \quad (20.1)$$

La podemos calcular con

```
mean(y)
## [1] 7.890915
```

Notemos que si conocemos solamente la variable y y queremos saber cuántos datos tenemos lo mejor es hacer

```
length(y)
## [1] 39
```

Y, aunque no muy recomendable, otro modo de calcular la media muestral es calcular la suma y dividir por el número de términos que estamos sumando. Esto lo podemos hacer utilizando las funciones *sum* y *length*.

```
sum(y)/length(y)
## [1] 7.890915
```

### 20.3.2 Media ajustada

La media muestral es, sin duda, la mejor forma de localizar una muestra. No obstante es muy sensible a datos anómalos o extremos (y que con frecuencia son errores introducidos en el banco de datos). Por ejemplo, a nuestros datos originales les vamos a añadir un dato anómalo. Le añadimos el valor 34.

```
yy = c(y, 3400)
```

Podemos comparar la media original y la nueva media muestral.

```
mean(y)
## [1] 7.890915

mean(yy)
## [1] 92.69364
```

Se ha modificado muchísimo. Puede que el dato sea real pero puede que sea un error. En cualquier caso es un único valor entre otros muchos. Puede interesarnos localizar nuestra muestra sin atender a estos datos anómalos. ¿Cómo? Una opción simple es fijarnos en una cierta proporción de los datos, por ejemplo, una proporción  $\alpha$  y eliminar el  $\alpha$  por uno de los datos más pequeños y el  $\alpha$  por uno de los más grandes. La media de los que quedan es la *media ajustada*. La media ajustada se obtiene con *mean* indicándole un parámetro adicional.

```
mean(y,trim=.1)
## [1] 7.89261
```

Estamos eliminando el 10% de los datos mayores y el 10% de los datos menores. La media de los restantes es nuestra media ajustada. Ahora podemos comparar la media ajustada de los datos originales y de los datos con la observación anómala.

```
mean(y,trim=.1)
## [1] 7.89261

mean(yy,trim=.1)
## [1] 7.901745
```

Vemos cómo no se ha modificado demasiado. Estamos describiendo la localización de la parte central de los datos despreciando los datos extremos a ambos lados.

### 20.3.3 Percentiles

Otro manera de localizar los datos es utilizar los *percentiles muestrales*. Supongamos que tomamos un valor  $p$  entre 0 y 1. El *percentil de orden  $p$*  es un valor que tiene por debajo el  $100 \times p$  por ciento de los datos y por encima el  $100 \times (1-p)$  por ciento. Denotaremos el percentil de orden  $p$  como  $q_p$ . Ordenamos nuestros datos de menor a mayor con la función *sort*.

```
sort(y)
```

¿Cuántos de los datos originales son menores o iguales a 1? Podemos contar a mano.<sup>1</sup> Otra posibilidad es utilizar la función *ecdf*.

---

<sup>1</sup>Cosa antigua en franco retroceso y que no está prohibido hacer. El inconveniente es cuando tenemos centenares o miles de observaciones.

```
Fn = ecdf(y)
Fn(1)

## [1] 0
```

Vemos pues que la proporción de datos que son inferiores a 1 es de 0. O dicho de otro modo: el valor 1 es el percentil de orden 0 de los datos.

La función básica para calcular los cuantiles es *quantile*. El detalle exacto del procedimiento utilizado para estimar estos valores se puede consultar con *help(quantile)*.<sup>2</sup>

La *mediana muestral* es el percentil de orden 0.5 esto es por debajo tiene al menos la mitad de los datos y por encima la otra mitad. La podemos obtener con

```
median(y)

## [1] 7.880398
```

De hecho, podemos plantearnos cómo conseguir un percentil de orden  $p$  (con  $0 < p < 1$ ) arbitrario. Tomemos  $p = 0.27$ .

```
quantile(y,probs = 0.27)

##      27%
## 7.778698
```

O bien  $p = 0.76$

```
quantile(y,probs = 0.76)

##      76%
## 8.005315
```

Cuando  $p = 0.25$  al percentil le llamamos *cuartil inferior*. Si  $p = 0.75$  tenemos el *cuartil superior*.

```
quantile(y,probs = c(0.25,0.75))

##      25%      75%
## 7.763305 8.001348
```

#### 20.3.4 Varianza y desviación estándar muestrales

Ahora pretendemos cuantificar lo dispersos que están nuestros datos. Las dos medidas más utilizadas son la varianza y la desviación estándar. La *varianza muestral* se define como

$$s^2 = \sum_{i=1}^n \frac{(x_i - \bar{x})^2}{n-1}, \quad (20.2)$$

y la desviación estándar (o típica) muestral se define como

$$s = \sqrt{s^2} = \sqrt{\sum_{i=1}^n \frac{(x_i - \bar{x})^2}{n-1}}. \quad (20.3)$$

---

<sup>2</sup>Se pueden ver hasta nueve procedimientos distintos.

La varianza se puede calcular con

```
var(y)
## [1] 0.05017174
```

y la desviación estándar la obtenemos con

```
sd(y)
## [1] 0.2239905
```

### 20.3.5 Rango

El mínimo y el máximo lo podemos obtener con

```
range(y)
## [1] 7.320905 8.364369
```

o bien con

```
min(y)
## [1] 7.320905
max(y)
## [1] 8.364369
```

El rango, esto es, el máximo valor menos el mínimo valor lo podemos obtener con

```
max(y)-min(y)
## [1] 1.043464
```

o bien con

```
diff(range(y))
## [1] 1.043464
```

### 20.3.6 Rango intercuartílico

Una medida más robusta que el rango es el rango intercuartílico. Se define como la diferencia entre los percentiles de orden 0.75 y 0.25, es decir, el cuartil superior menos el cuartil inferior. Se puede obtener con

```
IQR(y)
## [1] 0.2380429
```

### 20.3.7 La función genérica summary

Es una función que nos proporciona una descripción básica de los datos. En concreto, nos da el mínimo, el primer cuartil, la media, la mediana, el tercer cuartil y el máximo.

```
summary(y)
```

```
##   Min. 1st Qu. Median   Mean 3rd Qu.   Max.
## 7.321 7.763 7.880 7.891 8.001 8.364
```

Sin duda, es la opción más simple para obtener una descripción rápida de los datos.

## 20.4 MAD

Es una forma de cuantificar variabilidad más robusta que la desviación estándar muestral. Tenemos unos datos  $x_1, \dots, x_n$ . Supongamos que  $m$  es una mediana de estos datos. Consideremos los nuevos datos  $|x_i - m|$  con  $i = 1, \dots, n$ , es decir, consideramos el valor absoluto de la diferencia entre cada  $x_i$  y la mediana. El valor de MAD es la mediana de las valores  $|x_i - m|$  con  $i = 1, \dots, n$ , esto es, la mediana de las desviaciones absolutas respecto de la mediana de las observaciones. En resumen,

$$\begin{aligned} MAD(\{x_1, \dots, x_n\}) &= \\ &\text{mediana}(\{|x_i - \text{mediana}(\{x_1, \dots, x_n\})| : i = 1, \dots, n\}) \end{aligned} \quad (20.4)$$

Para entender el párrafo siguiente es necesario leer el tema de variables aleatorias. Usualmente MAD se suele utilizar para estimar la desviación estándar poblacional,  $\sigma$ . Si suponemos que estamos en una población normal, esto es, tenemos valores observados de una variable normal  $X$  con media  $\mu$  y varianza  $\sigma^2$ ,  $X \sim N(\mu, \sigma^2)$ , entonces la media y la mediana coinciden y

$$\frac{1}{2} = P(|X - \mu| \leq MAD) = P\left(\frac{|X - \mu|}{\sigma} \leq \frac{MAD}{\sigma}\right) = P\left(|Z| \leq \frac{MAD}{\sigma}\right) \quad (20.5)$$

siendo  $Z$  una normal estándar,  $Z \sim N(0, 1)$ , con función de distribución  $\Phi$ . Por tanto

$$\Phi(MAD/\sigma) - \Phi(-MAD/\sigma) = \frac{1}{2}.$$

De donde,

$$\frac{MAD}{\sigma} = \Phi^{-1}\left(\frac{3}{4}\right) = 0.67449.$$

Por ello, si pretendemos estimar  $\sigma$  a partir de MAD entonces hemos de considerar

$$\frac{1}{\Phi^{-1}\left(\frac{3}{4}\right)} MAD = 1.4826MAD. \quad (20.6)$$

## 20.5 Ejercicios

**Ej. 32** — Consideremos los siguientes datos.

43.86 33.14 37.04 29.29 21.49 34.98 18.09 18.84 36.20 27.82 22.86 32.11  
 22.45 38.22 44.55 39.03 33.25 18.27 34.44 24.88 24.58 42.12 30.04 19.58  
 34.00 32.98 28.35 25.75 22.78 15.88 38.97 13.47 21.42 34.19 16.49 15.17  
 31.42 17.00 37.06 30.35 19.65 34.62 16.48 19.42 42.89 23.89 29.26 45.64  
 32.29 22.96 29.60 39.98 21.86 18.25 35.96 30.57 40.79 17.21 27.07 28.56  
 15.59 23.51 18.78 37.72 14.40 28.40 43.17 22.65 27.85 41.56 42.44 16.57  
 23.55 29.66 20.72 28.28 42.10 13.76 27.27 19.69 20.18 23.80 14.37 22.02  
 29.06 34.52 21.91 19.98 16.24 44.56 18.54 35.96 30.12 32.82 45.76 28.75  
 32.01 19.39 23.76 41.72 32.90 31.47 15.04 12.74 44.11 38.65 27.18 35.52  
 15.70 38.95 30.59 15.43 45.60 14.98 23.11 22.11 23.03 19.91 34.95 16.05

Se pide:

1. Leer los datos utilizando el método que se prefiera.
2. Calcular la media, mediana, media recortada con una proporción del 0.05, los percentiles de orden 0.1 y 0.9.
3. Supongamos que se han seguido recogiendo datos. En concreto una segunda muestra con los siguientes valores.

123.34 78.23 89.6 1.2

Incorporar estas nuevas observaciones a los datos originales y calcular las descriptivas numéricas anteriores sobre los nuevos datos. Indicar cuáles de ellas varían y cuáles no justificando la respuesta.

## 20.6 Descripciones gráficas de los datos

En esta sección pretendemos ver algunas de las descripciones gráficas para variables numéricas y categóricas.

### 20.6.1 Añadimos variables y seleccionamos casos o variables

Vamos a añadir más información en nuestro banco de datos. Seguimos con las concentraciones de nitratos del ejemplo ???. En concreto se sabe que las muestras de agua se tomaron en dos localizaciones de la Albufera distintas. Las primeras 8 muestras se tomaron en el puerto de Catarroja y las 6 últimas muestras se tomaron en El Palmar. Nos fijamos ahora en las cepas utilizadas en el estudio tamidata::gse44456.

### 20.6.2 Frecuencias

La segunda variable que hemos introducido en el banco de datos es la zona en que tomamos la medida. Es pues una variable categórica que nos indica la pertenencia del dato a una categoría, en este caso, la zona en que se observa el dato. La descripción básica más simple son los *conteos o frecuencias absolutas*. Contamos el número de veces que se repiten cada una de las categorías. Tendremos el número de datos que se ha observado en cada zona. Los obtenemos de un modo simple con la función `table`.

```
table(z)

## z
##   control alcoholic
##      19          20
```

Si dividimos las frecuencias absolutas por el número total de datos tenemos las *frecuencias relativas*. Por ejemplo, con

```
prop.table(table(z))

## z
##   control alcoholic
## 0.4871795 0.5128205
```

O, de otro modo, si sumamos la tabla nos da el total de casos

```
sum(table(z))

## [1] 39
```

y podemos obtener las frecuencias relativas dividiendo los conteos o frecuencias absolutas por esta suma.

```
table(z)/sum(table(z))

## z
##   control alcoholic
## 0.4871795 0.5128205
```

Las frecuencias absolutas y relativas las podemos representar gráficamente con un diagrama de barras. Bien las frecuencias absolutas (figura 20.1) con el siguiente código

```
library(ggplot2)
df = data.frame(z)
png(paste0(dirTamiFigures, "EstDesc27.png"))
ggplot(df, aes(x = z)) + geom_bar() + xlab("strain")
dev.off()
```

o relativas. Como dibujo es el mismo y solamente nos cambia la escala que observamos en ordenadas.

### 20.6.3 Histograma

Para una variable cuantitativa una buena opción para observar la distribución de los datos es un histograma. La idea de un histograma es (demasiado) simple. Si  $x_1, \dots, x_n$  son los datos de los cuales queremos construir el histograma consideramos el intervalo que va del mínimo al máximo, es decir, el intervalo

$$[a, b] = [\min\{x_1, \dots, x_n\}, \max\{x_1, \dots, x_n\}]$$

y lo subdividimos en un número de subintervalos con la misma longitud. A estos subintervalos se les suele llamar *clases*. Supongamos que elegimos  $k$  clases. Entonces los subintervalos que consideramos son

$$[a, a + \delta), [a + \delta, a + 2\delta), \dots, [a + (k - 1)\delta, b]$$

donde

$$\delta = \frac{b - a}{k}$$

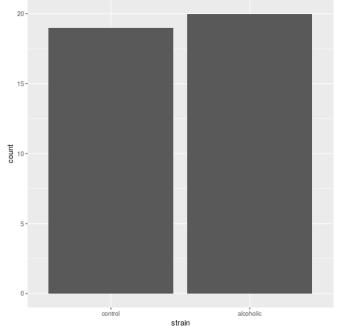


Figura 20.1: Diagrama de barras con las frecuencias absolutas.

Dependiendo del software que utilicemos los valores de  $a$  y  $b$  suelen elegirse como un poco menos que el mínimo y un poco más que el mínimo. El número de clases se elige de un modo automático pero siempre modificable por el usuario. Contamos el número de datos que hay en cada clase. Representamos una barras (que se representan pegadas una con otra lo que también nos permite diferenciarlo de un diagrama de barras) cuya base coincide con el subintervalo y cuya altura es proporcional al número de datos que hemos observado en dicho subintervalo. Este es el dibujo. Veamos cómo hacerlo con R. Si no le indicamos nada el programa decide el número de clases o subintervalos (figura 20.2).



Figura 20.2: Histograma sin elegir número de clases.

<sup>254</sup> Una buena explicación se puede consultar en [http://en.wikipedia.org/wiki/Box\\_plot](http://en.wikipedia.org/wiki/Box_plot).

#### 20.6.4 Diagramas de cajas

Es un dibujo basado en los cuartiles fundamentalmente.<sup>254</sup> La idea es representar una caja como la parte que más destaca en el dibujo. Dentro de la caja se representa con una línea (algo más gruesa habitualmente) la mediana. Los extremos de la caja coinciden con los percentiles del 25% y del 75%. La longitud de la caja es la diferencia de estos percentiles, es decir, el rango intercuartílico. La caja muestra la distribución de la mitad de los datos, la mitad de los datos que están centrados.

Se le añaden unos bigotes que salen de la caja. Son unas líneas que describen la variabilidad de los datos que están en los extremos. Hay varias opciones para elegir el punto al que llegan los bigotes. Entre otras opciones las dos más habituales son:

1. Que lleguen al mínimo y al máximo de los datos.
2. Que lleguen al mínimo y al máximo de los datos que están en el intervalo que tiene por extremo inferior el percentil del 25% menos 1.5 veces el rango intercuartílico y por extremo superior el percentil del 75% mas 1.5 veces el rango intercuartílico. Este valor de 1.5 obviamente podemos modificarlo.

Supongamos que nos planteamos representar un diagrama de cajas de toda la muestra. En la figura 20.3 tenemos el resultado.



Figura 20.3: Diagrama de cajas.

### 20.6.5 Estimadores kernel de la densidad

Es conveniente consultar [http://en.wikipedia.org/wiki/Kernel\\_density\\_estimation](http://en.wikipedia.org/wiki/Kernel_density_estimation). Supongamos  $x_1, \dots, x_m$  los valores observados (de expresión en este caso) el estimador kernel de la densidad en cada punto  $x$  se calcula utilizando la siguiente expresión

$$\hat{f}(x) = \sum_i^n \frac{1}{nh} K\left(\frac{x_i - x}{h}\right),$$

donde la función  $K$  es no negativa y verifica

$$\int_{-\infty}^{+\infty} K(u)du = 1,$$

y

$$K(-u) = K(u),$$

es decir, es simétrica respecto del origen. Una función  $K$  que se suele utilizar es la gaussiana dada por

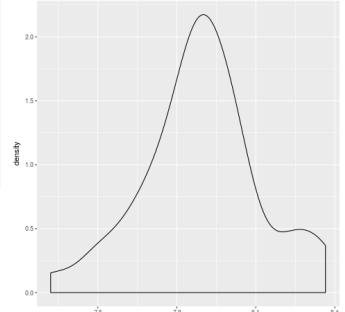
$$K(u) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}u^2}.$$

Otros ejemplos de funciones kernel que podemos usar las podemos ver en [http://en.wikipedia.org/wiki/Kernel\\_%28statistics%29](http://en.wikipedia.org/wiki/Kernel_%28statistics%29).

En la figura 20.4 aparece un estimador kernel de la densidad utilizando una función kernel gaussiana.

```
df = data.frame(y)
png(paste0(dirTamiFigures, "EstDesc32b.png"))
ggplot(df, aes(x=y)) + geom_density()
dev.off()

## pdf
## 2
```



### 20.6.6 Buscando datos anómalos

Tenemos unos datos numéricos  $x_1, \dots, x_n$  y queremos saber si hay alguno que se sale de madre. Si hay alguno que está muy alejado de los demás. Que es anómalo. Lo primero es precisar qué entendemos por dato anómalo.<sup>3</sup> Vamos a ver las dos definiciones más habitualmente utilizadas. En la primera utilizamos media y desviación estándar. En la segunda utilizamos cuartiles.

La primera es la que más tradición tiene. Dados los datos calculamos su media y desviación típica muestrales:  $\bar{x}$  y  $s$ . Se entiende por dato anómalo aquél que está fuera del intervalo

$$[\bar{x} - 3s, \bar{x} + 3s],$$

es decir, que o bien es extremo porque es menor que  $\bar{x} - 3s$  o bien es extremo porque es mayor que  $\bar{x} + 3s$ . ¿Tenemos datos anómalos en este sentido en y?

Figura 20.4: Estimador kernel de la función de densidad.

---

<sup>3</sup>La expresión inglesa es outlier.

```
y[y < mean(y) - 3 * sd(y)]
```

```
## named numeric(0)
```

```
y[y > mean(y) + 3 * sd(y)]
```

```
## named numeric(0)
```

Vemos que detectamos uno de ellos.

El segundo procedimiento utiliza los cuartiles. Denotemos por  $q_{25}$  y  $q_{75}$  los percentiles de orden 25% y 75%, esto es, los cuartiles inferior y superior. El rango intercuartílico sería

$$IQR = q_{75} - q_{25}.$$

La segunda forma de considerar un dato como extremo es considerar que lo es si está fuera del intervalo

$$[q_{25} - 1.5 \times IQR, q_{75} + 1.5 \times IQR].$$

Puede ser extremo por abajo si es menor que  $q_{25} - 1.5 \times IQR$  o por arriba si es mayor que  $q_{75} + 1.5 \times IQR$ . Determinemos los extremos del intervalo.

```
(lw = quantile(y, probs=0.25) - 1.5 * IQR(y))
```

```
##      25%
```

```
## 7.40624
```

```
(up = quantile(y, probs=0.75) + 1.5 * IQR(y))
```

```
##      75%
```

```
## 8.358412
```

Y veamos si hay puntos extremos por abajo

```
y[y < lw]
```

```
## GSM1085686_HE32H023_2_.CEL
```

```
## 7.320905
```

Detecta el punto añadido por nosotros. Y por arriba.

```
y[y > up]
```

```
## GSM1085697_HE32H034.CEL
```

```
## 8.364369
```

## 20.7 Ejercicios

**Ej. 33** — Vamos a realizar distintas representaciones gráficas con los datos del ejercicio 32. Se pide lo siguiente:

- Realizar distintos histogramas de los datos que aparecen en el ejercicio 32 modificando el número de clases. ¿Hay un comportamiento consistente en la representación gráfica?

2. Representar gráficamente un estimador kernel de la densidad. Observar el valor que se ha utilizado para el ancho de banda.
  3. Modificar el valor del ancho de banda observado en el apartado 2 doblando su valor y volver a representar el estimador kernel de la densidad.
  4. Modificar el valor del ancho de banda observado en el apartado 2 considerando la mitad de su valor y volver a representar el estimador kernel de la densidad.
  5. Comparar los tres estimadores kernel que hemos obtenido. ¿Qué ocurre cuando incrementamos el ancho de banda? ¿Y cuando lo disminuimos?

**Ej. 34** — Consideramos los datos del ejercicio 32. La muestra inicial la denotamos por  $x$  mientras que los datos ampliados los denotamos por  $xx$ . Supongamos que los datos  $x$  han sido obtenidos en distintas localizaciones. En concreto las localizaciones las tenemos codificadas de un modo numérico. Son las siguientes.

Se pide:

1. Introducir estos datos en un vector en R que denotaremos por  $y$  utilizando la función `c()` de concatenación.
  2. Realizar un diagrama de cajas de la variable  $x$ .
  3. Realizar diagramas de barras de los valores que tenemos en la variable  $x$  para las distintas localizaciones que nos vienen indicadas por la variable  $y$ .
  4. Los datos adicionales que aparecen en el vector  $xx$  han sido obtenidos en una cuarta localización. Completar el vector  $yy$  para incluir esta localización, es decir, en  $xx$  tenemos los datos ampliados mientras que en  $yy$  tendremos las localizaciones ampliadas.
  5. Muestra un diagrama de barras comparativo de los valores  $xx$  para las distintas localizaciones que aparecen en  $yy$ .

Ej. 35 — x

y

27.60	24.02	33.97	27.84	33.12	37.32	37.53	38.95	29.80	32.72	30.04	26.45
26.34	32.82	28.91	29.37	32.39	29.43	37.83	24.46	37.82	32.19	34.51	32.64
30.44	38.70	29.84	29.35	32.78	34.01	36.24	41.86	35.96	35.57	33.84	27.69
29.32	41.71	34.08	27.64	33.06	39.98	36.62	29.72	33.51	31.49	33.51	33.24
25.02	39.78	31.96	37.69	44.01	29.07	32.94	30.47	33.33	24.34	35.99	32.25
36.51	33.47	35.37	31.82	38.49	25.67	29.36	36.64	24.14	39.54		

Se pide:

1. Representar en dos gráficos distintos los estimadores kernel de ambas densidades.

- 2.Repetir el apartado anterior pero en la misma gráfica.
- 3.Representar las funciones de distribución de la primera muestra.  
Haced lo mismo con la función de distribución de la segunda muestra.
- 4.Representad las dos funciones de distribución en un mismo gráfico.

# Capítulo 21

## Probabilidad

255

### 21.1 Experimento, suceso y probabilidad

Nos movemos constantemente en situaciones en que no podemos predecir qué va a ocurrir.<sup>1</sup> Realizamos un viaje en coche entre dos ciudades dadas. El tiempo que tardamos en realizar el viaje depende de muchos factores que hacen incierto el resultado. Nos sometemos a una operación quirúrgica. El resultado también es incierto. Quizás los ejemplos más simples y, por ello, los que se suelen utilizar están asociados con juegos de azar. Lanzamos un dado, el resultado que obtenemos no lo podemos predecir. Y el número de ejemplos es infinito. Nos encontramos en situaciones en donde no podemos predecir lo que va a ocurrir.

**Experimento** Estas situaciones son lo que llamamos *experimentos*. Dado un conjunto de condiciones no podemos predecir el resultado. Los experimentos más simples de analizar suelen estar ligados a juegos de azar.

1. Un experimento puede ser lanzar una moneda en donde no podemos predecir si el resultado va a ser que sale una cara o bien que sale una cruz.
2. El experimento puede ser lanzar un dado. Aquí tenemos seis posibles resultados (cada una de las caras del dado). También es un experimento con resultado incierto si jugamos a la lotería. Aquí el resultado será el número que sale premiado en un sorteo.

**Espacio muestral** En estos ejemplos sabemos qué cosas pueden pasar, el conjunto de posibles resultados al que se le llama *espacio muestral* y se suele denotar con la letra  $\Omega$ . Sin embargo, cuando realizamos el experimento no sabemos exactamente *qué* va a pasar. No sabemos qué resultado entre los posibles se va a producir exactamente.

En el experimento consistente en lanzar una moneda tenemos dos posibles resultados: el resultado cara y el resultado cruz. Por ello podemos denotar el espacio muestral como

$$\Omega = \{cara, cruz\}.$$

<sup>1</sup> Lo cual no es de nuestro agrado. Pero la vida es así.

255 - Is he in?  
Is he in?  
Nathe!  
He can't hear you through  
your...  
triple bloody glazing.  
Script: Full Monty (1997)

Si lanzamos el dado podríamos representar el espacio muestral como

$$\Omega = \{1, 2, 3, 4, 5, 6\},$$

donde el número 2 significa que cuando lanzamos el dado *sale el número 2* y así sucesivamente.

En una lotería en la que se juegan 15000 números los posibles resultados son cada uno de estos números. Por tanto, el espacio muestral sería

$$\Omega = \{1, 2, \dots, 15000\}.$$

**Suceso aleatorio** Se llama suceso aleatorio a cualquier conjunto de resultados. Por ejemplo, si lanzamos un dado, un suceso puede ser saber si obtenemos un número par. Este suceso es el conjunto formado por tres resultados. Lo representamos como

$$A = \{2, 4, 6\}.$$

Nos puede interesar saber si el número que obtenemos es un valor mayor o igual a 4. Este es el suceso

$$A = \{4, 5, 6\}.$$

Dado un suceso que nos interesa nuestro problema es saber cómo de probable es que se produzca. Y conocer esto antes de realizar el experimento.

**Probabilidad** Consideremos el experimento en que lanzamos una moneda. Conocemos el conjunto de resultados que se pueden producir, el espacio muestral:  $\Omega = \{1, 2, \dots, 6\}$ . Nos planteamos la siguiente pregunta: ¿Qué probabilidad tengo de obtener un número par? Si el dado está bien construido no esperamos que un resultado ocurra con más frecuencia que otro. No esperamos que un resultado sea más probable que otro. Estamos usando la palabra *probabilidad* en su uso del lenguaje común. El término se usa de un modo impreciso pero comprensible por todos. Diríamos que los resultados son equiprobables o que tienen la misma probabilidad. En un experimento no queremos saber qué va a pasar. No tiene mucho sentido plantearnos esto porque es aleatorio sino que nos conformamos con saber la *probabilidad de lo que puede pasar*. En el ejemplo del dado parece razonable decir que la probabilidad de un número par sería

$$P(\text{Obtenemos un número par}) = P(\{2, 4, 6\}) = \frac{3}{6}.$$

¿Por qué? Si todos los resultados son equiprobables entonces la probabilidad parece razonable considerar que sería el número de resultados que corresponden con *salir un número par* dividido por el número de resultados que se pueden producir. En resumen, el número de resultados que son *favorables* a lo que me interesa (que salga un número par) dividido por el número total de resultados. En el mismo experimento: ¿qué probabilidad tenemos de un número mayor o igual a 4?

$$P(\text{Mayor o igual a 4}) = P(\{5, 6\}) = \frac{3}{6}.$$

¿Qué probabilidad tenemos de obtener una cara al lanzar una moneda? Por el mismo razonamiento:

$$P(\text{Cara}) = P(\{\text{Cara}\}) = \frac{1}{2}.$$

Un resultado es favorable y dos resultados son posibles cuando lanzamos la moneda.

Tenemos en R una función que nos sirve para elegir al azar de un grupo de elementos previamente definidos y de un modo equiprobable entre los posibles resultados del experimento. Es la función *sample*.

**Ejemplo 21.1 (Lanzamos una moneda muchas veces)** *Veamos cómo lanzar una moneda con R. Le decimos cuál es el espacio muestral*

```
Omega = c("cara", "cruz")
```

*y luego elegimos uno al azar en el espacio muestral.*

```
sample(Omega, 1)
```

```
## [1] "cruz"
```

*Volvamos a lanzar la moneda.*

```
sample(Omega, 1)
```

```
## [1] "cruz"
```

*Y una tercera vez.*

```
sample(Omega, 1)
```

```
## [1] "cara"
```

*De continuar iríamos obteniendo una serie de resultados cara o cruz. Lancemos 30 veces la moneda y veamos qué pasa.*

```
sample(Omega, 30, replace=TRUE)
```

```
## [1] "cruz" "cara" "cara" "cruz" "cara" "cruz"
## [7] "cara" "cara" "cara" "cruz" "cara" "cara"
## [13] "cruz" "cara" "cruz" "cruz" "cruz" "cara"
## [19] "cruz" "cara" "cara" "cruz" "cruz" "cara"
## [25] "cara" "cruz" "cruz" "cara" "cara" "cruz"
```

*Y otras 30 veces.*

```
sample(Omega, 30, replace=TRUE)
```

```
## [1] "cara" "cara" "cara" "cara" "cara" "cruz"
## [7] "cara" "cruz" "cruz" "cruz" "cara" "cruz"
## [13] "cara" "cruz" "cara" "cruz" "cruz" "cara"
## [19] "cruz" "cruz" "cara" "cara" "cara" "cruz"
## [25] "cruz" "cruz" "cara" "cruz" "cara" "cara"
```

*Podemos contar cuántas veces nos ha salido cara y cruz (el que quiere puede hacerlo manualmente).*

```
x = sample(Omega, 30, replace=TRUE)
table(x)

## x
## cara cruz
## 16 14
```

*Si dividimos por el total de lanzamientos tenemos la frecuencia relativa de veces que nos ha salido cada uno de los dos posibles resultados. En nuestro caso tenemos las siguientes frecuencias relativas observadas:*

```
table(x) / 30

## x
##      cara      cruz
## 0.5333333 0.4666667
```

*Vamos a lanzar 100 veces la moneda y calculamos las frecuencias relativas.*

```
x = sample(Omega, 100, replace=TRUE)
table(x) / 100

## x
## cara cruz
## 0.52 0.48
```

*¿Y por qué no lanzar la moneda 1000 veces?*

```
x = sample(Omega, 1000, replace=TRUE)
table(x) / 1000

## x
## cara cruz
## 0.522 0.478
```

*Y para acabar con la experiencia vamos a lanzarla 100000 veces.*

```
x = sample(Omega, 100000, replace=TRUE)
table(x) / 100000

## x
## cara cruz
## 0.49986 0.50014
```

*Es claro que conforme repetimos el experimento y observamos la frecuencia relativa de veces que ocurre cada resultado nos acercamos cada vez más al valor 0.5 para cada uno de los posibles resultados.*

*En la figura 21.1 representamos en el eje de abscisas el número de veces que lanzamos la moneda y en ordenadas la frecuencia relativa de veces que se ha observado cara. Podemos ver cómo las frecuencias relativas de aparición de cara van apróximándose al valor 0.5.*

```
## pdf
## 2
```

Supongamos que nos fijamos en la frecuencia relativa de aparición de las cruces. En la figura 21.2 representamos en abscisas el número de lanzamientos y en ordenadas la frecuencia relativa de aparición de cruces. Vemos cómo se estabiliza la frecuencia alrededor del valor 0.5.

**Ejemplo 21.2 (Lanzamiento de un dado)** ¿Cómo lanzamos un dado con R? Pues la función sample es adecuada. Empezamos definiendo el espacio muestral.

```
(Omega = 1:6)
```

```
## [1] 1 2 3 4 5 6
```

Y ahora lanzamos el dado.

```
sample(Omega, 1)
```

```
## [1] 3
```

O bien lo lanzamos 20 veces.

```
sample(Omega, 20, replace=TRUE)
```

```
## [1] 6 1 4 1 5 4 4 1 3 4 3 6 5 2 1 1 5 6 5 3
```

Esperamos que cuando lo lanzamos un gran número de veces la frecuencia de veces que ocurre cada resultado se aproxime a  $\frac{1}{6}$ .

```
x = sample(Omega, 1000, replace=TRUE)
table(x) / 1000
```

```
## x
##   1    2    3    4    5    6
## 0.176 0.168 0.167 0.176 0.147 0.166
```

Como así podemos comprobar.

En este tipo de experimentos con resultados equiprobables la probabilidad de cualquier suceso  $A$  viene dada por el cociente entre el número de resultados que tiene  $A$  (a los que se suele llamar *casos favorables* a la ocurrencia de  $A$ ) y el número total de resultados o *casos posibles*.

Si denotamos el cardinal o número de elementos de  $A$  con  $|A|$  entonces, cuando los resultados son equiprobables, podemos definir la probabilidad del suceso  $A$  como

$$P(A) = \frac{|A|}{|\Omega|} = \frac{\text{Casos favorables a que ocurra A}}{\text{Casos posibles}}. \quad (21.1)$$

Se suele conocer como la definición de probabilidad de Laplace. Describe con precisión situaciones como las descritas asociadas a algunos juegos de azar y algún otro caso de interés pero, desde luego, no todos los posibles experimentos en que estamos interesados.

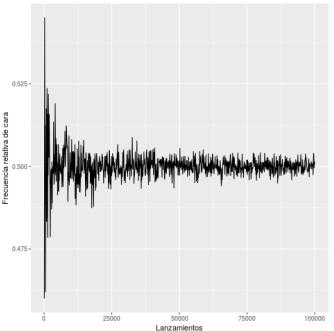


Figura 21.1: Frecuencias relativas de aparición de cara en sucesivos lanzamientos de una moneda correcta.

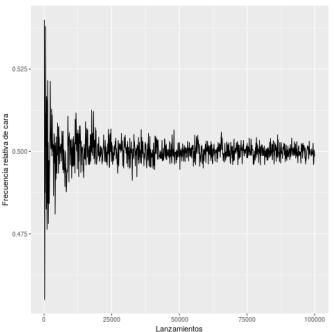


Figura 21.2: Frecuencias relativas de aparición de cruz en sucesivos lanzamientos de una moneda correcta.

Observemos que, en la definición de Laplace, si tomamos un suceso formado por un solo resultado entonces

$$P(\{\omega\}) = \frac{1}{|\Omega|} = \frac{1}{\text{Casos posibles}}. \quad (21.2)$$

La probabilidad de cara es  $1/2$ . La probabilidad de cruz también es  $1/2$ . La probabilidad de un seis cuando lanzamos un dado es  $1/6$ . También lo es la de obtener un 5. O un 4. Finalmente vemos que

$$P(A) = \sum_{\omega \in A} \frac{1}{|\Omega|}.$$

### 21.1.1 Contando: variaciones, permutaciones y combinaciones

En experimentos con resultados equiprobables hay que contar el número total de resultados y el número de casos que contiene el suceso aleatorio de interés, esto es, hemos de contar casos favorables y casos posibles.

Para ellos es fundamental un breve repaso de combinatoria: variaciones, permutaciones y combinaciones.

**Variaciones sin repetición** Supongamos que tenemos un conjunto de  $n$  elementos (por ejemplo, el conjunto  $\{1, \dots, n\}$ ) y pretendemos saber cuántas secuencias ordenadas de  $k$  elementos (con  $k < n$ ) podemos formar. Para entender el problema supongamos que  $n = 3$  y  $k = 2$ . Entonces las posibles secuencias serían

12 21 13 31 23 32

Tenemos seis secuencias porque consideramos distintas 12 y 21. ¿Cómo contarlas sin enumerarlas todas ellas? El razonamiento es sencillo. Para la primera posición tenemos 3 posibilidades. Una vez elegido el número que ocupa la primera posición nos quedan 2 posibilidades para el segundo. En total  $3 \times 2$ .

En general, si consideramos  $n$  y  $k$  tenemos  $n$  posibilidades para el primero. Dada la elección nos quedan  $(n - 1)$  elecciones para el segundo. Dadas las dos primeras elecciones nos quedan  $(n - 2)$  elecciones para el tercero. Para la última posición nos quedarán  $(n - k + 1)$  posibles elecciones. En total tendremos

$$n \times (n - 1) \times \dots \times (n - k + 1).$$

Esto recibe el nombre de **variaciones sin repetición**.

**Permutaciones** ¿De cuántas maneras podemos ordenar  $n$  elementos distintos? Podemos seguir con el razonamiento del párrafo anterior. Cuando ordenamos  $n$  elementos tenemos que elegir el primero de ellos, con  $n$  posibles elementos. Una vez tenemos el primero, para el segundo tenemos  $n - 1$  y así sucesivamente. Podemos ver que tenemos variaciones sin repetición donde  $n = k$ . Por tanto el número total de **permutaciones** es

$$n! = n \times (n - 1) \times \dots \times 1.$$

**Combinaciones** ¿Cuántos conjuntos distintos de  $k$  elementos podemos formar con un total de  $n$ ? Estamos en una situación similar a las variaciones sin repetición salvo que no queremos que intervenga el orden. Por ejemplo, las secuencias

12 21

son distintas como secuencias. Sin embargo, los conjuntos

{1, 2} {2, 1}

son el mismo. Una vez hemos elegido los elementos de un conjunto (por ejemplo, los elementos 1 y 2) hemos de plantearnos cuántas secuencias ordenadas podemos formar,  $2!$  en el ejemplo. En resumen, de un conjunto de  $k$  elementos podemos formar  $k!$  secuencias (ordenadas) distintas. Tenemos  $n$  elementos. Con ellos podemos formar  $n \times (n - 1) \times \dots \times (n - k + 1)$  secuencias ordenadas. Pero cada  $k!$  de estas secuencias tenemos los mismos elementos. En resumen, el número de conjuntos distintos de  $k$  elementos que podemos formar con  $n$  elementos distintos es

$$\binom{n}{k} = \frac{n \times (n - 1) \times \dots \times (n - k + 1)}{k!}.$$

Fácilmente podemos comprobar que

$$\binom{n}{k} = \frac{n!}{k!(n - k)!}.$$

El número  $\binom{n}{k}$  recibe el nombre de **combinaciones de  $n$  elementos tomados de  $k$  en  $k$** .

**Ejemplo 21.3 (Factorial y combinaciones)** Dado un valor de  $n$ , por ejemplo,  $n = 10$  podemos obtener el valor de  $10!$  con

```
factorial(10)
## [1] 3628800
```

Las combinaciones de  $n$  elementos tomados de  $k$  en  $k$  para  $n = 10$  y  $k = 5$  las obtenemos con

```
choose(10, 5)
## [1] 252
```

Obviamente estamos eligiendo 5 elementos de un total de 10. De ahí el nombre de la función.

**Ejemplo 21.4 (Póquer)** Supongamos el póquer cerrado sin comodines. Nos planteamos la probabilidad de que nos sirvan en una mano exactamente una pareja. Estamos en una situación de resultados equiprobables. Los resultados posibles son todos los posibles subconjuntos de 5 elementos del total de 52 cartas. Por tanto, el número de manos distintas será

$$\binom{52}{5}.$$

Contemos ahora el número de manos que contienen exactamente una pareja (y no dos parejas o un trío). Vamos contando. Primero elegimos el número del cual formamos la pareja. Tenemos 13 posibilidades. Una vez elegido el palo tenemos cuatro cartas con el mismo número, por tanto,  $\binom{4}{2}$  posibles parejas con ese número. Ahora hemos de elegir los otros tres números que nos aparecen en la mano. Tenemos 12 números disponibles (quitamos el que hemos utilizado para formar la pareja) y elegimos tres números con un total de  $\binom{12}{3}$  posibilidades. Pero una vez elegidos los números tenemos cuatro cartas de cada número. Por lo tanto, por cada elección de números tenemos  $4^3$ . En total como casos favorables nos encontramos con  $13\binom{4}{2}\binom{12}{3}4^3$  casos favorables. La probabilidad buscada es

$$\frac{13\binom{4}{2}\binom{12}{3}4^3}{\binom{52}{5}}.$$

La podemos calcular con R.

```
(casosfavorables = 13 * choose(4,2) * choose(12,3) * 4^3)
## [1] 1098240

(casosposibles = choose(52,5))
## [1] 2598960

casosfavorables / casosposibles
## [1] 0.422569
```

Tenemos una probabilidad de 2.366.

Esto es como se debe de hacer. Y ahora como no se debe de hacer. Vamos a jugar con R. Empezamos definido las cartas que tenemos

```
(cartas = rep(1:13,4))
##  [1]  1  2  3  4  5  6  7  8  9 10 11 12 13  1  2
## [16]  3  4  5  6  7  8  9 10 11 12 13  1  2  3  4
## [31]  5  6  7  8  9 10 11 12 13  1  2  3  4  5  6
## [46]  7  8  9 10 11 12 13
```

Y ahora extraemos al azar una mano.

```
(mano = sample(cartas,5))
## [1] 11 10 12  8 11
```

¿Cómo sé que tenemos una pareja? Una forma sencilla es contar la frecuencia de cada número.

```
(conteosmano = table(mano))
## mano
##  8 10 11 12
##  1  1  2  1
```

Y ver cuántos conteos me devuelve.

```
length(conteosmano)
## [1] 4
```

*Si me devuelve 5 quiere decir que no se repite ningún número. Si me devuelve 4 quiere decir que hay una pareja exactamente. Contemos pues el número de veces que se produce esta situación y repitamos la experiencia muchas veces. La frecuencia relativa de éxitos nos ha de dar la probabilidad. Un poco de código más complicado de R.*

```
nsimulaciones = 1000
exitos = 0
for(i in 1:nsimulaciones){
  mano = sample(cartas,5)
  conteosmano = table(mano)
  if(length(conteosmano) == 4) exitos = exitos + 1
}
exitos / nsimulaciones

## [1] 0.416
```

### 21.1.2 Un poco de teoría

<sup>2</sup>

En lo anterior hemos visto ejemplos de experimentos aleatorios donde todos los resultados son *equiprobables* y por ello la probabilidad de cada suceso no era más que el número de elementos que contiene (casos favorables) dividido por el total de resultados posibles (casos posibles). ¿Este es el único caso que nos encontramos de experimento aleatorio? Desde luego que no. De hecho, una probabilidad es cualquier función que a los sucesos les da valores entre 0 y 1 verificando algunos principios (o axiomas) razonables.

**Definición 21.1 (Probabilidad)** Una función de conjunto,  $P$ , definida sobre los sucesos es una probabilidad si verifica

1.  $P(A) \geq 0$  para todo suceso  $A$ .
2.  $P(\Omega) = 1$ .
3.  $P$  es  $\sigma$ -aditiva, es decir, si  $\{A_n\}_{n \geq 1}$  son sucesos disjuntos dos a dos entonces

$$P\left(\bigcup_{n \geq 1} A_n\right) = \sum_{n \geq 1} P(A_n).$$

A partir de la definición anterior se deducen algunas propiedades muy útiles.

1. La probabilidad del vacío es cero:  $P(\emptyset) = 0$ .
2. Si tenemos sucesos tales que  $A \subset B$  entonces  $P(A) \leq P(B)$ .

---

<sup>2</sup> Esta sección incluye algo de teoría. Su lectura no es muy necesaria para seguir el curso aunque sí que es conveniente. No hay nada tan práctico como la teoría.

3. Si los sucesos  $A_1, \dots, A_n$  no son disjuntos entonces

$$\begin{aligned} P\left(\bigcup_{i=1}^n A_i\right) &= \\ \sum_{i=1}^n P(A_i) - \sum_{i < j} P(A_i \cap A_j) + \dots + (-1)^{n+1} P(A_1 \cap \dots \cap A_n) \end{aligned}$$

En particular si tenemos dos sucesos

$$P(A_1 \cup A_2) = P(A_1) + P(A_2) - P(A_1 \cap A_2). \quad (21.4)$$

4. A partir del punto anterior es inmediato que

$$P(A^c) = 1 - P(A). \quad (21.5)$$

5. Dados los sucesos  $A_1, \dots, A_n$ , la relación existente entre la probabilidad de la unión de los  $A_i$  y la probabilidad de cada uno de ellos es la siguiente:

$$P\left(\bigcup_{i=1}^n A_i\right) \leq \sum_{i=1}^n P(A_i).$$

## 21.2 Ejercicios

### Ejercicio 36

Seis personas se sientan a comer en un restaurante. Hay seis sillas alrededor de la mesa.

1. ¿De cuántas formas distintas pueden sentarse?
2. Supongamos que las sillas están enfrentadas, tres en un lado de la mesa y otras tres al otro lado. Además las seis personas son tres parejas que han quedado a cenar. Siguiendo una antigua costumbre se sientan a un lado los hombres y al otro lado las mujeres.  
¿De cuántas formas distintas se pueden sentar?

### Ejercicio 37

Supongamos el póquer cerrado sin comodines. Calcular la probabilidad de obtener un póquer cuando nos dan una mano.

### Ejercicio 38

Consideremos el experimento aleatorio consistente en lanzar dos veces un dado. Se pide:

1. ¿Qué probabilidad tenemos de obtener dos veces el número 6?
2. ¿Y de obtener el par de valores 1 y 5?
3. ¿Qué probabilidad tenemos de que coincidan el primer y el segundo resultado, esto es, de que el primer lanzamiento sea un uno y el segundo también o de que el primer lanzamiento sea un dos y el segundo también, etc?
4. ¿Qué probabilidad tenemos de que la suma de los dos valores sea 7? ¿Y de que la suma de los dos valores sea mayor o igual que 7?  
¿Y de que sea mayor que 7?

## 21.3 Variable aleatoria

Supongamos el experimento consistente en elegir al azar o aleatoriamente a una individuo de la Comunidad Valenciana. Obviamente el espacio muestral está formado por los distintos individuos. Si los numeramos tenemos  $\Omega = \{\omega_i\}_{i=1}^N$ , donde  $N$  es el número total de personas de la Comunidad. Elección al azar supone que cada individuo tiene la misma probabilidad de ser elegido. Por tanto tenemos

$$P(\{\omega_i\}) = \frac{1}{N}.$$

El resultado mismo no suele tener un interés especial para nosotros. Seleccionamos aleatoriamente estas personas porque tenemos interés en sus ingresos, en el número de personas que conviven con el o ella, si está casada o es soltera, su glucosa o tensión arterial. Son ejemplos de valores que nos pueden interesar del individuo seleccionado dependiendo de que estemos realizando un estudio de tipo socio económico o bien un estudio de salud. Por ello nuestro interés no está en el individuo  $\omega$  que obtenemos cuando seleccionamos a una persona al azar sino que nuestro interés es un valor asociado a  $\omega$  que denotamos por  $X(\omega)$  y que puede ser su tensión arterial. Lo que es aleatorio es  $\omega$  porque lo seleccionamos aleatoriamente. Una vez tenemos  $\omega$  el valor  $X(\omega) = x$  viene dado. Elegimos al azar a una persona, una vez elegida su edad no es aleatoria, es la que tiene. Elegimos al azar una muestra de agua en una planta de tratamiento de aguas residuales. Luego determinamos la demanda química de oxígeno. A la función que asocia a un resultado un valor numérico se le llama **variable aleatoria**. Lo podemos denotar como

$$\begin{aligned} X : \Omega &\rightarrow \mathbb{R} \\ \omega &\rightarrow X(\omega) = x. \end{aligned}$$

### 21.3.1 Variable aleatoria discreta

Es una variable aleatoria que toma un número finito de valores o bien toma un número infinito de posibles valores que podemos numerar.  
<sup>3</sup>

Un ejemplo muy simple y habitual es la edad de un individuo seleccionado al azar. La edad suele cuantificarse con valores enteros. Decimos que una persona tiene 24 años o que tiene 25 años. Por lo tanto es un valor entero. Si cuantificamos la edad en años entonces los valores posibles de la variable aleatoria son 0, 1, 2, ... . En cualquier caso un número finito de posibles valores.

Otro ejemplo puede ser el número de organismos que observamos en una placa al microscopio, el número de huevos en una buitrera, el número de píxeles defectuosos en un monitor de ordenador, el número de árboles dentro de un quadrat en un muestreo espacial. Todos son ejemplos de variable aleatoria discreta. Como vemos habitualmente son valores que resultan de contar. Es lo más frecuente pero no siempre es así.

---

<sup>3</sup>Para ser rigurosos una variable discreta puede tomar también un número infinito numerable de valores. En fin, detalles técnicos a los que no hay que prestar demasiado interés en un curso como este.

Supongamos que denotamos por  $\mathbb{D} = \{x_1, x_2, \dots\}$  el conjunto de valores posibles para la variable. Las probabilidades que hemos de conocer son

$$P(X = x_i),$$

es decir, la probabilidad de que la variable tome cada uno de sus valores posibles. Estas probabilidades  $P(X = x_i)$  con  $i = 1, \dots, n$  reciben el nombre de **función de probabilidad** de la variable aleatoria  $X$ . Cualquier otra probabilidad que nos interese la podemos obtener a partir de la función de probabilidad.

Por ejemplo, supongamos  $\mathbb{D} = \{0, 1, \dots, 10\}$  y la función de probabilidad aparece en la tabla 21.1.

x	0.000	1.000	2.000	3.000	4.000	5.000	6.000	7.000	8.000	9.000	10.000
$P(X = x)$	0.107	0.268	0.302	0.201	0.088	0.026	0.006	0.001	0.000	0.000	0.000

Tabla 21.1: Función de probabilidad de una variable discreta. En la primera fila el valor y en la segunda la probabilidad de tomar este valor.

A partir de la función de probabilidad  $P(X = x_i)$  podemos calcular cualquier otra probabilidad. Por ejemplo:

$$P(X \leq 2) = P(X = 0) + P(X = 1) + P(X = 2),$$

o bien,

$$P(X \geq 7) = P(X = 7) + P(X = 8) + P(X = 9) + P(X = 10).$$

También

$$P(4 \leq X \leq 7) = P(X = 4) + P(X = 5) + P(X = 6) + P(X = 7).$$

$$P(4 < X \leq 7) = P(X = 5) + P(X = 6) + P(X = 7).$$

$$P(4 < X < 7) = P(X = 5) + P(X = 6).$$

De un modo genérico podemos escribir que

$$P(X \in A) = \sum_{x \in A} P(X = x),$$

siendo  $A$  cualquier conjunto (por ejemplo, un intervalo).

### 21.3.2 Ejercicios

#### Ejercicio 39

Consideremos el experimento aleatorio consistente en lanzar dos veces un dado. Un resultado del experimento puede ser  $\omega = (1, 3)$  indicando que en primer lugar hemos obtenido un 1 y en el segundo lanzamiento hemos obtenido un 3. Consideraremos la variable aleatoria que asocia al resultado obtenido la suma de los valores que obtenemos en el primer y en el segundo lanzamiento. Si  $\omega = (i, j)$  entonces  $X(\omega) = i + j$ .

1. Indicar qué valores puede tomar la variable  $X$ .
2. Obtener la función de probabilidad de la variable  $X$ .
3. Obtener las probabilidades siguientes:  $P(X \leq 1)$ ,  $P(X \leq 2)$ ,  $P(X > 2)$ ,  $P(X \leq 4)$ ,  $P(4 \leq X \leq 6)$ ,  $P(4 < X \leq 6)$ ,  $P(4 \leq X < 6)$ .

### 21.3.3 Variable aleatoria continua

Consideremos el siguiente experimento. Cogemos una muestra de panga vietnamita y medimos la concentración de mercurio en dicha muestra. El resultado  $\omega$  es la muestra que hemos tomado. De esta muestra nos interesa solamente la concentración de mercurio. Nos interesa el valor asociado a la muestra y no la muestra misma. Por ello podemos definir  $X(\omega) = x$  donde  $x$  es la concentración medida de mercurio. El valor aleatorio que observamos los denotamos por  $X$ , la variable aleatoria. Una vez hemos observado el valor, esto es, una vez se ha realizado la determinación de mercurio el valor ya no es aleatorio. Es un valor dado, por ejemplo, una concentración de 0.5 miligramos por kilogramo. Este valor ya no lo denotamos con la letra mayúscula sino con la letra en minúscula,  $x$ . ¿Qué nos interesa conocer sobre estos valores aleatorios? Posiblemente muchas cosas pero lo más básico (y de lo cual se deduce cualquier otra) es conocer la probabilidad que tenemos de que el valor que observemos esté entre dos números. ¿Qué probabilidad tenemos de que la muestra que analicemos de un valor entre 0.3 y 0.6? ¿Qué probabilidad hay de observar un valor entre 0.4 y 0.8? O bien, si una cierta normativa afirma que un valor por encima de 0.5 no se permite entonces parece natural plantearse: ¿cuál es la probabilidad de observar una muestra por encima de este valor? O por el contrario: ¿con qué frecuencia observamos muestras que no alcance el valor 0.5? Las probabilidades que acabamos de indicar se denotan como:  $P(0.3 \leq X \leq 0.6)$ ,  $P(0.4 \leq X \leq 0.8)$ ,  $P(X \geq 0.5)$  y  $P(X \leq 0.5)$ . Sin tener que referirnos a valores concretos podemos denotar de un modo genérico todos los casos anteriores como  $P(a \leq X \leq b)$  donde  $a$  y  $b$  toman los valores que queramos. Cuando consideramos  $P(0.3 \leq X \leq 0.6)$  estamos tomando  $a = 0.3$  y  $b = 0.6$ . También  $P(X \geq 0.5)$  tiene esta forma ya que estamos tomando  $a = 0.5$  y  $b = +\infty$ . Obviamente,  $P(X \leq 0.5)$  corresponde con  $a = -\infty$  y  $b = 0.5$ .

En resumen, cuando trabajamos con una variable aleatoria lo fundamental es conocer las probabilidades  $P(a \leq X \leq b)$  donde  $a$  y  $b$  son números reales o  $a = -\infty$  o  $b = +\infty$ .

*Cuando una variable es continua* entonces la probabilidad anterior se puede calcular como

$$P(a < X \leq b) = \int_a^b f(x)dx.$$

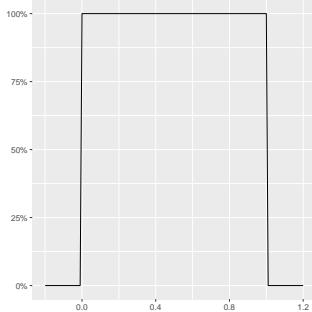
La función  $f$  recibe el nombre de **función de densidad** (de probabilidad) de la variable  $X$ .

De hecho se tiene que

$$P(a < X \leq b) = P(a \leq X < b) = P(a \leq X \leq b) = \int_a^b f(x)dx.$$

**Ejemplo 21.5 (Uniforme en el intervalo unitario)** *Una variable aleatoria uniforme en el intervalo  $[0, 1]$  es un experimento que ya hemos visto. En las calculadoras suele haber una función conocida como rand que cuando la usamos nos devuelve un valor entre 0 y 1. La idea es que es imprevisible el valor y no esperamos que aparezca alrededor de nada. Simplemente dentro del intervalo. La función de densidad viene dada por*

$$f(x) = \begin{cases} 1 & \text{si } 0 \leq x \leq 1 \\ 0 & \text{si } x < 0 \text{ ó } x > 1. \end{cases}$$



La podemos ver representada en la figura 21.3.

Supongamos que tomamos  $0 \leq a \leq b \leq 1$  entonces

$$P(a \leq X \leq b) = \int_a^b f(x)dx = \int_a^b dx = b - a,$$

es decir, la probabilidad de que la variable está en el intervalo  $[a, b]$  depende solamente de la longitud del intervalo y no de dónde está colocado dentro del intervalo  $[0, 1]$ .

¿Podemos generar un valor aleatorio que se comporte como una uniforme en  $[0, 1]$ ? Simplemente con

Figura 21.3: Función de densidad de una variable uniforme en el intervalo  $[0, 1]$ .

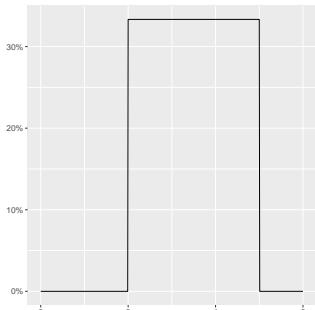
```
runif(1,min=0,max=1)
```

```
## [1] 0.6736497
```

De hecho, podemos simular el número de puntos que queramos. Por ejemplo, generemos 20 valores.<sup>4</sup>

```
runif(20,min=0,max=1)
```

```
## [1] 0.4254709 0.1286097 0.8850760 0.6972555
## [5] 0.3407570 0.2811757 0.7881495 0.3535969
## [9] 0.3210780 0.8187089 0.6592973 0.2631164
## [13] 0.4046266 0.4530579 0.7039735 0.2226044
## [17] 0.6317559 0.4801175 0.4283014 0.0634963
```



**Ejemplo 21.6** También podemos considerar la uniforme en un intervalo arbitrario  $[a, b]$  donde  $a$  y  $b$  son números arbitrarios siendo  $a$  menor que  $b$ . La función de densidad de la variable es

$$f(x) = \begin{cases} \frac{1}{b-a} & \text{si } a \leq x \leq b \\ 0 & \text{en otro caso.} \end{cases}$$

Supongamos  $a = 2$  y  $b = 5$ . La función de densidad la podemos ver en la figura 21.4.

Además, si consideramos un par de puntos  $c$  y  $d$  tales que  $a \leq c \leq d \leq b$  entonces:

$$P(c \leq X \leq d) = \int_c^d f(x)dx = \int_c^d \frac{1}{b-a} dx = \frac{d-c}{b-a}.$$

Otra vez la probabilidad de que el valor aleatorio de  $X$  esté en el intervalo  $[c, d]$  solamente depende de lo largo que es el intervalo y no de dónde está dentro de  $[a, b]$ .

### 21.3.4 Ejercicios

**Ej. 40** — Consideremos una variable aleatoria uniforme en el intervalo  $[0, 1]$ . Se pide:

1. ¿Qué probabilidad tenemos de que la variable sea menor o igual que 0.5? En otras palabras: ¿cuánto vale  $P(X \leq 0.5)$ ?

<sup>4</sup>Más no que ocupa demasiado espacio. En cualquier caso podemos probar a cambiar el valor 20 por el número de valores que nos apetezca simular.

2. ¿Y  $P(X < 0.5)$ ?
3. Calcular  $P(X \geq 0.5)$  y  $P(X > 0.5)$ .
4. Determinar las siguientes probabilidades:  $P(0.6 < X \leq 0.9)$ ,  $P(0.6 \leq X < 0.9)$  y  $P(0.6 \leq X \leq 0.9)$ .

**Ej. 41** — Consideremos una variable aleatoria uniforme en el intervalo  $[0, 8]$ . Se pide:

1. ¿Qué probabilidad tenemos de que la variable sea menor o igual que 0.5? En otras palabras: ¿cuánto vale  $P(X \leq 0.5)$ ?
2. ¿Y  $P(X < 0.5)$ ?
3. Calcular  $P(X \geq 0.5)$  y  $P(X > 0.5)$ .
4. Determinar las siguientes probabilidades:  $P(0.6 < X \leq 0.9)$ ,  $P(0.6 \leq X < 0.9)$  y  $P(0.6 \leq X \leq 0.9)$ .

### 21.3.5 Función de distribución

Dada una variable aleatoria  $X$  se define la función de distribución (o función de distribución acumulada) como la función que para valor real  $x$  nos da la probabilidad de que la variable sea menor o igual que este valor, es decir,

$$F_X(x) = F(x) = P(X \leq x). \quad (21.6)$$

Obviamente si la variable que tenemos es discreta y toma valores en  $\{x_1, x_2, \dots\}$  entonces

$$F_X(x) = \sum_{x_i: x_i \leq x} P(X = x_i),$$

es decir, sumamos la probabilidad de que la variable tome cada uno de los valores que puede tomar y que son menores o iguales al valor  $x$ .

**Ejemplo 21.7** Consideramos la variable aleatoria discreta tal que su función de probabilidad aparece en la tabla 21.1. Vamos a determinar la función de distribución. Las probabilidades que aparecen en la tabla 21.1 aparecen representadas en la figura ???. La función de distribución la hemos representada en la figura 21.6.

En el caso de una variable que sea continua se tiene la igualdad

$$F_X(x) = \int_{-\infty}^x f(t) dt.$$

En general se tiene la siguiente importante igualdad.

$$P(a < X \leq b) = F(b) - F(a) \text{ con } a \leq b, a, b \in \mathbb{R}. \quad (21.7)$$

La igualdad afirma que la probabilidad de que la variable sea estrictamente mayor que  $a$  y menor o igual que  $b$  lo podemos obtener como la diferencia de la función de distribución en los extremos.

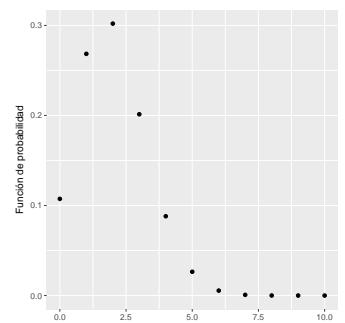


Figura 21.5: Probabilidades de la tabla 21.1.

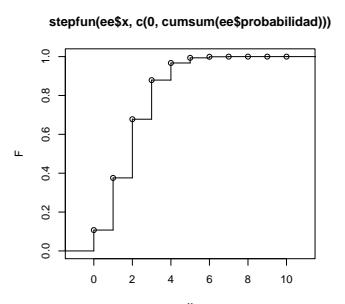


Figura 21.6: Función de distribución de la variable con función de proba-

### 21.3.6 Función de distribución muestral o empírica

Tenemos una muestra de datos numéricos que denotamos por  $\{x_1, \dots, x_n\}$ <sup>5</sup>. ¿Qué es la distribución empírica de estos valores? Supongamos un experimento que consiste en elegir uno de los valores  $x_i$  completamente al azar o, de otro modo, con la misma probabilidad. Sería un experimento que de un modo equiprobable produce cada los resultados  $\{x_1, \dots, x_n\}$ . Esta distribución de probabilidad se conoce como **distribución empírica** y a la correspondiente función de distribución como la **función de distribución empírica o muestral**. Es, por tanto, la distribución de probabilidad que da probabilidad  $\frac{1}{n}$  a cada uno de los datos. De otro modo un experimento que produce el valor  $x_i$  con probabilidad  $1/n$ . Dados los datos  $x_1, \dots, x_n$ , la función de distribución muestral se define como

$$F_n(x) = \frac{|\{x_i : x_i \leq x\}|}{n}, \quad (21.8)$$

donde  $|\cdot|$  denota el cardinal, esto es, el número de elementos del conjunto. Para cada punto  $x$  consideramos el conjunto formado por todos los datos  $x_i$  que son menores o iguales que  $x$ . Contamos el número de puntos en este conjunto (su cardinal) y lo dividimos por el total de datos. En resumen,  $F_n(x)$  nos da la proporción de datos que son menores o iguales que  $x$ . Y esto lo consideramos para todos los valores de  $x$  posibles.

Vamos a considerar dos funciones para obtener la función de distribución muestral. La primera es la función *ecdf* y podemos verla en la figura 21.7.

```
plot(ecdf(y))
```

Si queremos conocer el valor de  $F_n$  en un valor determinado, por ejemplo para  $x = 37$  podemos hacer

```
ecdf(y)(37)
```

```
## [1] 1
```

o bien en 40,

```
ecdf(y)(40)
```

```
## [1] 1
```

La segunda opción es la función *Ecdf* del paquete [71, Hmisc]. Aparece en la figura 21.8.

```
library(Hmisc)
Ecdf(y)
```

### 21.3.7 Ejercicios

**Ej. 42** — Consideremos el experimento aleatorio consistente en lanzar dos veces un dado. Un resultado del experimento puede ser  $\omega = (1, 3)$  indicando que en primer lugar hemos obtenido un 1 y en el segundo lanzamiento hemos obtenido un 3. Consideraremos la variable aleatoria

<sup>5</sup>En estas notas podrían ser las expresiones a nivel de sonda de un microarray.

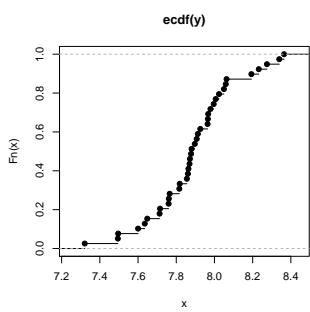


Figura 21.7: Función de distribución muestral con la función *ecdf*.

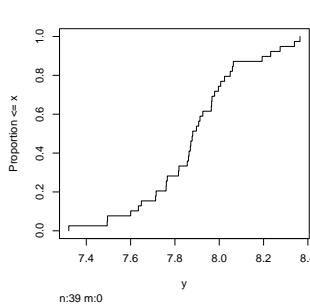


Figura 21.8: Función de distribución muestral con la función *Ecdf*.

que asocia al resultado obtenido la suma de los valores que obtenemos en el primer y en el segundo lanzamiento. Si  $\omega = (i,j)$  entonces  $X(\omega) = i + j$ . Se pide:

- 1.Determinar la función de distribución de la variable aleatoria  $X$ .
- 2.Representar de un modo manual la función de distribución que hemos determinado en el punto 1.
- 3.Representar la función de distribución utilizando la función *step-fun*.

**Ej. 43** — Supongamos una variable uniforme en el intervalo  $[2, 6]$  que denotamos como  $X \sim U(2, 6)$ . Se pide:

- 1.Determinar la función de distribución de la variable aleatoria  $X$ .
- 2.Representar gráficamente la función de distribución de la variable aleatoria  $X$ .

**Ej. 44** — Supongamos una variable uniforme en el intervalo  $[2, 6]$  que denotamos como  $X \sim U(2, 6)$ . Se pide:

- 1.Representar gráficamente la función de distribución utilizando las funciones *plot* y *punif*.

## 21.4 Media y varianza

Una variable suele describirse de un modo simple mediante su media y su varianza. La media nos da una idea de alrededor de qué valor se producen los valores aleatorios de la variable mientras que la varianza cuantifica la dispersión de estos valores alrededor de la media.

### 21.4.1 Media de una variable aleatoria discreta

**Ejemplo 21.8 (La media como límite de medias muestrales)** *Supongamos que tenemos una variable que puede tomar los valores  $\{0, 9\}$  con probabilidades dadas en la tabla 21.2. El experimento supongamos que consiste en elegir al azar una vivienda en una gran población (por ejemplo, Valencia) y observar el número de personas que habitan la vivienda. En la fila etiquetada con  $x$  tenemos el número de personas y en la fila etiquetada  $P(X = x)$  la frecuencia de cada valor posible que asumimos conocidas.*

x	0.00	1.00	2.00	3.00	4.00	5.00	6.00	7.00	8.00	9.00
$P(X = x)$	0.20	0.11	0.13	0.24	0.27	0.02	0.01	0.01	0.00	0.01

Tabla 21.2: Función de probabilidad de la variable aleatoria que nos da el número de personas que residen en una vivienda.

*Podemos simular la selección aleatoria de una vivienda en esa población utilizando la función sample.*

```
x = 0:9
probabilidades = c(0.20,0.11,0.13,0.24,0.27,0.02,0.015,0.009,
0.0009,0.0051)
sample(x,size=1,replace=TRUE,prob=probabilidades)

## [1] 4
```

Supongamos que repetimos el proceso de extracción 100 veces.

```
n = 100
(y = sample(x, size=n, replace=TRUE, prob=probabilidades))

## [1] 2 9 1 4 2 3 4 0 4 1 3 3 0 1 3 4 4 4 9 0 2 4
## [23] 1 3 3 1 4 4 4 2 4 3 6 0 3 4 3 2 1 3 4 3 4 4
## [45] 7 1 0 4 0 3 2 0 3 5 2 3 4 0 2 4 4 3 1 3 0 3
## [67] 4 3 4 3 2 2 3 4 3 4 4 4 1 3 0 2 3 0 3 4 3
## [89] 2 4 0 3 4 4 4 2 4 4 4 0
```

Las frecuencias que observamos de cada tipo son las siguientes

```
prop.table(table(y))

## y
## 0   1   2   3   4   5   6   7   9
## 0.13 0.09 0.13 0.27 0.33 0.01 0.01 0.01 0.02
```

¿Se parecen? En las categorías más probables bastante. Repitamos el proceso con 1000 muestras.

```
y = sample(x, size=1000, replace=TRUE, prob=probabilidades)
prop.table(table(y))

## y
## 0   1   2   3   4   5   6   7
## 0.190 0.100 0.133 0.259 0.265 0.023 0.013 0.008
## 8   9
## 0.002 0.007
```

Vemos cómo se parecen más las frecuencias observadas a las probabilidades de cada uno de los valores. Denotemos los valores simulados por  $\{y_1, \dots, y_n\}$ . Su media muestral será  $\bar{y}_n = \frac{\sum_{i=1}^n y_i}{n}$  pero

$$\bar{y}_n = \sum_{i=1}^n \frac{y_i}{n} = \sum_{x=0}^n x \frac{n_x}{n}$$

donde  $n_x$  denota el número de veces que aparece el resultado  $x$ . Obviamente cuando el número de datos que generamos va creciendo la frecuencia relativa de veces que aparece el resultado  $x$  que viene dada por el cociente  $n_x/n$ , se va aproximando a la probabilidad  $P(X = x)$ . Por ello tendremos que

$$\bar{y}_n = \sum_{x=0}^n x \frac{n_x}{n} \rightarrow \sum_{x=0}^n x P(X = x).$$

En nuestro caso el número medio de personas que viven en la vivienda vendría dada por

```
(mu = sum(x * probabilidades))

## [1] 2.4761
```

En la figura 21.9 hemos representado con una línea horizontal cuya ordenada es la media que acabamos que calcular. Luego vamos simulando valores según la variable discreta que acabamos de proponer. En

el eje de abscisas consideramos el número de valores que vamos promediando. Vemos cómo la media muestral se va aproximando al valor dado de la media.

**Definición 21.2** Si  $X$  es una variable aleatoria discreta que toma los valores  $\{x_1, x_2, \dots\}$  entonces la media de  $X$  se define como

$$EX = \mu_X = \mu = \sum_{i=1}^{+\infty} x_i P(X = x_i).$$

**Ejemplo 21.9** Lanzamos la moneda si sale cara la variable  $X$  vale uno y si sale cruz la variable  $X$  vale cero. ¿Cuál es su media?

$$\mu = 1 \times p + 0 \times (1 - p) = p.$$

### 21.4.2 Varianza y desviación típica

**Definición 21.3 (Varianza y desviación típica)** Si  $X$  es una variable aleatoria discreta que toma los valores  $\{x_1, x_2, \dots\}$  entonces la varianza de  $X$  se define como

$$var(X) = \sigma^2 = E(X - \mu)^2 = \sum_{i=1}^{+\infty} (x_i - \mu)^2 P(X = x_i).$$

Habitualmente además de la varianza se suele utilizar para medir variabilidad la desviación típica dada por

$$\sigma = \sqrt{var(X)} = \sqrt{\sum_{i=1}^{+\infty} (x_i - \mu)^2 P(X = x_i)}.$$

En variables continuas las definiciones de media y varianza son las análogas sustituyendo sumatorios por integrales.

**Definición 21.4** Si  $X$  es una variable continua con función de densidad  $f$  entonces la media de  $X$  se define como

$$EX = \mu = \int_{-\infty}^{+\infty} xf(x) dx$$

**Definición 21.5** Si  $X$  es una variable continua con función de densidad  $f$  entonces la varianza de  $X$  se define como

$$var(X) = \sigma^2 = \int_{-\infty}^{+\infty} (x - \mu)^2 f(x) dx,$$

mientras que la desviación típica de  $X$  se define como

$$\sigma_X = \sigma = \sqrt{\int_{-\infty}^{+\infty} (x - \mu)^2 f(x) dx}.$$

Hemos visto antes que si vamos observando valores de una variable entonces las sucesivas medias muestrales se van aproximando a la media poblacional. ¿Ocurre algo similar con la varianza y con la desviación típica? Por supuesto que sí.

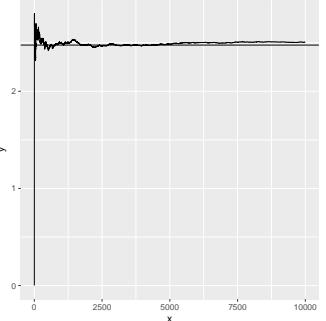


Figura 21.9: Medias muestrales del número de personas que habitan una vivienda en función del tamaño de la muestra. La línea horizontal punteada indica la media poblacional.

## 21.5 Variable Bernoulli

Empezamos con algún ejemplo que motiva esta distribución.

1. Lanzamos una moneda y nos planteamos si sale cara (y lo llamamos éxito) o si no sale cara (y lo llamamos fracaso). Nos fijamos en la ocurrencia o no de un suceso: salir cara. Tenemos un espacio muestral  $\Omega = \{\text{cara}, \text{cruz}\}$  y nos planteamos si se produce  $A = \{\text{cara}\}$ .
2. Elegimos al azar a una persona en la población de la Comunidad Valenciana y nos fijamos si es diabético (y lo consideramos éxito) o no lo es (fracaso). En este caso el espacio muestral es el conjunto de individuos de la Comunidad y el suceso que nos interesa sería el conjunto de individuos que son diabéticos en dicha Comunidad.

Tenemos, en los ejemplos anteriores, distintos experimentos donde nos fijamos en un suceso  $A$  y nos planteamos si ocurre o no ocurre este suceso. Una situación de este tipo recibe el nombre de *prueba de Bernoulli*. Dada una prueba de Bernoulli consideramos la variable aleatoria que nos indica si  $A$  se ha producido, esto es, la variable

$$X(\omega) = 1_A(\omega),$$

donde  $1_A(\omega) = 1$  si  $\omega \in A$  y cero en otro caso. Del experimento que realizamos solamente nos importa saber si un determinado suceso ha tenido lugar. De hecho, como éxito es que el resultado  $\omega$  pertenezca al suceso  $A$  entonces la probabilidad de éxito será  $p = P(A)$ .

El rango de la variable es  $\{1, 0\}$  donde 1 indica el éxito y 0 el fracaso. Una variable de este tipo recibe el nombre de variable Bernoulli. Es el ejemplo más simple de variable aleatoria. Su función de probabilidad sería:  $f_X(1) = P(X = 1) = p$  y  $f_X(0) = P(X = 0) = 1 - p$  que, de un modo conjunto, podemos representar como

$$P(X = x) = p^x(1 - p)^{1-x} \text{ para } x = 0, 1. \quad (21.9)$$

Si una variable sigue una distribución de probabilidad se dice que es una variable Bernoulli o que se distribuye Bernoulli y se denota

$$X \sim Bi(1, p).$$

**Ejemplo 21.10 (Lanzamiento de moneda correcta)** La función de probabilidad es  $f_X(1) = P(X = 1) = 1/2$  y  $f_X(0) = P(X = 0) = 1/2$ .

**Ejemplo 21.11 (Selección aleatoria de un individuo)** Seleccionamos a un individuo al azar en la población de la Comunidad Valenciana. La función de probabilidad es  $f_X(1) = P(X = 1) = p$  y  $f_X(0) = P(X = 0) = 1 - p$ . El valor de  $p$  nos indica la proporción real de diabéticos en la población. Habitualmente el valor de  $p$  no es conocido.

## 21.6 Variable binomial

Consideraremos un experimento con dos posibles resultados. A uno de ellos le llamamos éxito y al otro le llamamos fracaso. Suponemos que hay una probabilidad  $p$  de obtener un éxito (donde  $0 \leq p \leq 1$ ) y, por lo tanto, una probabilidad  $1 - p$  de obtener un fracaso. Un experimento de este tipo recibe el nombre de *prueba de Bernoulli*.

El ejemplo más simple, lanzamiento de una moneda donde identificamos salir cara con éxito y salir cruz como fracaso. En este caso, la probabilidad de éxito es  $p = 0.5$ .

Otro ejemplo, elegimos al azar a una persona de la población española e identificamos éxito con que la persona tenga un cierto atributo (sea diabético por ejemplo) y fracaso con que no tenga el atributo. La probabilidad de éxito  $p$  coincide con la proporción que realmente hay de diabéticos en la población.

Repetimos  $n$  veces una prueba de Bernoulli independientemente una de otra (lanzamos  $n$  veces una moneda) y observamos la variable aleatoria que nos da el número total de éxitos. Una variable de este tipo recibe el nombre de *variable binomial con  $n$  pruebas y una probabilidad de éxito  $p$*  y se suele denotar como

$$X \sim Bi(n, p).$$

Observemos que los valores que puede tomar esta variable son  $0, 1, 2, \dots, n$ , esto es, desde cero éxitos hasta  $n$  éxitos. ¿Qué probabilidad tenemos de observar un número determinado de éxitos? Esta probabilidad,  $P(X = x)$  es la función de probabilidad de la binomial y se prueba que tiene la siguiente expresión.

$$P(X = x) = \binom{n}{x} p^x (1 - p)^{n-x}. \quad (21.10)$$

Si tenemos una variable binomial  $X$  con  $n$  pruebas y una probabilidad de éxito  $p$ ,  $X \sim Bi(n, p)$ , entonces su media es

$$EX = \mu = \sum_{x=0}^n x \binom{n}{x} p^x (1 - p)^{n-x} = np, \quad (21.11)$$

mientras que su varianza viene dada por

$$var(X) = \sigma^2 = \sum_{x=0}^n (x - np)^2 \binom{n}{x} p^x (1 - p)^{n-x} = np(1 - p). \quad (21.12)$$

**Ejemplo 21.12 (Función de probabilidad de la binomial)** *La función dbinom nos permite calcular la función de probabilidad de una variable binomial. Por ejemplo, ¿qué probabilidad tenemos de obtener 70 caras si lanzamos 123 veces una moneda. La respuesta es*

```
dbinom(70, size=123, prob=0.5)
```

```
## [1] 0.02230619
```

**Ejemplo 21.13 (Simulación de una variable binomial)** *Supongamos que queremos lanzar una moneda 30 veces con R.<sup>6</sup> Esto lo podemos hacer con*

```
rbinom(30, size=1, prob=.5)
## [1] 1 0 1 1 0 0 1 1 0 1 1 0 1 1 0 1 0 1 1 1 1 0 1
## [24] 1 1 1 0 1 0 0
```

---

<sup>6</sup> Algo no demasiado raro aunque pueda parecerlo.

*El uno corresponde con éxito (cara) y el cero con fracaso. Si repetimos el proceso posiblemente no obtengamos los mismos resultados.*

```
rbinom(30, size=1, prob=.5)

## [1] 0 0 1 0 1 0 0 1 0 1 1 1 0 0 1 0 0 1 0 0 1 0 1
## [24] 1 0 1 1 1 1 0
```

*Realmente nos interesa el número de éxitos y no el orden en que se producen. Esto lo podemos hacer con*

```
rbinom(1, size=30, prob=.5)

## [1] 16
```

*Si lo repetimos posiblemente no obtengamos el mismo número de unos.*

```
rbinom(1, size=30, prob=.5)

## [1] 16
```

*Supongamos que queremos simular 40 veces el experimento consistente en lanzar 30 veces una moneda y contamos en cada caso el número de unos.*

```
rbinom(40, size=30, prob=.5)

## [1] 19 14 14 14 16 16 14 17 12 13 14 20 19 16 14
## [16] 17 11 22 13 13 15 15 16 16 17 13 20 18 11 13
## [31] 12 8 12 18 17 17 12 13 15 18
```

*Si la moneda no está bien construida y pretendemos que la cara tenga una probabilidad de 0.6, entonces repetimos el experimento anterior con*

```
rbinom(40, size=30, prob=.6)

## [1] 18 20 17 21 20 20 24 20 19 17 22 22 16 17 15
## [16] 14 20 20 19 19 17 18 20 21 18 18 17 13 19 14
## [31] 19 17 17 22 20 15 25 18 18 17
```

#### Ejemplo 21.14 (De cómo calcular probabilidades de la distribución binomial)

*Supongamos que queremos calcular la probabilidad de obtener 23 éxitos cuando realizamos 30 pruebas de Bernoulli donde la probabilidad de éxito es 0.6, es decir, pretendemos calcular para  $X \sim Bi(30, 0.6)$  la función de probabilidad en  $x = 23$  dada por*

$$P(X = x) = \binom{30}{23} (0.6)^{30} (1 - 0.6)^{30-23}. \quad (21.13)$$

```
dbinom(23, size=30, prob=.6)
## [1] 0.02634109
```

Podemos conocer las probabilidades de cada uno de los posibles resultados, es decir, la función de probabilidad  $P(X = x)$ , con

```
dbinom(0:30, size=30, prob=.6)
```

```
##  [1] 1.152922e-12 5.188147e-11 1.128422e-09
##  [4] 1.579791e-08 1.599538e-07 1.247640e-06
##  [7] 7.797748e-06 4.010270e-05 1.729429e-04
## [10] 6.341240e-04 1.997491e-03 5.447702e-03
## [13] 1.293829e-02 2.687184e-02 4.894513e-02
## [16] 7.831221e-02 1.101265e-01 1.360387e-01
## [19] 1.473752e-01 1.396186e-01 1.151854e-01
## [22] 8.227527e-02 5.048710e-02 2.634109e-02
## [25] 1.152423e-02 4.148722e-03 1.196747e-03
## [28] 2.659437e-04 4.274096e-05 4.421478e-06
## [31] 2.210739e-07
```

En la figura 21.10 tenemos la representación gráfica de estas probabilidades.

También podemos obtener la función de la distribución binomial en cualquier punto, es decir, la probabilidad  $P(X \leq 12)$  es

```
pbinom(12, size=30, prob=.6)
## [1] 0.02123988
```

La figura 21.12 muestra la función de distribución de una variable aleatoria con distribución binomial con  $n = 30$  pruebas y una probabilidad de éxito  $p = 0.6$ .

## 21.6.1 Ejercicios

**Ej. 45** — Se pide:

- 1.Simular 100 valores con distribución binomial con 20 pruebas y una probabilidad de éxito en cada prueba de 0.3. Guardar estos valores en el vector  $x$ .
- 2.Calcular la media y varianza muestrales de los valores generados.
- 3.Comparar la media muestral observada con  $20 \times 0.3$  y la varianza muestral observada con  $20 \times 0.3 \times 0.7$  que corresponden con la media y la varianza teóricas.
- 4.Repetir los apartados anteriores sustituyendo las 100 simulaciones por 1000, por 10000 y por 100000. Comparar en cada caso los valores teóricos con los valores muestrales.

**Ej. 46** — Generamos 10000 valores con distribución binomial con 20 pruebas y una probabilidad de éxito por prueba de 0.3. Comparar la función de distribución muestral (vista en la práctica anterior) con la función de distribución teórica que acabamos de ver.

**Ej. 47** — Consideremos una variable aleatoria con distribución binomial con 45 pruebas y una probabilidad de éxito de 0.67. Se pide:

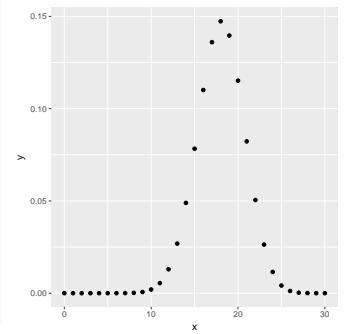


Figura 21.10: Para una variable binomial con  $n = 30$  y una probabilidad de éxito de  $p = 0.6$  mostramos la función de probabilidad que para cada  $x$  nos da la probabilidad de que la

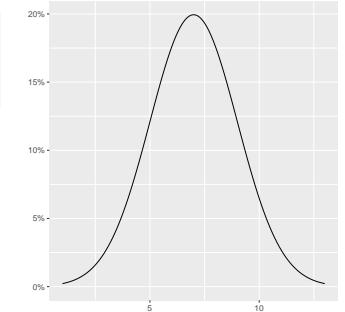


Figura 21.11: Función de densidad de una normal con media 7 y varianza 4.

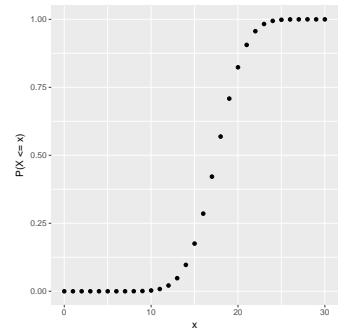


Figura 21.12: Función de distribución de una binomial con  $n = 30$  pruebas y una probabilidad de éxito  $p = 0.6$ . Para cada abcisa  $x$  tenemos la probabilidad de que la variable sea menor o igual que ese valor  $x$ ,  $P(X \leq x)$ .

1. $P(X \leq 23)$ .
2. $P(X < 23)$ .
3. $P(X > 29)$ .
4. $P(X \geq 29)$ .
5. $P(34 < X \leq 45)$ .
6. $P(34 \leq X \leq 45)$ .

**Ej. 48** — Dos especialistas en plagas vegetales difieren en su apreciación de la proporción de palmeras afectadas por el picudo en la Comunidad Valenciana. Uno de ellos (especialista A) afirma que un 30 % de las palmeras están afectadas. El otro, especialista B, afirma que es un 45 % de las palmeras. Un tercer especialista decide tomar una muestra aleatoria simple de la población total de palmeras. En total muestrea un total de 325 palmeras y observa 133 palmeras afectadas. Se pide:

- 1.Calcula la probabilidad de las afirmaciones de cada uno de los especialistas.
- 2.¿Qué afirmación es más probable? ¿Con cuál de los dos juicios nos quedaríamos?
- 3.Si es cierta la afirmación del especialista A: ¿qué probabilidad tenemos de observar 133 o menos?
- 4.Si es cierta la afirmación del especialista B: ¿qué probabilidad tenemos de observar 133 o menos?
- 5.Se decide continuar el muestreo y observamos el estado de 145 palmeras más de las cuales están afectadas 56. *Utilizando solamente la nueva muestra* responde a las preguntas 1, 2, 3 y 4.
- 6.Responder las preguntas 1, 2, 3 y 4 utilizando *conjuntamente* toda la muestra.

## 21.7 Distribución Poisson

Esta distribución aparece ligada a experimentos en los que nos interesa la ocurrencia de un determinado suceso a lo largo de un intervalo finito de tiempo<sup>7</sup>, verificándose las siguientes condiciones:

1. la probabilidad de que el suceso ocurra en un intervalo pequeño de tiempo es proporcional a la longitud del intervalo, siendo  $\lambda$  el factor de proporcionalidad,
2. la probabilidad de que el suceso ocurra en dos o más ocasiones en un intervalo pequeño de tiempo es prácticamente nula.

Fenómenos como el número de partículas que llegan a un contador Geiger procedentes de una fuente radiactiva, el número de llamadas que llegan a una centralita telefónica durante un intervalo de tiempo, las bombas caídas sobre la región de Londres durante la Segunda Guerra mundial y las bacterias que crecen en la superficie de un cultivo, entre otros, pueden ser descritos mediante una variable aleatoria Poisson.

---

<sup>7</sup>En un planteamiento más general, el intervalo finito de tiempo puede ser sustituido por un subconjunto acotado de  $\mathbb{R}^k$

La distribución de *Poisson* de parámetro  $\lambda$  es una de las distribuciones de probabilidad discretas más conocida. Denotaremos que la variable  $X$  sigue una distribución de Poisson con

$$X \sim Po(\lambda).$$

El soporte de la distribución es  $D_X = \{0, 1, 2, 3, \dots\}$  y su función de probabilidad viene dada por

$$f_X(x) = P(X = x) = \frac{e^{-\lambda} \lambda^x}{x!}, \text{ si } x = 0, 1, \dots \quad (21.14)$$

y cero en el resto.<sup>8</sup>

La función de distribución tiene por expresión

$$F_X(x) = \sum_{n \leq x} \frac{e^{-\lambda} \lambda^n}{n!}.$$

Diremos que  $X$  es una *variable Poisson de parámetro  $\lambda$*  o que  $X$  sigue una distribución de Poisson y lo denotaremos

$$X \sim Po(\lambda).$$

¿Por qué es tan importante la distribución de Poisson en las aplicaciones? Posiblemente<sup>9</sup> la razón sea que aproxima a la distribución binomial cuando tenemos un gran número de pruebas y una pequeña probabilidad de éxito en cada una de ellas. Esto es hacemos muchas pruebas pero es muy raro que tengamos un éxito en una de ellas. Por ello, a esta distribución se le suele llamar la *distribución de los sucesos raros*.

**Proposición 21.1 (La distribución de Poisson como límite de distribuciones binomiales)**  
*Sea  $\{X_n\}_{n \geq 1}$  una sucesión de variables tales que  $X_n \sim Bi(n, p_n)$  y  $\lim_{n \rightarrow +\infty} np_n = \lambda$  entonces*

$$\lim_{n \rightarrow +\infty} P(X_n = x) = \frac{e^{-\lambda} \lambda^x}{x!}.$$

**Demostración 21.1** Consideremos la sucesión de variables aleatorias  $X_n \sim Bi(n, p_n)$  en la que a medida que  $n$  aumenta,  $p_n$  disminuye de forma tal que  $np_n \approx \lambda$ . Más concretamente,  $np_n \rightarrow \lambda$ . Tendremos para la función de probabilidad que

$$P(X_n = x) = \binom{n}{x} p_n^x (1 - p_n)^{n-x} = \frac{n!}{x!(n-x)!} p_n^x (1 - p_n)^{n-x},$$

y para  $n$  suficientemente grande,

$$\begin{aligned} P(X_n = x) &\approx \frac{n!}{x!(n-x)!} \left(\frac{\lambda}{n}\right)^x \left(1 - \frac{\lambda}{n}\right)^{n-x} \\ &= \frac{\lambda^x n(n-1)\cdots(n-x+1)}{x! n^x} \left(1 - \frac{\lambda}{n}\right)^n \left(1 - \frac{\lambda}{n}\right)^{-x}. \end{aligned}$$

---

<sup>8</sup>Es fácil comprobar que la función que acabamos de considerar en 21.14 es de probabilidad. Es no negativa y

$$\sum_{x=0}^{+\infty} e^{-\lambda} \frac{\lambda^x}{x!} = e^{-\lambda} \sum_{x=0}^{+\infty} \frac{\lambda^x}{x!} = e^{-\lambda} e^\lambda = 1.$$

<sup>9</sup>Probablemente queda mejor en estas notas

*Al pasar al límite,*

$$\frac{n(n-1)\cdots(n-x+1)}{n^x} \rightarrow 1, \quad \left(1 - \frac{\lambda}{n}\right)^n \rightarrow e^{-\lambda}, \quad \left(1 - \frac{\lambda}{n}\right)^{-x} \rightarrow 1,$$

*y tendremos*

$$\lim_{n \rightarrow +\infty} f_{X_n}(x) = \frac{e^{-\lambda} \lambda^x}{x!}.$$

La utilidad de este resultado reside en permitir la aproximación de la función de cuantía de una binomial con  $n$  pruebas y probabilidad de éxito  $p$ ,  $Bi(n, p)$ , mediante la función de cuantía de una Poisson con parámetro  $\lambda = np$  cuando  $n$  es grande y  $p$  pequeño.

## 21.8 Distribuciones geométrica y binomial negativa

### 21.8.1 Distribución geométrica

¿Cuál es el número de lanzamientos de una moneda (correcta) que hemos de realizar antes de obtener la primera cara? No prefijamos el número de lanzamientos de la moneda. Solamente prefijamos que queremos alcanzar el primer éxito, la primera cara, y *contamos* el número de lanzamientos que hemos realizado.

En general, consideremos una sucesión de pruebas Bernoulli (éxito o fracaso) independientes con la misma probabilidad de éxito,  $p$ . La variable aleatoria que nos interesa es el número de pruebas que hemos de realizar hasta obtener el primer éxito (donde no vamos a contar el éxito). La denotamos por  $X$ . ¿Cuál es su función de probabilidad? Supongamos que denotamos por  $A_i$  el suceso consistente en obtener éxito en la prueba  $i$ -ésima. Es claro que  $\{X = 0\} = A_1$  y

$$P(X = 0) = P(A_1) = p.$$

En general si  $x > 0$  tenemos

$$P(X = x) = P\left(A_1^c \cap \dots \cap A_x^c \cap A_{x+1}\right) = P(A_x) \prod_{i=1}^x P(A_i^c) = p(1-p)^x,$$

donde la segunda igualdad se sigue de la independencia de los sucesos  $A_i$ . En consecuencia, si  $X$  sigue una distribución geométrica con probabilidad de éxito  $p$  entonces su función de probabilidad viene dada por

$$P(X = x) = p(1-p)^x \text{ si } x = 0, 1, \dots,$$

y cero en el resto.

### 21.8.2 Distribución binomial negativa

<sup>256</sup> En esta sección estudiamos la distribución binomial negativa en el modo en que nos resulta útil cuando estudiámos expresión diferencial utilizando conteos.

256

Consideraremos pruebas Bernoulli independientes con la misma probabilidad de éxito,  $p$ , en cada una de ellas. Fijamos el número de **fracasos** que queremos obtener, por ejemplo,  $r$  fracasos. Nos planteamos

**el número de éxitos hasta obtener el r-éximo fracaso.** La función de probabilidad vendrá dada por

$$P(X = k) = \binom{k+r-1}{k} p^k (1-p)^r \quad (21.15)$$

con  $k = 0, 1, \dots$  donde el nombre de binomial negativa es consecuencia de

$$\begin{aligned} \binom{k+r-1}{k} &= \\ \frac{(k+r-1)!}{k!(r-1)!} &= \frac{(k+r-1)\dots r}{k!} = \\ (-1)^k \frac{(-r)(-r-1)\dots(-r-k+1)}{k!} &= (-1)^k \binom{-r}{k}. \end{aligned} \quad (21.16)$$

Se demuestra que

$$E(X) = \mu = \frac{pr}{1-p}, \quad (21.17)$$

y que su varianza viene dada por

$$var(X) = \sigma^2 = \frac{pr}{(1-p)^2}. \quad (21.18)$$

Aunque perdemos la interpretación podemos utilizar un valor de  $r$  que sea un número positivo cualquiera, no necesariamente un número entero. Recordemos que la función gamma de Euler generaliza el concepto de número factorial. La función gamma se define como  $\Gamma(r) = \int_0^{+\infty} x^r e^{-x} dx$ . Y, en particular, se tiene que  $\Gamma(r) = (r-1)\Gamma(r-1)$ . Si  $r$  es un entero positivo tendremos  $\Gamma(r) = (r-1)!$ . Teniendo en cuenta esto es natural la generalización.

$$P(X = k) = \frac{\Gamma(k+r)}{k!\Gamma(r)} p^k (1-p)^r \quad (21.19)$$

con  $k = 0, 1, \dots$

En lo anterior hemos parametrizado la familia utilizando  $k$  y  $r$ . Supongamos que parametrizamos utilizando  $\mu$  y  $r$ . Tendremos entonces  $p = \mu/(r+\mu)$  y sustituyendo

$$P(X = k) = \frac{\Gamma(k+r)}{k!\Gamma(r)} \left(\frac{\mu}{r+\mu}\right)^k \left(\frac{r}{r+\mu}\right)^r. \quad (21.20)$$

Si ahora definimos el parámetro  $\phi = 1/r$  entonces la forma que adopta la función de probabilidad es

$$P(X = k) = \frac{\Gamma(k+1/\phi)}{k!\Gamma(1/\phi)} \left(\frac{\mu}{\mu+1/\phi}\right)^k \left(\frac{1}{1+\mu\phi}\right)^{1/\phi}. \quad (21.21)$$

Además se sigue fácilmente que

$$var(X) = \sigma^2 = \mu + \phi\mu^2. \quad (21.22)$$

Por ello el parámetro  $\phi$ , que recibe el nombre de parámetro de dispersión, cuantifica el incremento de la varianza respecto de la media.

## 21.9 Distribución normal

**Definición 21.6 (Variable normal)** Una variable aleatoria  $X$  se dice que sigue una distribución normal con media  $\mu$  y varianza  $\sigma^2$  (o, simplemente, que es una variable aleatoria normal) y se denota con  $X \sim N(\mu, \sigma^2)$ <sup>10</sup> si su función de densidad viene dada por

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}} \quad (21.23)$$

De otro modo, si tomamos dos valores arbitrarios  $a$  y  $b$  con  $a \leq b$  entonces

$$P(a \leq X \leq b) = \int_a^b f(x)dx.$$

En esta sección asumimos siempre que la variable aleatoria sigue una distribución normal con media  $\mu$  y con varianza  $\sigma^2$ ,  $X \sim N(\mu, \sigma^2)$ .

En la figura 21.11 aparece un ejemplo de la función definida en 21.23. En concreto es una normal con media 7 y varianza 4.

Ya lo indicamos en la propia definición pero se puede demostrar que la media y varianza de una normal son  $\mu$  y  $\sigma^2$ . En definitiva estamos afirmando que se verifican las siguientes ecuaciones.

$$\begin{aligned} \mu &= \int_{-\infty}^{+\infty} xf(x)dx, \\ \sigma^2 &= \int_{-\infty}^{+\infty} (x - \mu)^2 f(x)dx. \end{aligned}$$

**Nota 21.1 (La normal estándar)** Una variable aleatoria  $Z$  se dice que tiene una distribución normal estándar cuando su media es cero y su varianza es uno:  $Z \sim N(0, 1)$ . Su función de densidad es

$$f(x) = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2}. \quad (21.24)$$

La representación gráfica de esta densidad la tenemos en la figura 21.13.

Si  $Z$  es una normal estándar entonces la función de distribución, esto es, la función que para cada valor  $z$  nos da la probabilidad de que la variable sea menor o igual que este valor  $z$  es la siguiente

$$\Phi(z) = \int_{-\infty}^z \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} dx. \quad (21.25)$$

Dado un punto  $z$  el valor de la función  $\Phi(z)$  nos da el área bajo la curva de la densidad normal entre  $-\infty$  y el punto  $z$ . En la figura 21.14 hemos rayado en negro esta zona para  $z = 1.3$

Hay tablas que nos proporcionan el valor de esta función para diferentes valores de  $z$ .<sup>11</sup> Esto era necesario cuando no teníamos herramientas informáticas. Ahora lo lógico es utilizar software. En concreto el valor de  $\Phi(1.3)$  (área de la zona rayada en negro en la figura 21.14) lo obtendríamos con R del siguiente modo.

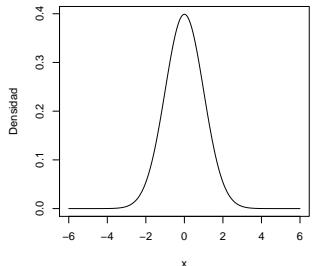


Figura 21.13: Función de densidad de una normal estándar o típica.

<sup>10</sup>También se denota con frecuencia  $X \sim N(\mu, \sigma)$ , es decir, se indica la media  $\mu$  y la desviación típica o estándar  $\sigma$ .

<sup>11</sup>Simplemente poniendo en Google “tablas de la normal” nos aparecen un montón de tablas. Cualquier libro de texto de hace unos años lleva al final del texto unas tablas de la normal.

```
pnorm(1.3)
```

```
## [1] 0.9031995
```

En la figura 21.15 tenemos representada la función  $\Phi$ .

**Nota 21.2 (Estandarización o tipificación)** Si tenemos una variable aleatoria con distribución normal con media  $\mu$  y varianza  $\sigma^2$  entonces la variable aleatoria  $Z = \frac{X-\mu}{\sigma}$  sigue una distribución normal con media 0 y con varianza 1, esto es, se verifica

$$Z = \frac{X - \mu}{\sigma} \sim N(0, 1). \quad (21.26)$$

<sup>12</sup> Esta transformación recibe el nombre de tipificación o estandarización de la variable aleatoria  $X$ .

En la figura 21.16 mostramos la densidad de una variable  $X$  normal con media 7 y varianza 4 y la densidad de la variable tipificada  $Z$ .

**Nota 21.3 (De cómo calculaban los antiguos las probabilidades con la normal)** ¿Qué problema nos planteamos? Suponemos que una cierta cantidad sigue una distribución normal con unos parámetros (media y varianza) dados. Nos planteamos cómo calcular la probabilidad de que la variable esté en un cierto intervalo. Por ejemplo, sabemos que el valor aleatorio que observamos sigue una distribución normal con media 56 y desviación típica 9. ¿Qué probabilidad tenemos de que la variable aleatoria tome un valor entre 60 y 63? Nos planteamos el valor de la siguiente probabilidad:  $P(60 \leq X \leq 63)$ . Esta probabilidad corresponde con la zona rayada de negro en la figura 21.17. Para calcular este área se aplicaban las siguientes igualdades donde  $Z = (X - 56)/9$ ,

$$\begin{aligned} P(60 \leq X \leq 63) &= P\left(\frac{60 - 56}{9} \leq \frac{X - 56}{9} \leq \frac{63 - 56}{9}\right) = \\ P\left(\frac{60 - 56}{9} \leq Z \leq \frac{63 - 56}{9}\right) &= P\left(Z \leq \frac{63 - 56}{9}\right) - P\left(Z \leq \frac{60 - 56}{9}\right). \end{aligned} \quad (21.27)$$

Pero la variable  $Z$  es una normal estándar por lo que

$$P\left(Z \leq \frac{63 - 56}{9}\right) - P\left(Z \leq \frac{60 - 56}{9}\right) = \Phi\left(\frac{63 - 56}{9}\right) - \Phi\left(\frac{60 - 56}{9}\right).$$

De un modo genérico lo que acabamos de indicar es que si  $X \sim N(\mu, \sigma^2)$  entonces

$$P(60 \leq X \leq 63) = \Phi\left(\frac{63 - \mu}{\sigma}\right) - \Phi\left(\frac{60 - \mu}{\sigma}\right),$$

siendo  $\Phi$  la función de distribución de una normal estándar.

Si suponemos que  $X \sim N(\mu, \sigma^2)$  y tomamos dos números  $a$  y  $b$  tales que  $a \leq b$  entonces

$$P(a \leq X \leq b) = \Phi\left(\frac{b - \mu}{\sigma}\right) - \Phi\left(\frac{a - \mu}{\sigma}\right), \quad (21.28)$$

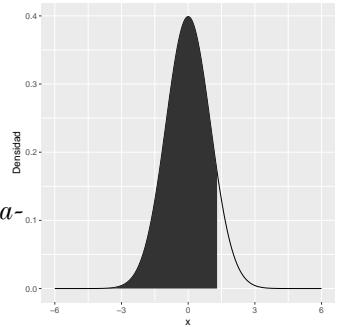


Figura 21.14: La función de distribución de la normal estándar en el punto 1.3 corresponde con el área de la zona rayada.

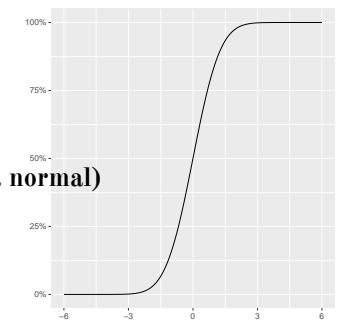


Figura 21.15: La función de distribución de la normal estándar.

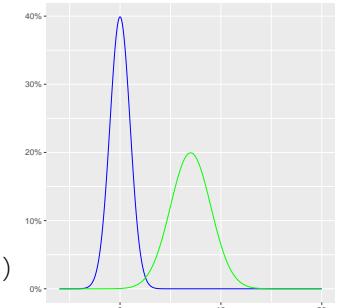
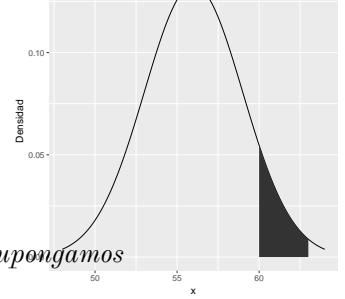


Figura 21.16: Función de densidad de una normal con media 7 y varianza 4 (verde) y de una normal típica con media 0 y varianza 1 (azul).



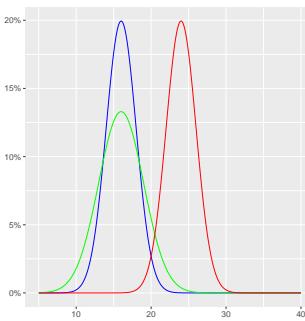
**Ejemplo 21.15 (Calculando la función de densidad de una normal)** Supongamos que  $\mu = 16$  y  $\sigma^2 = 4$ . La función de densidad en un punto la podemos calcular con

```
dnorm(14,mean= 16, sd= 2)
## [1] 0.1209854
```

o, en un conjunto de puntos, con

```
x0 = seq(10,22,1)
dnorm(x0,mean= 16, sd= 2)

## [1] 0.002215924 0.008764150 0.026995483
## [4] 0.064758798 0.120985362 0.176032663
## [7] 0.199471140 0.176032663 0.120985362
## [10] 0.064758798 0.026995483 0.008764150
## [13] 0.002215924
```

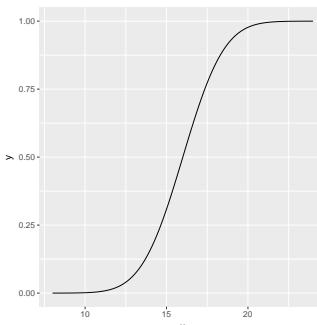


En la figura 21.18 aparecen tres densidades normales con parámetros distintos de modo que veamos el efecto de modificar la media y la varianza. En concreto se representan las densidades de las distribuciones normales  $N(16, 4)$ ,  $N(24, 4)$  y  $N(16, 9)$ .

**Ejemplo 21.16** Podemos generar valores aleatorios con distribución normal.

```
rnorm(20,mean= 16, sd= 2)
## [1] 17.03234 18.49385 15.11046 20.57400 19.35615
## [6] 16.17576 16.69089 16.81915 13.23524 14.93939
## [11] 14.30180 18.22035 17.23786 17.45294 18.62862
## [16] 16.47633 18.45759 18.46499 15.02872 16.07962
```

Figura 21.18: Funciones de densidad de una normal con media 16 y desviación típica 2 (azul), de una normal con media 16 y desviación típica 3 (verde) y una normal con media 24 y desviación típica 2 (rojo).



La función de distribución de la variable  $X$ , es decir,  $F(x) = P(X \leq x)$  la obtenemos con

```
pnorm(14,mean= 16, sd= 2)
## [1] 0.1586553
```

Podemos representar esta función (figura 21.19).

También podemos plantear el problema inverso. Consideramos una probabilidad, por ejemplo 0.34, y buscamos el valor de  $x$  donde  $P(X \leq x) = 0.34$  o dicho de otro modo el percentil de orden 0.34.

```
qnorm(0.34,mean= 16, sd= 2)
## [1] 15.17507
```

<sup>12</sup> Esta afirmación es simplemente consecuencia de la siguiente igualdad (y perdón por la ecuación)

$$\int_a^b \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{1}{2}\frac{(x-\mu)^2}{\sigma^2}} dx = \int_{\frac{a-\mu}{\sigma}}^{\frac{b-\mu}{\sigma}} \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}x^2} dx.$$

Figura 21.17: Densidad de una  $N(56, 9)$ . El área de la zona rayada en negro corresponde a la probabilidad de que la variable esté entre 60 y 63.

**Nota 21.4 (¿Cómo interpretar la desviación típica?)** Hemos visto cómo una desviación típica mayor supone una mayor variabilidad. La variable aleatoria tiende a producir valores más dispersos. Hemos representado la función de densidad de distintas distribuciones normales y vemos cómo cuando la desviación típica es mayor entonces la gráfica es más plana. Los valores normales se observan alrededor de la media  $\mu$  y están más o menos dispersos según el valor de  $\sigma$  sea mayor o menor. Hay una interpretación sencilla de la desviación estándar. Consideremos el intervalo  $[\mu - \sigma, \mu + \sigma]$ , ¿qué probabilidad tenemos de que una variable aleatoria con distribución normal esté en este intervalo?

$$P(\mu - \sigma \leq X \leq \mu + \sigma) = P(-1 \leq \frac{X - \mu}{\sigma} \leq 1) = P(-1 \leq Z \leq 1) \quad (21.29)$$

siendo  $Z$  una variable con distribución normal estándar,  $Z \sim N(0, 1)$ . Pero,

$$\begin{aligned} P(-1 \leq Z \leq 1) &= \int_{-1}^{+1} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx = \\ &\int_{-\infty}^{+1} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx - \int_{-\infty}^{-1} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx = \Phi(1) - \Phi(-1). \end{aligned} \quad (21.30)$$

Vamos a calcular la diferencia anterior utilizando R.

```
pnorm(1,mean=0,sd=1) - pnorm(-1,mean=0,sd=1)
## [1] 0.6826895
```

Por tanto, si  $X$  es una variable aleatoria con distribución normal con media  $\mu$  y desviación típica  $\sigma$  entonces la probabilidad de que la variable esté entre  $\mu - \sigma$  y  $\mu + \sigma$  es

$$P(\mu - \sigma \leq X \leq \mu + \sigma) = 0.6826895. \quad (21.31)$$

De un modo análogo si consideramos el intervalo  $[\mu - 2\sigma, \mu + 2\sigma]$  entonces

$$P(\mu - 2\sigma \leq X \leq \mu + 2\sigma) = P(-2 \leq Z \leq 2) \quad (21.32)$$

que viene dado por

```
pnorm(2,mean=0,sd=1) - pnorm(-2,mean=0,sd=1)
## [1] 0.9544997
```

Y finalmente si consideramos el intervalo  $[\mu - 3\sigma, \mu + 3\sigma]$  se tiene

$$P(\mu - 3\sigma \leq X \leq \mu + 3\sigma) = P(-3 \leq Z \leq 3) \quad (21.33)$$

que es igual a

```
pnorm(3,mean=0,sd=1) - pnorm(-3,mean=0,sd=1)
## [1] 0.9973002
```

En la tabla 21.3 tenemos las probabilidades que hemos calculado. De un modo sencillo podemos decir: la variable dista de la media en una desviación estándar con una probabilidad de 0.68, en dos desviaciones con una probabilidad de 0.95 y en tres desviaciones estándar con una probabilidad de 0.99.

$P(\mu - \sigma \leq X \leq \mu + \sigma)$	0.6826895
$P(\mu - 2\sigma \leq X \leq \mu + 2\sigma)$	0.9544997
$P(\mu - 3\sigma \leq X \leq \mu + 3\sigma)$	0.9973002

Tabla 21.3: Probabilidad de que la variable diste de la media en un número dado de desviaciones típicas

## 21.10 Ejercicios

**Ej. 49** — Se pide:

- 1.Simular 100 valores con distribución normal con media 20 y desviación típica 3. Guardar estos valores en el vector  $x$ .
- 2.Calcular la media y varianza muestrales de los valores generados.
- 3.Comparar la media muestral observada con 20 y la varianza muestral observada con 9 que corresponden con la media y la varianza teóricas.
- 4.Repetir los apartados anteriores sustituyendo las 100 simulaciones por 1000, por 10000 y por 100000. Comparar en cada caso los valores teóricos con los valores muestrales.

**Ej. 50** — Consideremos una variable aleatoria con distribución normal con media 20 y desviación típica 3. Se pide:

1. $P(X \leq 23)$ .
2. $P(X < 23)$ .
3. $P(X > 29)$ .
4. $P(X \geq 29)$ .
5. $P(34 < X \leq 45)$ .
6. $P(34 \leq X \leq 45)$ .

## Capítulo 22

# Distribución muestral

### 22.1 Población y muestra aleatoria

Podemos tener interés en estudiar alguna característica en una población grande. Por ejemplo, la población puede ser una población animal o vegetal: los patos de la Albufera; las palmeras de la provincia de Valencia; una especie de pájaros en la provincia de Alicante; toda la población española, etc.

Fijémonos en las palmeras de la provincia de Valencia. Supongamos que la característica que nos interesa es si la palmera está afectada por el *picudo rojo*. Queremos conocer la proporción  $p$  de palmeras afectadas por la plaga. Para ello lo que podemos hacer es recorrer toda la provincia e ir palmera por palmera observando si el picudo ha afectado a la palmera. Cuando hayamos observado todas las palmeras la proporción de palmeras afectadas será el cociente entre el número de palmeras afectadas y el número total de palmeras que hay. Realmente no es difícil. Sin embargo, parece laborioso, caro y, posiblemente, innecesario. Es más barato (y esperemos que suficiente) elegir al azar un conjunto de  $n$  palmeras (con un número  $n$  no muy grande) y observar su estado. Es decir, tomar una *muestra aleatoria* de palmeras y, con lo que observamos en la muestra, intentar estimar el valor de la proporción en toda la población o *proporción poblacional*.

### 22.2 Distribución muestral de una variable binomial

**Ejemplo 22.1** Tenemos una población de individuos. Intentamos estudiar la prevalencia de una enfermedad no muy frecuente, la hidatidosis. Vamos a suponer que la proporción real de personas con la enfermedad es  $p = 0.034$ .

```
N=2374560  
X = rbinom(N, size=1, prob=.034)
```

Hemos numerado a la población y guardado los datos en el vector  $X$ . Si la  $i$ -ésima persona tiene la enfermedad entonces guardamos en la posición  $i$  del vector  $X$  un valor uno, si no la tiene guardamos un valor cero. Por ejemplo, los 10 primeros individuos de la población son

```
X[1:10]
## [1] 0 0 0 0 0 0 0 0 0 0
```

*Y el individuo que ocupa la posición 100000 es*

```
X[100000]
## [1] 0
```

*Ahora vamos a simular el muestreo aleatorio de la población. Tomamos una muestra de tamaño  $n = 100$  y observamos la proporción de personas enfermas en la muestra.*

```
n = 100
x = sample(X,n)
x

## [1] 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [23] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
## [45] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
## [67] 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0
## [89] 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0
```

*De hecho el total de individuos enfermos en la muestra lo podemos ver con*

```
sum(x)
## [1] 4
```

*Repitiendo el proceso obtenemos*

```
x = sample(X,100)
sum(x)

## [1] 2
```

*Y si lo hacemos como unas 20 veces obtenemos los siguientes valores observados*

```
## [1] 3 5 5 2 3 2 3 7 6 4 4 0 4 4 2 4 6 3 3 2
```

*Si, en lugar de contar el número de enfermos observados, nos fijamos en la proporción observada tenemos*

```
sumas/n

## [1] 0.03 0.05 0.05 0.02 0.03 0.02 0.03 0.07 0.06
## [10] 0.04 0.04 0.00 0.04 0.04 0.02 0.04 0.06 0.03
## [19] 0.03 0.02
```

*¿Con qué frecuencia observamos 3 enfermos?  
La obtenemos con la expresión*

$$\binom{n}{3} \cdot 0.034^3 (1 - 0.034)^{97}$$

*En general si en una selección aleatoria de  $n$  individuos contamos el número de individuos con la enfermedad (que entendemos como el número de éxitos) entonces*

$$P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$$

*siendo  $p$  la proporción en la población de enfermos (que en nuestro ejemplo estamos suponiendo  $p = 0.034$ ).*

### 22.2.1 Ejercicios

**Ej. 51 —**<sup>257</sup> Muchos equipos de investigación pretenden realizar un estudio sobre el porcentaje de personas que tienen cáncer de colon. Si una muestra aleatoria de diez personas se pudo obtener, y si la probabilidad de probabilidad de tener cáncer de colon es 0.05, ¿cuál es la probabilidad de que un equipo de investigación obtenga  $\hat{p} = 0.1$ ? ¿Y la de  $\hat{p} = 0.05$ ?

<sup>257</sup> [151, pág. 79, problemas 2-3]

**Ej. 52 —**<sup>258</sup> Alguien afirma que la probabilidad de perder dinero cuando se utiliza una estrategia de inversión para la compra y venta de los productos básicos es de 0.1. Si esta afirmación es correcta: ¿cuál es la probabilidad de obtener  $\hat{p} \leq 0.05$  sobre la base de una muestra aleatoria de 25 de los inversores?

<sup>258</sup> [151, pág. 80, problema 4]

**Ej. 53 —**<sup>259</sup> Imaginemos que un millar de equipos de investigación extraen una muestra al azar de una distribución binomial con  $p = 0.4$ , cada estudio está basado en una muestra de tamaño 30. Tendremos 1000 valores de  $\hat{p}$ . Si promediamos estos 1000 valores: ¿Cuál sería aproximadamente el resultado? Si calculamos la varianza muestral de los valores de  $\hat{p}$ : ¿Cuál sería aproximadamente el resultado?

<sup>259</sup> [151, pág. 80, problemas 6-7]

## 22.3 Distribución muestral de la media bajo normalidad

Supondremos que es una población normal con media  $\mu = 160$  y una desviación estándar de 10.23. Podría corresponder con la altura de los individuos. Suponemos que hay  $N = 237456$  personas en la población en estudio. Como no vamos a medir a tantas personas ahorraremos tiempo generando aleatoriamente estos valores.

```
N = 237456
X = rnorm(N, mean=160, sd=10.23)
```

Es una población de  $2.37456 \times 10^5$  individuos. Por  $X$  estamos denotando *toda la población*. Es decir, suponemos (de un modo irreal) que tenemos a toda la población. Podemos ver los 10 primeros elementos de la población con

```
X[1:10]
```

```
## [1] 143.7714 148.5352 151.5329 151.3984 175.6894
## [6] 172.3557 164.0881 176.9400 148.9404 142.8552
```

La figura 22.1 tenemos un histograma de toda la población. La figura 22.2 tiene un estimador kernel de la densidad de probabilidad.

Como conocemos toda la población podemos conocer su media o *media poblacional* y vale:

```
(mu=mean(X))
## [1] 160.0128
```

Tomamos una muestra de  $n = 100$  individuos de la población con la función *sample*.

```
n = 100
x = sample(X,n)
```

Parece natural *aproximar* o, dicho con propiedad, *estimar* el valor desconocido de  $\mu$  con la media muestral. Veamos el valor:

```
mean(x)
## [1] 157.9384
```

Si repetimos la selección de los individuos y el cálculo de la media muestral tenemos

```
x = sample(X,n)
mean(x)
## [1] 158.9058
```

Podemos repetirlo muchas veces e iremos obteniendo valores distintos. En la figura ?? estimador kernel de la densidad de los valores generados.

Supongamos que repetimos lo anterior pero incrementando el tamaño de la muestra. En lugar de tomar muestras de tamaño 100 pasamos a tomar muestras de tamaño 400. En la figura ?? mostramos las medias muestrales obtenidas para diferentes muestras. Las medias muestrales están mucho más próximas a la media poblacional.

Finalmente supongamos que tomamos muestras de tamaño creciente y mostramos en abscisas el tamaño de la muestra y en ordenadas la media muestral observada. En la figura 22.3 tenemos el resultado. Lo repetimos. En la figura ?? tenemos las medias observadas. No obtenemos las mismas medias muestrales pero si un comportamiento aleatorio similar.

Si denotamos la muestra aleatoria que estamos extrayendo de la población con  $X_1, \dots, X_n$  entonces la media muestral (que utilizaremos para estimar la media poblacional) tiene una distribución (o se distribuye como) una normal con media  $\mu$  (la media poblacional) y con varianza  $\sigma_{\bar{X}_n}^2 = \frac{\sigma^2}{n}$ , la varianza poblacional dividida por el tamaño de la muestra. De un modo resumido esto se expresa con

$$\bar{X}_n \sim N(\mu, \frac{\sigma^2}{n}) \quad (22.1)$$

si  $X_1, \dots, X_n$  son variables aleatorias independientes y con distribución

$$X_i \sim N(\mu, \sigma^2), \text{ con } i = 1, \dots, n.$$

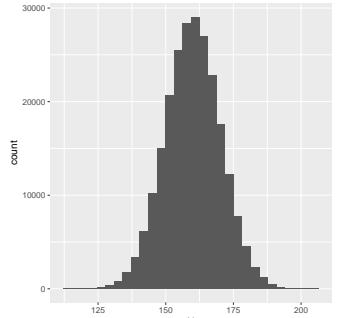


Figura 22.1: Histograma de la población de alturas.

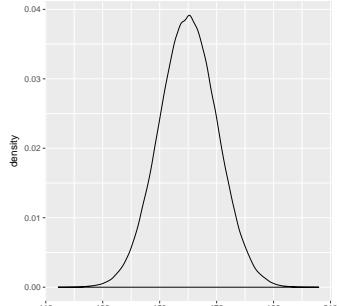


Figura 22.2: Estimador kernel de la densidad de las alturas.

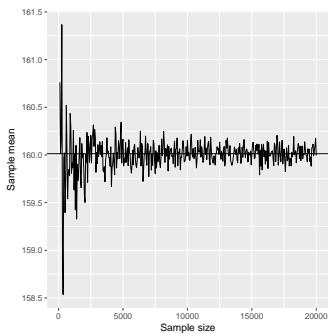


Figura 22.3: Generamos muestras de tamaño creciente. Calculamos la media muestral de cada muestra. En el eje de abscisas mostramos el tamaño de la muestra que hemos generado. En el eje de ordenadas el valor observado de la media muestral. La línea horizontal muestra la media poblacional. Vemos cómo las medias muestrales se aproximan a la media de la población.

El resultado dado en [22.1](#) también lo podemos expresar como

$$\sqrt{n} \frac{\bar{X}_n - \mu}{\sigma} \sim N(0, 1) \quad (22.2)$$

Nos interesa conocer probabilidades como

$$P(\bar{X} \leq b)$$

o en general probabilidades como

$$P(a \leq \bar{X} \leq b)$$

donde  $a$  y  $b$  son valores que nos pueden ir interesando dependiendo del problema.

**Ejemplo 22.2** Por ejemplo, supongamos que nos interesa saber qué probabilidad tiene la media muestral de ser menor que 162. Como estamos suponiendo que conocemos toda la población podemos tomar como varianza la de toda la población.

```
sigma = sd(X)
```

Y la probabilidad la podemos obtener con

```
pnorm(162,mean=mu,sd=sigma/sqrt(n))
## [1] 0.9741615
```

¿Qué probabilidad tenemos de que la media esté entre 159 y 162?

```
pnorm(162,mean=mu,sd=sigma/sqrt(n)) -
  pnorm(159,mean=mu,sd=sigma/sqrt(n))
## [1] 0.8135001
```

¿O de que sea mayor que 160?

```
1 - pnorm(160,mean=mu,sd=sigma/sqrt(n))
## [1] 0.5050096
```

Si

$$\Phi(x) = \int_{-\infty}^x \frac{1}{\sqrt{2\pi}\sigma} \exp -\frac{(t-\mu)^2}{2\sigma^2} dt.$$

lo que estamos haciendo con R es simplemente aplicar que

$$P(\bar{X} \leq b) = \Phi(b)$$

o

$$P(a \leq \bar{X} \leq b) = \Phi(b) - \Phi(a)$$

o

$$P(a \leq \bar{X}) = 1 - \Phi(a)$$

### 22.3.1 Ejercicios

**Ej. 54 —**<sup>260</sup> Supongamos  $n = 16$ ,  $\sigma = 2$  y  $\mu = 30$ . Supongamos <sup>260</sup> [151, pág. 84, problema 8] normalidad. Determinar:

1. $P(\bar{X} \leq 29),$
2. $P(\bar{X} > 30.5),$
3. $P(29 \leq \bar{X} \leq 31).$

<sup>261</sup> [151, pág. 84, problemas 10-11]

**Ej. 55 —**<sup>261</sup> Alguien dice que dentro de un determinado barrio, el costo medio de una casa es de  $\mu = 100000$  euros con una desviación estándar de  $\sigma = 10.000$  euros. Supongamos que, basándonos en  $n = 16$  viviendas, observamos una media muestral  $\bar{X} = 95.000$ . Suponiendo normalidad, ¿cuál es la probabilidad de obtener una media muestral como la observada o menor si las afirmaciones sobre la media y desviación estándar son verdaderas? ¿Y la probabilidad de tener una media muestral entre 97500 y 102500 euros?

<sup>262</sup> [151, pág. 85, problema 13]

**Ej. 56 —**<sup>262</sup> Supongamos que eres un profesional de la salud interesado en los efectos de la medicación en la presión arterial diastólica de las mujeres adultas. Para un medicamento en particular que está siendo estudiado, se encuentra que para  $n = 9$  mujeres, la media muestral es  $\bar{X} = 85$  y la varianza muestral es  $s^2 = 160.78$ . Estimar el error estándar de la media muestral asumiendo que tenemos una muestra aleatoria.

<sup>263</sup> [151, pág. 85, problema 12]

**Ej. 57 —**<sup>263</sup> Una compañía afirma que las primas pagadas por sus clientes para el seguro de automóviles tiene una distribución normal con media  $\mu = 750$  euros y desviación estándar  $\sigma = 100$  euros. Suponiendo normalidad, ¿cuál es la probabilidad de que para  $n = 9$  clientes elegidos al azar, la media muestral tome un valor entre 700 y 800 euros?

## 22.4 Distribución muestral de la media en poblaciones no normales. Teorema central del límite

El resultado que damos en 22.1 es aproximadamente cierto incluso aunque los datos no sigan aproximadamente una distribución normal.

### 22.4.1 Aproximación de la distribución binomial

Con datos binomiales, el estimador de la proporción  $p$ ,  $\hat{p}$  podemos verlo como una media muestral

$$\hat{p} = \frac{\sum_{i=1}^n X_i}{n},$$

donde  $X_i = 1$  si hay un éxito en la i-ésima prueba de Bernoulli. Utilizando el teorema central del límite tenemos que

$$Z = \frac{\hat{p} - p}{\sqrt{p(1-p)/n}}. \quad (22.3)$$

## 22.4. DISTRIBUCIÓN MUESTRAL DE LA MEDIA EN POBLACIONES NO NORMALES. TEOREMA

Si  $n$  es suficientemente grande tenemos que para cualesquiera valores reales  $a$  y  $b$  tenemos

$$\begin{aligned} P(a \leq \hat{p} \leq b) &= \\ P(\hat{p} \leq b) - P(\hat{p} \leq a) &= \\ P(Z \leq \frac{b-p}{\sqrt{p(1-p)/n}}) - P(Z \leq \frac{a-p}{\sqrt{p(1-p)/n}}). & \quad (22.4) \end{aligned}$$

La calidad de la aproximación depende los valores de  $n$  y de  $p$ . Una regla simple es que la aproximación es buena si  $np \geq 15$  y  $n(1-p) \geq 15$ .

### 22.4.2 Ilustración del teorema central del límite

Hemos utilizado en la sección anterior este resultado probabilístico. El teorema central del límite dice que si tenemos variables aleatorias  $X_1, X_2, \dots$  independientes entre sí y con una misma distribución<sup>1</sup> entonces la media muestral se comporta asintóticamente según una distribución normal. En concreto, si la media y varianza común a todas las variables son  $\mu$  y  $\sigma^2$  entonces

$$\lim_{n \rightarrow +\infty} P\left(\frac{\bar{X}_n - \mu}{\sqrt{n}\sigma} \leq z\right) = P(Z \leq z) \quad (22.5)$$

donde  $Z$  es una variable con distribución normal con media 0 y varianza 1, es decir, una normal estándar o típica.

### 22.4.3 Ejercicios

**Ej. 58 —** 1. Supongamos una distribución binomial con  $p = 0.5$  y  $n = 10$  y queremos calcular la probabilidad de que  $\hat{p}$  sea menor o igual a  $7/10$ . Obtener el valor exacto y el valor aproximado utilizando la aproximación dada por el teorema central del límite.

2. Repetir el punto anterior obteniendo el valor exacto y el valor aproximado de  $P(0.3 \leq \hat{p} \leq 0.7)$ .

**Ej. 59 —** 1. Supongamos una distribución binomial con  $p = 0.5$  y  $n = 100$  y queremos calcular la probabilidad de que  $\hat{p}$  sea menor o igual a  $0.55$ . Obtener el valor exacto y el valor aproximado utilizando la aproximación dada por el teorema central del límite.

2. Repetir el punto anterior obteniendo el valor exacto y el valor aproximado de  $P(0.45 \leq \hat{p} \leq 0.55)$ .

---

<sup>1</sup> En definitiva repetimos independientemente un mismo experimento y observamos una misma cantidad cada vez.



# Capítulo 23

## Estimación

### 23.1 Introducción

Tratamos el problema de la estimación. En concreto, en poblaciones normales, nos planteamos la estimación de la media y varianza. También consideramos la estimación de una proporción. Se aborda el problema del cálculo del tamaño de la muestra para estimar los parámetros con un error máximo dado.

### 23.2 La población

¿Qué es una población? La Estadística se ocupa del estudio de grandes poblaciones. Pero, otra vez, ¿y qué es una población? La respuesta no es simple ni tampoco es única.

El primer sentido que podemos dar al término *población* es una gran colección de elementos de los cuales queremos conocer algo. Algunos ejemplos son:

1. la población española a día 2 de noviembre de 2011;
2. la población de [samaruc](#) el 2 de diciembre de 2010;
3. la población de [fartet](#) en la Comunidad Valenciana en febrero de 2012;
4. Los pinos de los [Montes Universales](#);

y muchísimos ejemplos similares. Todos los ejemplos que acabamos de proponer se caracterizan porque *toda la población está ahí* y, en principio, podríamos observarla. Podemos tener la santa paciencia de observar todos y cada uno de los pinos de los Montes Universales. Estos ejemplos son poblaciones *finitas*, es decir, son grandes conjuntos de individuos pero un número finito. Tenemos interés en alguna característica de la población. Por ejemplo, ¿cuál es la longitud media de un samaruc adulto de la Albufera? ¿Cuál es la proporción de pinos en los Montes Universales afectados por la procesionaria?

¿Cómo podemos obtener este valor? Es una población finita y los peces están ahí. Pues los cogemos y medimos cada uno de ellos. La cantidad buscada no es más que la media de las longitudes de cada uno de los peces. Simple de decir sí pero, obviamente, impracticable. Aunque, sobre el papel, es una población accesible; realmente, es una población

inaccesible. *No podemos acceder a toda la población.* Por ello, hemos de considerar el experimento consistente en elegir al azar un individuo de la población. Si el individuo elegido lo denotamos por  $\omega$ , una vez lo tenemos, podemos medirlo, podemos observar la característica que nos interesa, podemos observar la variable  $X$  en el individuo  $\omega$ , siendo  $x$  el valor que observamos. Este proceso es lo que se denota breviadamente como  $X(\omega) = x$ . Sin embargo, no observamos la variable aleatoria  $X$  una sola vez. Elegimos independientemente y de la misma población de peces  $n$  individuos. Antes de elegirlos tenemos una colección de valores aleatorios que son independientes entre sí y que repiten el experimento. Esto es lo que se conoce como *muestra aleatoria* y denotamos  $\{X_1, \dots, X_n\}$ . Los valores observados una vez hemos elegido a los  $n$  peces los denotamos con  $x_1, \dots, x_n$ .

En otras ocasiones la población no es un concepto tan concreto. Por ejemplo, estamos la demanda biológica de oxígeno en muestras de agua tomadas en la playa de [Canet d'en Berenguer](#). En principio, el número de muestras es infinito. Además, aunque asumimos que las muestras tienen un comportamiento aleatorio similar en la playa de Canet, es claro que necesitamos indicar, al menos, el día en que las tomamos. En este caso, la población no existe. Son las infinitas repeticiones que podemos hacer del experimento consistente en tomar una muestra de agua y determinar la demanda biológica de oxígeno.

### 23.3 Estimación puntual

**Nota 23.1 (Estimando una media)** *Vamos a ilustrar los conceptos con una población finita. Y muy grande. Suponemos que tenemos una población de 237456 personas que conocemos la altura de cada una de las personas.<sup>1</sup> Los datos los tenemos en el vector  $X$ .*

*Por ejemplo, las estaturas de las diez primeras personas (en centímetros) son*

```
## [1] 155.0 156.6 157.4 161.9 159.2 156.0 159.3
## [8] 158.3 159.3 158.1
```

*Por  $X$  estamos denotando toda la población. Por lo tanto la media de la población o media poblacional la conocemos. Simplemente hemos de realizar la media muestral de todas las estaturas.*

```
(mu=mean(X))
```

```
## [1] 159.9931
```

*Pretendemos estimar la media poblacional (que en nuestro ejemplo artificial conocemos) utilizando una muestra aleatoria de 10 individuos (una muestra no muy grande ciertamente). Podemos elegir la muestra aleatoria con la función sample. Por ejemplo, una primera muestra estaría compuesta por los individuos*

```
n = 10
(x = sample(X,n))
```

---

<sup>1</sup> Hemos tenido la [santa paciencia](#) de medir la estatura de cada uno de ellos. Y ellos se han dejado.

```
## [1] 157.4714 158.0356 163.9513 164.0696 151.9937
## [6] 164.0996 159.1570 156.9069 166.9971 165.5937
```

*Nuestra estimación sería*

```
(mediamuestral = mean(x))
## [1] 160.8276
```

*de modo que el error que cometemos es*

```
mediamuestral - mu
## [1] 0.834481
```

*Supongamos que repetimos la estimación varias veces y veamos los diez primeros valores que obtenemos*

```
## [1] 159.1075 160.4753 160.6225 158.7048 159.1453
## [6] 157.2618 155.9279 163.4243 158.8823 162.1287
```

*Veamos un resumen de los errores observados.*

```
errores = estimaciones-mu
summary(errores)

##      Min.    1st Qu.     Median      Mean    3rd Qu.
## -4.06524 -1.29182 -0.00975 -0.03127  1.26638
##          Max.
##   4.50297
```

*De hecho, en la figura 23.1 hemos representado un estimador kernel de las estimaciones y una línea vertical mostrando la media real.*

*Así es como funciona la estimación puntual. Hemos visto una situación irreal en que tenemos todos los valores que componen la población y, por lo tanto, podemos conocer la media de la población. Esto no es nunca así. Conocemos una muestra aleatoria de la población que denotaremos por  $X_1, \dots, X_n$  y, con estos valores, pretendemos estimar la media de la población  $\mu$ . De hecho, hemos utilizado estos valores y obtenido un método para estimar  $\mu$ , a esto le podemos llamar un estimador de  $\mu$  y denotarlo como*

$$\hat{\mu} = \bar{X}_n = \sum_{i=1}^n \frac{X_i}{n}.$$

*Antes de tomar la muestra,  $\bar{X}_n$  es un valor aleatorio pero después de tomar la muestra tenemos que  $X_1 = x_1, \dots, X_n = x_n$ , es decir, que no son valores aleatorios sino unos valores dados. En el ejemplo anterior, antes de utilizar la función sample tenemos un valor aleatorio pero después tenemos el valor observado correspondiente. Por lo tanto una vez seleccionada la muestra decimos que tenemos los valores observados  $x_1, \dots, x_n$  y la variable aleatoria  $\bar{X}_n$  toma el valor fijo  $\bar{x}_n = \sum_{i=1}^n x_i/n$ .*

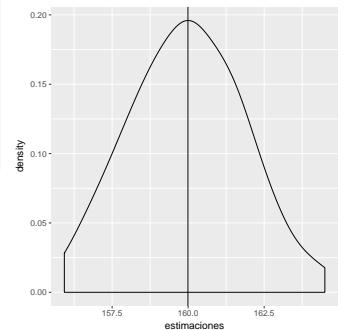


Figura 23.1: Estimador kernel de la densidad de las estimaciones obtenidas eligiendo muestras de tamaño 10 de la población. La línea vertical tiene como abscisa común la media poblacional.

## 23.4 Algunas definiciones

**Definición 23.1 (Muestra aleatoria)** Una muestra aleatoria (de tamaño  $n$ ) son  $n$  valores aleatorios  $X_1, \dots, X_n$  que tienen la misma distribución y son independientes entre sí.

**Definición 23.2 (Estimador)** Un estimador es cualquier función de la muestra aleatoria  $X_1, \dots, X_n$  que toma valores admisibles para el parámetro que estimamos.

## 23.5 Estimación puntual de la media

Si pretendemos estimar la media  $\mu$  de una población su estimador usual es la media muestral,  $\bar{X}_n$ . Si tenemos una muestra aleatoria donde cada  $X_i$  sigue una distribución normal entonces

$$\bar{X}_n \sim N\left(\mu, \frac{\sigma^2}{n}\right) \quad (23.1)$$

Si las distintas variables  $X_i$  que componen la muestra no siguen una distribución muestral entonces asumiendo que tenemos una muestra grande el resultado que damos en 23.1 es aproximadamente cierto. En cualquier caso, la varianza de la media muestral  $\bar{X}_n$

## 23.6 Intervalo de confianza para la media

Hemos considerado en el apartado anterior la estimación puntual de la media poblacional,  $\mu$ , mediante la media muestral,  $\bar{X}$ . Es una opción natural. Otra opción puede ser estimar la media de la población  $\mu$  dando un intervalo que la contenga. En lugar de decir “estimamos la media poblacional en 161.19”, esto es utilizar un valor numérico exclusivamente podemos utilizar una expresión como “la media poblacional es mayor que 158.37 y menor que 162.79”. Este segundo tipo de estimación es la que se hace con un intervalo de confianza. En resumen estimamos la media poblacional  $\mu$  o bien mediante un punto (estimador puntual) o bien mediante un intervalo (estimación por intervalos). Al primer método se le llama *estimador puntual* y al segundo *intervalo de confianza*.

### 23.6.1 Asumimos que conocemos la varianza

Vamos a asumir en un primer momento algo que no tiene ninguna realidad (raramente nos lo vamos a encontrar en una aplicación real) pero nos facilita la presentación. En concreto asumimos que no conocemos la media  $\mu$  (es lo que queremos estimar) pero, sin embargo, sí conocemos la desviación estándar,  $\sigma$ .

Una segunda hipótesis (que sí se verifica con frecuencia) que vamos a asumir es que los datos proceden de una población normal. Ya discutiremos qué hacemos después con cada una de estas hipótesis.

Siendo  $\mu$  y  $\sigma$  la media y la desviación típica reales, por el teorema central del límite se verifica que aproximadamente (o con más precisión si el tamaño de la muestra aleatoria,  $n$ , es grande) la media muestral tiene una distribución normal con media  $\mu$  y desviación estándar  $\sigma/\sqrt{n}$ ,

$$\bar{X} \sim N\left(\mu, \frac{\sigma^2}{n}\right),$$

o, lo que es equivalente, que

$$\sqrt{n} \frac{\bar{X} - \mu}{\sigma} \sim N(0, 1),$$

donde  $N(0, 1)$  es una normal con media cero y varianza uno, lo que se conoce como una *normal estándar* o *normal típica*.<sup>2</sup>

Notemos que, *asumiendo la desviación estándar conocida*, en la expresión  $\sqrt{n} \frac{\bar{X} - \mu}{\sigma}$  conocemos todos los términos que aparecen ( $\bar{X}, \sigma$  y  $n$ ) una vez hemos tomado la muestra salvo el valor de la media poblacional  $\mu$ . Precisamente es lo que queremos estimar.

Fijemos una probabilidad alta, por ejemplo, una probabilidad de 0.95. Podemos determinar un valor positivo  $c$  tal que

$$P\left(-c \leq \sqrt{n} \frac{\bar{X} - \mu}{\sigma} \leq c\right) = 0.95$$

Por las propiedades de simetría de la normal  $c$  ha de verificar que

$$P\left(\frac{\bar{X} - \mu}{\sigma} \leq c\right) = 0.975$$

Podemos determinarlo con la función *qnorm* del siguiente modo.

```
qnorm(0.975, mean=0, sd=1)
## [1] 1.959964
```

En resumen que

$$P\left(-1.96 \leq \sqrt{n} \frac{\bar{X} - \mu}{\sigma} \leq 1.96\right) = 0.95$$

o, lo que es equivalente,

$$P\left(\bar{X} - 1.96 \frac{\sigma}{\sqrt{n}} \leq \mu \leq \bar{X} + 1.96 \frac{\sigma}{\sqrt{n}}\right) = 0.95.$$

Vemos que el intervalo  $[\bar{X} - 1.96 \frac{\sigma}{\sqrt{n}}, \bar{X} + 1.96 \frac{\sigma}{\sqrt{n}}]$  tiene una probabilidad de 0.95 de contener a  $\mu$  o también de *cubrir* a  $\mu$ . Si ahora sustituimos los valores aleatorios  $X_i$  con  $i = 1, \dots, n$  con los valores observados en la muestra entonces el intervalo aleatorio  $[\bar{X} - 1.96 \frac{\sigma}{\sqrt{n}}, \bar{X} + 1.96 \frac{\sigma}{\sqrt{n}}]$  pasa a ser un intervalo fijo  $[\bar{x} - 1.96 \frac{\sigma}{\sqrt{n}}, \bar{x} + 1.96 \frac{\sigma}{\sqrt{n}}]$  que conocemos como *intervalo de confianza con nivel de confianza* 0.95.

**Nota de R 23.1 (Intervalo de confianza con R)** *Vamos a evaluarlo con R. Empezamos tomando los datos.*

```
(x = sample(X, 10))
## [1] 163.7176 168.5702 154.3190 157.4909 166.1140
## [6] 156.4414 159.6397 165.5520 158.5075 167.7663
```

---

<sup>2</sup>Es equivalente porque si una variable aleatoria  $X$  suponemos que tiene distribución normal con media  $\mu$  y varianza  $\sigma^2$  entonces la variable aleatoria  $(X - \mu)/\sigma$  sigue una distribución normal con media 0 y varianza 1 que se conoce como una normal típica o normal estándar. Esto se suele indicar como:  $X \sim N(\mu, \sigma^2)$  entonces  $(X - \mu)/\sigma \sim N(0, 1)$

Determinamos la media muestral y la desviación estándar muestral.

```
media = mean(x)
s = sd(x)
```

Por lo tanto el intervalo tendrá por extremo inferior

```
(extremo.inferior = mean(x) - qnorm(0.975,mean=0,sd=1)*sd(x)/sqrt(n))
## [1] 158.6343
```

y por extremo superior.

```
(extremo.superior = mean(x) + qnorm(0.975,mean=0,sd=1)*sd(x)/sqrt(n))
## [1] 164.9894
```

Si en lugar de 0.95 elegimos como nivel de confianza (utilizando la notación habitual)  $1 - \alpha$  con  $\alpha$  un valor pequeño (en el ejemplo anterior  $\alpha = 0.05$ ) podemos determinar  $c$  que verifique

$$P\left(-c \leq \sqrt{n} \frac{\bar{X} - \mu}{\sigma} \leq c\right) = 1 - \alpha$$

o que

$$P\left(\frac{\bar{X} - \mu}{\sigma} \leq c\right) = 1 - \frac{\alpha}{2}.$$

Denotamos el valor de  $c$  que verifica lo anterior como  $Z_{1-\frac{\alpha}{2}}$ . Finalmente el intervalo

$$\left[\bar{X} - Z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}, \bar{X} + Z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}\right]$$

cubre a  $\mu$  con una probabilidad de  $1 - \alpha$  y el intervalo de confianza es el que obtenemos cuando sustituimos los valores aleatorios por los valores observados. Es decir, el intervalo de confianza viene dado por

$$\left[\bar{x} - Z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}, \bar{x} + Z_{1-\frac{\alpha}{2}} \frac{\sigma}{\sqrt{n}}\right]$$

De un modo genérico: ¿Cómo podemos determinar el intervalo de confianza que acabamos de ver? Fijamos el nivel de confianza  $1 - \alpha$  (en este caso a  $1 - \alpha = 0.99$  o equivalentemente  $\alpha = 0.01$ ).

```
alpha = 0.01
```

Determinamos el extremo inferior:

```
(extremo.inferior = mean(x) - qnorm(1-alpha/2,mean=0,sd=1)*sd(x)/sqrt(n))
## [1] 157.6359
```

y el superior

```
(extremo.superior = mean(x) + qnorm(1-alpha/2,mean=0,sd=1)*sd(x)/sqrt(n))
## [1] 165.9879
```

El intervalo es el siguiente:

```
c(extremo.inferior,extremo.superior)
## [1] 157.6359 165.9879
```

*El último paso es rogar a Dios que las cosas hayan ido bien. Tenemos una confianza de  $1 - \alpha$  (0.99 en el ejemplo) de que el valor real de la media esté en este intervalo. Pero esto no quiere decir que realmente lo está. Si repetimos un gran número de veces el valor real de la media está en el intervalo un  $(1 - \alpha) \times 100\%$  de la veces (un 99% en el ejemplo) pero puede ocurrir (desgracias de la vida) que estemos en el  $\alpha \times 100$  (en el ejemplo un 1%) restante. En general la cosa va bien porque elegimos un nivel de confianza grande (próximo a uno) pero no siempre va bien.*

### 23.6.2 No asumimos la varianza conocida

No es realista asumir que conocemos la varianza  $\sigma^2$  (o la desviación típica  $\sigma$ ). En absoluto lo es. ¿Por qué vamos a desconocer la media y conocer la varianza? <sup>3</sup> Salvo situaciones realmente esotéricas esto no es así. En consecuencia lo lógico es sustituir la desviación típica poblacional  $\sigma$  por su estimador natural  $S = \sqrt{\frac{1}{n-1} \sum_{i=1}^n (X_i - \bar{X})^2}$ . Si lo hacemos tendremos la cantidad

$$\sqrt{n} \frac{\bar{X} - \mu}{S}. \quad (23.2)$$

En la expresión anterior lo conocemos todo (una vez tenemos los datos) excepto el valor de  $\mu$ . Sin embargo, la distribución de probabilidad de esta cantidad *ya no es una normal estándar*. Nos aparece una distribución de probabilidad nueva que se conoce como la distribución t de Student. <sup>4</sup>

De hecho, William Sealy Gossett demostró que la cantidad  $\sqrt{n} \frac{\bar{X} - \mu}{S}$  se comporta como una t de Student con  $n - 1$  grados de libertad. Esto lo denotaremos como

$$T = \sqrt{n} \frac{\bar{X} - \mu}{S} \sim t_{n-1}. \quad (23.3)$$

En la figura 23.2 hemos representado la función de densidad de una t de Student con 9 grados de libertad.

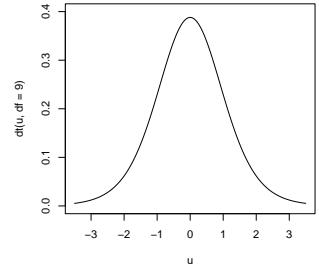


Figura 23.2: Función de densidad de una t de Student con 9 grados de libertad

<sup>3</sup> ¿Van por ahí los datos diciendo *somos normales no te decimos la media pero te decimos la varianza?* Los pobres datos no dicen nada.

<sup>4</sup> La expresión exacta de esta densidad es  $f(x) = \frac{\Gamma(\frac{n+1}{2})}{\sqrt{n}\pi} \left(1 + \frac{x^2}{n}\right)^{-(n+1)/2}$  donde

$\Gamma(x) = \int_0^{+\infty} t^{x-1} e^{-t} dt$  es la función Gamma de Euler. La miráis y ya está. No hace falta saberla pero por lo menos es bueno verla una vez en la vida.

**Nota 23.2** (¿Qué relación tiene la t de Student con la normal?) *Hay dos puntos interesantes a tener en cuenta cuando utilizamos una distribución t de Student. La primera es qué relación tiene con una distribución normal estándar y la segunda es qué ocurre cuando modificamos los grados de libertad.*

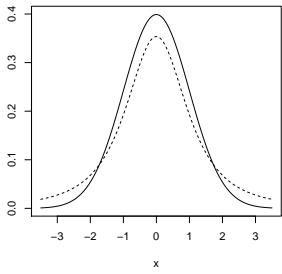


Figura 23.3: Funciones de densidad de la normal estándar (trazo continuo) y de densidades t de Student con 2 grados de libertad.

Para ver la relación con la normal estándar hemos representado en la figura 23.3 la densidad de la normal en trazo continuo y la densidad de una t de Student con 2 grados de libertad en trazo discontinuo. Vemos que tienen una forma similar, ambas están centradas en cero. Sin embargo, la densidad de la normal está más concentrada alrededor de cero. La densidad de la t de Student está más repartida. Y esto ocurre para cualquier número de grados de libertad.

¿Qué ocurre cuando incrementamos el número de grados de libertad? Cuando se va incrementando el número de grados la densidad de la t de Student se aproxima a la densidad de la normal. En la figura 23.4 se ilustra y comenta este hecho.

Y ahora vamos a repetir lo visto en la sección anterior sustituyendo a la normal estándar con la densidad de la t de Student con  $n-1$  grados de libertad. Dada una probabilidad, por ejemplo 0.95, podemos determinar el valor  $c$  tal que

$$P(-c \leq T \leq c) = 0.95.$$

En concreto verificará que

$$P(T \leq c) = 0.975$$

y el valor de  $c$  (por ejemplo, para 9 grados de libertad) lo obtendremos con

`qt(0.975, df=9)`

```
## [1] 2.262157
```

Denotamos el valor de  $c$  tal que

$$P(T \leq c) = 1 - \frac{\alpha}{2},$$

como  $t_{n-1,1-\alpha/2}$ . Entonces

$$P\left(\bar{X} - t_{n-1,1-\alpha/2} \frac{S}{\sqrt{n}} \leq \mu \leq \bar{X} + t_{n-1,1-\alpha/2} \frac{S}{\sqrt{n}}\right) = 1 - \alpha. \quad (23.4)$$

El intervalo de confianza lo obtenemos sustituyendo los valores aleatorios por los valores observados.

**Teorema 23.1** *Si suponemos que tenemos una muestra aleatoria de datos normales  $X_1, \dots, X_n$  y observamos los datos  $X_1 = x_1, \dots, X_n = x_n$  entonces el intervalo*

$$\left[ \bar{x} - t_{n-1,1-\alpha/2} \frac{s}{\sqrt{n}}, \bar{x} + t_{n-1,1-\alpha/2} \frac{s}{\sqrt{n}} \right]$$

es un intervalo de confianza con nivel de confianza  $1 - \alpha$  para la media  $\mu$ . De un modo abreviado el intervalo anterior se puede escribir como

$$\bar{x} \pm t_{n-1,1-\alpha/2} \frac{s}{\sqrt{n}}$$

**Nota de R 23.2 (Cálculo del intervalo de confianza con R)** ¿Y cómo hacerlo con R? Fijamos el nivel de confianza (en este caso a 0.99).

```
alpha = 0.01
```

Determinamos el extremo inferior:

```
(extremo.inferior = mean(x) - qt(1-alpha/2,df=n-1)*sd(x)/sqrt(n))
## [1] 156.5431
```

y el superior

```
(extremo.superior = mean(x) + qt(1-alpha/2,df=n-1)*sd(x)/sqrt(n))
## [1] 167.0806
```

El intervalo es el siguiente

```
c(extremo.inferior,extremo.superior)
## [1] 156.5431 167.0806
```

<sup>5</sup>

**Nota 23.3 (Obtención del intervalo de confianza utilizando t.test)** No hace falta escribir todo lo anterior para calcular el intervalo de confianza. El intervalo de confianza para la media lo podemos obtener con la función t.test. De hecho, nos da el intervalo de confianza y alguna cosa más que, de momento, no necesitamos.

```
alpha = 0.05
t.test(x,conf.level=1-alpha)

##
##  One Sample t-test
##
## data: x
## t = 99.808, df = 9, p-value = 5.161e-15
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
## 158.1444 165.4793
## sample estimates:
## mean of x
## 161.8119
```

Si queremos una salida en la que solamente nos aparezca el intervalo de confianza podemos hacer

```
alpha = 0.05
t.test(x,conf.level=1-alpha)$conf.int

## [1] 158.1444 165.4793
## attr(,"conf.level")
## [1] 0.95
```

---

<sup>5</sup>Podemos comprobar que el intervalo que obtenemos asumiendo varianza conocida es más pequeño que el que obtenemos asumiendo varianza desconocida.

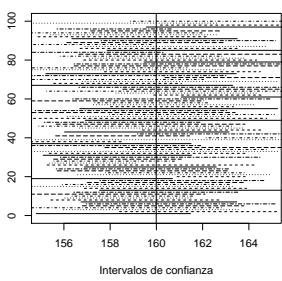


Figura 23.5: Intervalos de confianza de la media en una población normal. Hemos simulado 100 intervalos. La línea vertical tiene como abscisa el valor real de la media. ¿Cuántos intervalos no contienen a la media real? Cuéntalos.

**Nota 23.4 (Evaluando el intervalo de confianza)** Podemos repetir el proceso de estimación muchas veces y ver en cuántas ocasiones el intervalo contiene al verdadero valor de la media. Generamos 100 intervalos con muestras de tamaño  $n = 100$ . En la figura 23.5 hemos representado los 100 intervalos que hemos obtenido. La línea vertical indica la media poblacional. Cada segmento horizontal se ha dibujado de modo que las abcisas de sus extremos corresponden con el extremo inferior y superior del correspondiente intervalo de confianza.

**Ejemplo 23.1 (Datos gse21942)** Vamos a utilizar los datos tamida-ta::gse21942.

```
library(Biobase)
data(gse21942, package="tamidata")
y = exprs(gse21942)[54651,]
x = pData(gse21942)[, "FactorValue..DISEASE.STATE."]
y0 = y[x == "multiple sclerosis"]
```

Vamos a obtener el intervalo de confianza para la expresión media. La función básica es t.test.

```
t.test(y0)

##
##  One Sample t-test
##
## data: y0
## t = 7.5787, df = 13, p-value = 4.023e-06
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
##  3.517188 6.321905
## sample estimates:
## mean of x
## 4.919546
```

Si queremos obtener el intervalo de confianza solamente podemos hacer lo siguiente.

```
y0.t = t.test(y0)
y0.t$conf.int

## [1] 3.517188 6.321905
## attr(,"conf.level")
## [1] 0.95
```

o simplemente

```
t.test(y0)$conf.int

## [1] 3.517188 6.321905
## attr(,"conf.level")
## [1] 0.95
```

Por defecto el nivel de confianza que se elige es 0.95. Podemos modificarlo. Bien bajándolo

```
t.test(y0,conf.level=.90)$conf.int
## [1] 3.769982 6.069111
## attr(,"conf.level")
## [1] 0.9
```

*o bien tomando un nivel de confianza mayor, 0.99.*

```
t.test(y0,conf.level=.99)$conf.int
## [1] 2.964190 6.874903
## attr(,"conf.level")
## [1] 0.99
```

**Ejemplo 23.2** *Lo vamos a ilustrar con datos de los resultados de la selectividad en la Comunidad Valenciana. Estos datos se pueden encontrar en [http://www.edu.gva.es/univ/val/prueba\\_acceso.htm](http://www.edu.gva.es/univ/val/prueba_acceso.htm). Los datos nos dan la media y desviación estándar muestrales de las notas obtenidas en Matemáticas II en el examen de selectividad en la Comunidad Valenciana el año 2010. Tenemos estos valores distinguiendo por universidad y si los resultados se han obtenido en la fase general o en la fase específica. Tenemos unos datos agregados. Conocemos unos resúmenes de los datos pero no los datos mismos.<sup>6</sup>*

*Utilizando los datos de la tabla 23.1 vamos a calcular intervalos de confianza para la nota media en cada una de las universidades y en el total de todas las universidades. Vamos a centrarnos en los resultados globales, esto es, la última línea de la tabla. Empezamos construyendo un intervalo de confianza para la media global. En el siguiente código*

```
media = 5.021
desviacion.estandar = 2.077
n = 4499
alpha = .01
(extremoinferior = media - qt(1-alpha/2,df=n-1) *
desviacion.estandar/ sqrt(n))

## [1] 4.941204

(extremosuperior = media + qt(1-alpha/2,df=n-1) *
desviacion.estandar / sqrt(n))

## [1] 5.100796
```

*De modo que el intervalo de confianza viene dado por [4.941,5.101]. Y ahora para la nota en la fase general.*

```
media = 4.969
desviacion.estandar = 1.909
n = 1724
alpha = .01
```

---

<sup>6</sup>No es tan extraña esta situación. Con frecuencia cuando lees un informe o bien un artículo científico no sueles disponer de los datos originales sino de los resúmenes que de los mismos proporcionan los autores en la publicación. Tiene sentido e interés ver cómo calcular estos intervalos a partir de los datos resumidos.

Tabla 23.1: Resumen de los resultados de Matemáticas II. Las etiquetas indican: *Matric.*, matriculados; *Present.*, presentados; *Aptos*, aptos; *Media*, nota media; *DE*, desviación estándar; *Present. FG*, presentados fase general; *Present. FE*, presentados fase específica; *Aprob. FE*, aprobados fase específica; *Media FG*, media fase general; *DE FG*, desviación típica fase general; *Media FE*, media fase específica; *DE FG*, desviación típica fase específica. En filas tenemos las universidades de Alicante (UA), la Jaume I de Castellón (UJI), la Miguel Hernández de Elche (UMH), la Universidad de Valencia (UV) y todos los estudiantes en el Sistema de Universidades Valencianas (SUV).

Universidad	Matric.	Present.	Aptos	Media	DE
UA	783	771	401	4,989	2,124
UJI	530	518	249	4,857	2,088
UMH	693	673	262	4,369	1,922
UPV	1073	1049	589	5,274	2,070
UV	1525	1488	851	5,212	2,053
SUV	4604	4499	2352	5,021	2,077

Universidad	Present. FG	Present. FE	Aprob. FE
UA	276	495	254
UJI	213	305	148
UMH	267	406	154
UPV	422	627	357
UV	546	942	537
SUV	1724	2775	1450

Universidad	Media FG	DE FG	Media FE	DE FE
UA	5,053	1,839	4,954	2,267
UJI	4,852	1,874	4,860	2,225
UMH	4,304	1,812	4,412	1,989
UPV	5,135	1,959	5,367	2,137
UV	5,170	1,89	5,237	2,141
SUV	4,969	1,909	5,054	2,174

```
(extremoinferior =
  media - qt(1-alpha/2,df=n-1) * desviacion.estandar/ sqrt(n))

## [1] 4.850441

(extremosuperior =
  media + qt(1-alpha/2,df=n-1) * desviacion.estandar / sqrt(n))

## [1] 5.087559
```

*El intervalo es [4.85, 5.088]. Y terminamos con la media en la fase específica.*

```
media = 5.054
desviacion.estandar = 2.174
n = 2775
alpha = .01
(extremoinferior = media - qt(1-alpha/2,df=n-1) *
  desviacion.estandar/ sqrt(n))

## [1] 4.947624

(extremosuperior = media + qt(1-alpha/2,df=n-1) *
  desviacion.estandar / sqrt(n))

## [1] 5.160376
```

## 23.7 Error absoluto y tamaño de la muestra

En esta sección nos planteamos (por primera vez) un problema de cálculo de tamaño de la muestra. ¿Cuántos datos hemos de tener para que nuestro estudio tenga validez? Es una pregunta muy genérica sin respuesta. La respuesta necesita que concretemos más lo que queremos de nuestros datos. En esta sección nos planteamos la siguiente pregunta: ¿Cuántos datos necesitamos para que cuantos estimamos una media poblacional el error máximo que cometemos sea menor que una cantidad que previamente especificamos? Por ejemplo, queremos conocer la concentración media de un elemento en una zona. Queremos conocer esta cantidad, denotada por  $\mu$ , con un error máximo de  $\delta$  unidades siendo  $\delta$  una cantidad positiva que fijamos nosotros. Lo primero que hemos de tener en cuenta es que en Estadística nunca podemos afirmar *con seguridad* nada. Podemos pedir que sea muy probable o, mejor, que tengamos *muchísima confianza* en que ocurra pero que *seguro* que ocurra es mucho pedir. Siempre hacemos afirmaciones basadas en la probabilidad de sucesos que pueden o no ocurrir y, por lo tanto, afirmar que nuestro error va a ser menor que un cierto nivel  $\delta$  *siempre* o *seguro* no es posible.

Vamos a responder a la pregunta anterior utilizando los datos tamaño::gse21942. Pretendemos estimar la expresión media en esclerosis múltiple del gen en fila 54651. Ya hemos determinado un intervalo de confianza para  $\mu$  con un nivel  $1 - \alpha$ . Por ejemplo, con  $\alpha = 0.05$  el intervalo es

```
t.test(y0,conf.level=.95)$conf.int
## [1] 3.517188 6.321905
## attr(,"conf.level")
## [1] 0.95
```

Tenemos una confianza de 0.95 de que el valor de  $\mu$  esté dentro (esté cubierto por) el intervalo anterior. Asumamos que la cosa ha ido bien, esto es, asumamos que  $\mu$  está dentro. No lo sabemos pero *confiamos* con un nivel de confianza 0.95 que esto sea así. Pues bien, lo asumimos. A la pregunta: ¿en cuánto estimas  $\mu$ ?, respondemos (sin dudar) que en

```
mean(y0)
## [1] 4.919546
```

esto es, respondemos dando el valor de la media muestral  $\bar{x}$  de las concentraciones utilizadas para calcular el intervalo. Esta media muestral es el punto medio del intervalo. Por lo tanto si  $\mu$  está en intervalo entonces la diferencia entre  $\mu$  y el centro es menor o igual que la mitad de la longitud de dicho intervalo que vale

```
## [1] 2.804717
```

¿Qué error absoluto tenemos con la muestra que se tomó? Guardamos el intervalo de confianza en *y0.ci*.

```
y0.ci = t.test(y0,conf.level=.95)$conf.int
```

Sabemos que el error absoluto es la mitad de la longitud del intervalo de confianza. Lo calculamos.

```
(y0.ci[2] - y0.ci[1]) / 2
## [1] 1.402359
```

Supongamos que nos parece excesivo y pretendemos un error absoluto máximo de 0.5.

¿Cuál era el tamaño muestral original?

```
length(y0)
## [1] 14
```

El intervalo de confianza para la media de una población normal viene dado por la siguiente expresión.

$$\left[ \bar{X} - t_{n-1,1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}}, \bar{X} + t_{n-1,1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}} \right]$$

En consecuencia, el error absoluto vendrá dado por

$$t_{n-1,1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}}$$

Esta expresión en R se calcula con el siguiente código.

```
alpha = .05
n = length(y0)
qt(1-alpha/2,df=n-1)*sd(y0)/sqrt(n)

## [1] 1.402359
```

Suponemos que la desviación estándar no se va a modificar mucho si tomamos más muestras.

```
sd0 = sd(y0)
```

Fijamos un valor para el error que queremos cometer.

```
delta = 0.2
```

Se tiene que verificar

$$t_{n-1,1-\frac{\alpha}{2}} \frac{S}{\sqrt{n}} \leq \delta$$

Si asumimos que  $s$  es aproximadamente constante.

```
m = n + 10
qt(1-alpha/2,df=m-1)*sd(y0)/sqrt(m)

## [1] 1.0256
```

Vemos que no es suficiente. ¿Y 100 observaciones más?

```
m = n + 100
qt(1-alpha/2,df=m-1)*sd(y0)/sqrt(m)

## [1] 0.4506785
```

Quizás nos hemos pasado. Veamos con 90.

```
m = n + 90
qt(1-alpha/2,df=m-1)*sd(y0)/sqrt(m)

## [1] 0.4723448
```

Una manera de hacerlo sería

```
for(m in 80:95)
  print(c(m,qt(1-alpha/2,df=m-1)*sd(y0)/sqrt(m)))

## [1] 80.000000 0.5405073
## [1] 81.000000 0.5370561
## [1] 82.000000 0.5336702
## [1] 83.000000 0.5303475
## [1] 84.000000 0.5270861
## [1] 85.000000 0.5238842
## [1] 86.00000 0.52074
## [1] 87.000000 0.5176517
## [1] 88.000000 0.5146177
## [1] 89.000000 0.5116364
```

```
## [1] 90.000000 0.5087064
## [1] 91.000000 0.5058261
## [1] 92.000000 0.5029943
## [1] 93.000000 0.5002094
## [1] 94.000000 0.4974704
## [1] 95.000000 0.4947758
```

## 23.8 Estimación de la varianza en poblaciones normales

Si  $X_1, \dots, X_n$  es una muestra aleatoria de una normal con media  $\mu$  y varianza  $\sigma^2$  entonces

$$\sum_{i=1}^n \frac{(X_i - \bar{X}_n)^2}{\sigma^2} = \frac{(n-1)S^2}{\sigma^2} \sim \chi_{n-1}^2.$$

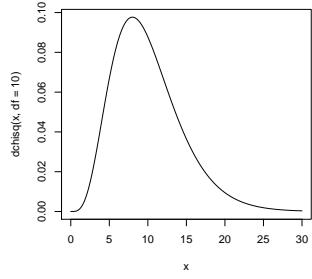


Figura 23.6: Función de densidad de una ji-cuadrado con 10 grados de libertad.

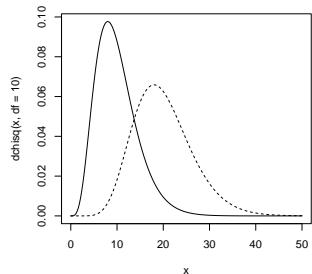
Estamos diciendo que la variable aleatoria  $\frac{(n-1)S^2}{\sigma^2}$  tiene una distribución ji-cuadrado con  $n-1$  grados de libertad.<sup>7</sup> En la figura 23.6 hemos representado la función de densidad de una ji-cuadrado con 10 grados de libertad.

Con objeto de ver cómo cambia la forma de la función de densidad cuando cambia el número de grados de libertad en la figura 23.7 mostramos las densidades de la ji-cuadrado con 10 grados de libertad (trazo continuo) y con 20 grados de libertad (trazo discontinuo).

Denotamos el percentil de orden  $p$  de una ji-cuadrado con  $k$  grados de libertad como  $\chi_{p,k}$ . Es decir, si la variable  $X$  se distribuye como una ji-cuadrado con  $k$  grados de libertad,  $X \sim \sim \chi_k^2$ , entonces

$$P(X \leq \chi_{p,k}^2) = p.$$

El valor anterior lo podemos obtener con la función `qchisq`.



Entonces tenemos que

$$P\left(\chi_{\alpha/2,n-1}^2 \leq \frac{(n-1)S^2}{\sigma^2} \leq \chi_{1-\alpha/2,n-1}^2\right) = 1 - \alpha.$$

Y por lo tanto,

$$P\left(\frac{(n-1)S^2}{\chi_{1-\alpha/2,n-1}^2} \leq \sigma^2 \leq \frac{(n-1)S^2}{\chi_{\alpha/2,n-1}^2}\right) = 1 - \alpha.$$

<sup>7</sup> La densidad de una distribución ji-cuadrado con  $k$  grados de libertad es

$$f(x) = \frac{1}{2^{k/2}\Gamma(k/2)} x^{\frac{k}{2}-1} e^{-\frac{x}{2}} \text{ para } x \geq 0 \text{ y cero en otro caso.}$$

Es decir el intervalo

$$\left[ \frac{(n-1)S^2}{\chi_{1-\alpha/2,n-1}^2}, \frac{(n-1)S^2}{\chi_{\alpha/2,n-1}^2} \right]$$

es un intervalo de confianza para  $\sigma^2$  con nivel de confianza  $1 - \alpha$ .

Es claro que el correspondiente intervalo de confianza (con nivel de confianza  $1 - \alpha$ ) para la desviación estándar poblacional vendrá dado por

$$\left[ \sqrt{\frac{(n-1)S^2}{\chi_{1-\alpha/2,n-1}^2}}, \sqrt{\frac{(n-1)S^2}{\chi_{\alpha/2,n-1}^2}} \right]$$

Veamos cómo hacerlo con R. La función que nos da los percentiles de la ji-cuadrado es `pchisq`. Tomamos los datos.

```
n = 100
x = sample(x, 100)
```

Y construimos el intervalo de confianza.

```
alpha = .05
s2 = var(x)
(extremoinferior = (n-1)*s2 / qchisq(1-alpha/2, df=n-1))

## [1] 21.71652

(extremosuperior = (n-1)*s2 / qchisq(alpha/2, df=n-1))

## [1] 38.01578
```

El intervalo de confianza para  $\sigma^2$  con un nivel de confianza de 0.95 es [21.717, 38.016].



## Capítulo 24

# Contraste de hipótesis

### 24.1 Introducción

Se introduce el problema del contraste de hipótesis en una población.

En concreto, vamos a asumir que tenemos una muestra aleatoria de una distribución normal,  $X_1, \dots, X_n$ , variables aleatorias independientes y con una misma distribución. La distribución común que asumimos es normal con media  $\mu$  y varianza  $\sigma^2$ . Es decir, estamos asumiendo que

$$X_i \sim N(\mu, \sigma^2)$$

y que los distintos valores son independientes entre si.

En este tema nos planteamos el problema del contraste de hipótesis y lo estudiamos estudiando, fundamentalmente, los contrastes sobre la media  $\mu$  de la variable que observamos (concentración de contaminante, nivel de radiación o de ruido).

Asumimos normalidad en los datos con los que trabajamos. Pero: ¿es razonable esta hipótesis? Nos ocupamos al final de este tema de lo que se conoce como contrastes de normalidad.

### 24.2 Contrastados para una muestra

Vamos a plantear el problema del contraste de hipótesis utilizando el problema de una muestra en poblaciones normales.

#### 24.2.1 Un contraste unilateral

Empezamos comentando un ejemplo que nos ayude a entender el problema del contraste de hipótesis.

**Ejemplo 24.1 (Un problema inventado)** *Un fabricante de bombillas afirma que sus bombillas tienen una duración media de al menos 1500 horas. Muy bien, esta persona afirma esto. Nuestro problema es tomar una entre dos decisiones. Admitir que lo que afirma es correcto o bien que no lo es y la duración media real es claramente menor que 1500. Lo primero que necesitamos para tomar la decisión son datos. Hemos tomado una muestra de bombillas y hemos repetido el experimento consistentemente en tenerlas en funcionamiento ininterrumpido hasta que la bombilla deja de funcionar.<sup>1</sup> En definitiva tenemos una muestra aleatoria de*

<sup>1</sup> Alguna bombilla dura algo más. <http://www.centennialbulb.org/index.htm>.

*duraciones de bombillas de este fabricante. Supongamos que las duraciones observadas son*

```
## [1] 1742.3 1306.8 1310.5 1509.2 1211.7 1177.3
## [7] 1622.3 1051.9 1470.8 1408.8 1550.2 1375.1
## [13] 1210.3 2091.8 1607.8 1396.2 1307.0 1427.5
## [19] 1607.4 1431.1 2086.8 1667.8 1659.9 1495.3
## [25] 1251.4 1135.3 1595.0 1612.1 1328.9 1555.0
## [31] 1939.2 1992.0 1555.5 1902.2 1261.7 1920.0
## [37] 1419.2 1927.3 1188.9 1363.4 1458.5 1706.3
## [43] 2032.8 1751.9 1766.6 1111.4 1672.1 1602.6
## [49] 1491.4 1636.5 1396.7 1614.8 1809.7 1529.5
## [55] 2070.2 2021.9 1892.9 1661.3 1468.7 1265.1
## [61] 1569.9 1753.7 1986.8 1324.9 1373.4 1785.6
## [67] 1624.5 962.3 1484.7 1905.8 1884.8 1640.5
## [73] 1912.0 1794.9 1497.7 2158.2 2222.1 1796.2
## [79] 1183.5 1314.7 1272.7 1439.5 1928.9 2304.5
## [85] 2079.1 2016.7 1493.3 1590.0 1421.1 1943.5
## [91] 1951.3 1378.1 1195.5 1435.4 1395.1 1381.3
## [97] 1345.8 1827.2 1646.1 1845.5
```

*La afirmación del fabricante la consideramos como una hipótesis que hemos de evaluar. En concreto nos planteamos dicha hipótesis y su negación. De un modo más preciso la hipótesis de una duración media menor o igual a 1500 y su negación serían*

$$\begin{aligned} H_0 : \mu &\leq 1500, \\ H_1 : \mu &> 1500, \end{aligned}$$

*donde  $H_0$  y  $H_1$  son las hipótesis nula y alternativa respectivamente.<sup>2</sup> Hemos de elegir entre una de ellas. Elegir supone decidir. Hemos de decidir, con los datos que tenemos sobre las duraciones de las bombillas, cuál de las hipótesis es más compatible con los datos: la nula o la alternativa. Esto supone tomar una decisión: bien rechazamos la hipótesis nula o bien no rechazamos la hipótesis nula.*

*¿Y cómo decidimos? El procedimiento que se utiliza es tomar una función de la muestra aleatoria,  $T(X_1, \dots, X_n)$ , y en función de los valores que toma decidir. En este contraste, el estadístico habitualmente utilizado es el siguiente*

$$T = \frac{\bar{X}_n - 1500}{S/\sqrt{n}}. \quad (24.1)$$

*Si consideramos los valores observados tenemos*

$$t_0 = \frac{\bar{x}_n - 1500}{s/\sqrt{n}}, \quad (24.2)$$

*que toma el siguiente valor*

---

<sup>2</sup>En algunos textos se denotan las hipótesis nula y alternativa como  $H_0$  y  $H_a$  respectivamente.

```
(t0 = (mean(x) - 1500) / (sd(x)/sqrt(n)))
## [1] 3.337196
```

*Si observamos la definición de  $T$  parece razonable definir como regla de decisión: rechazamos la hipótesis nula si*

$$T \geq c$$

*donde  $c$  es un valor que elegiremos adecuadamente. Si asumimos que la media poblacional es  $\mu = 1500$ , es decir, el valor límite entre ambas hipótesis entonces se verifica que  $T$  (definido en 24.1) sigue una distribución  $t$  de Student con  $n-1$  grados de libertad, es decir,*

$$T = \frac{\bar{X}_n - 1500}{S/\sqrt{n}} \sim t_{n-1}. \quad (24.3)$$

*Supongamos que no queremos equivocarnos rechazando la hipótesis nula cuando es cierta más que un 5 % de las ocasiones. Al error que cometemos cuando hacemos esto se le llama error tipo I. Esto supone que, colocándonos en la situación límite entre las hipótesis nula y alternativa, hemos de elegir el valor de la constante  $c$  de modo que*

$$P(T \geq c) = 0.05, \quad (24.4)$$

*o, equivalentemente, que*

$$P(T \leq c) = 1 - 0.05 = 0.95. \quad (24.5)$$

*La constante  $c$  es el percentil o cuantil de orden 0.95 de una distribución  $t$  de Student con  $n-1$  grados de libertad que denotamos  $t_{n-1, 0.95}$ . El valor de  $c$  lo podemos calcular con R con el siguiente código*

```
qt(.95, df=99)
## [1] 1.660391
```

*ya que  $n = 100$ . Ya lo tenemos todo. El valor observado de  $T$  es  $t_0$ .*

```
t0
## [1] 3.337196
```

*Como  $t_0$  es mayor que  $t_{n-1, 0.95}$  rechazamos la hipótesis nula lo que indica que el fabricante no mentía: sus bombillas tienen una duración media superior a 1500.*

*Gráficamente podemos representar lo que acabamos de hacer. En la figura 24.1 representamos la densidad cuando la media vale  $\mu_0 = 1500$ . En línea discontinua más a la izquierda (trazo discontinuo) indicamos la posición del estadístico  $t_0$ .*

```
## pdf
## 2
```

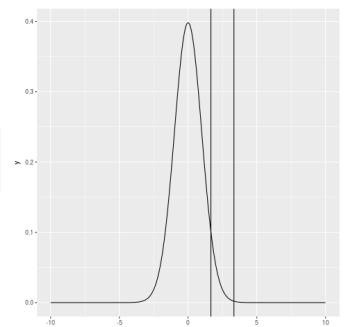


Figura 24.1: Contraste unilateral sobre la media.

Decisión	Realidad	
	$H_0$	$H_1$
Rechazamos $H_0$	Error tipo I	
No rechazamos $H_0$		Error tipo II

Tabla 24.1: Errores que podemos cometer en un problema de contraste de hipótesis.

Y ahora planteamos el problema de un modo genérico. Consideramos el contraste de hipótesis.

$$\begin{aligned} H_0 : \mu &\leq \mu_0, \\ H_1 : \mu &> \mu_0. \end{aligned}$$

Hemos de tomar una entre dos posibles decisiones: *rechazamos la hipótesis nula* o bien *no rechazamos la hipótesis nula*. Un problema de contraste de hipótesis consiste en tomar una decisión. Cuando decidimos nos equivocamos.<sup>3</sup> De hecho, nos podemos equivocar de dos modos. El primer tipo de error consiste en rechazar la hipótesis nula cuando realmente es cierta, el error tipo I. Es el error al que tradicionalmente se le ha prestado más atención. Se considera el *error a controlar*. Por ello, se le pone una cota, un valor máximo. Esta cota es el *nivel de significación  $\alpha$* . El valor más habitual para  $\alpha$  es 0.05. Otros valores que habitualmente se utilizan son 0.01 o 0.1. El otro posible error consiste en no rechazar la hipótesis nula cuando realmente no es cierta. Es el error tipo II. Como todo error ha de ser pequeño. Este tipo de error lo podemos controlar utilizando muestras grandes. Fijamos un  $\alpha$  y con más muestra tendremos un menor error tipo II.

El procedimiento para realizar un contraste de hipótesis es el siguiente:

1. Planteamos el contraste de hipótesis.
2. Fijamos un nivel de significación  $\alpha$ .
3. Tomamos los datos.
4. Evaluamos el valor del estadístico.
5. Si el estadístico está en la región crítica rechazamos la hipótesis nula. En otro caso, no rechazamos dicha hipótesis nula.

Suponemos fijado  $\alpha$ . Calculamos el estadístico.

$$T = \frac{\bar{X}_n - \mu_0}{S/\sqrt{n}} \quad (24.6)$$

y el valor observado

$$t_0 = \frac{\bar{x}_n - \mu_0}{s/\sqrt{n}}. \quad (24.7)$$

Bajo la hipótesis de que la media poblacional  $\mu$  vale  $\mu_0$ ,  $\mu = \mu_0$ , se tiene que

$$T = \frac{\bar{X}_n - \mu_0}{S/\sqrt{n}} \sim t_{n-1}, \quad (24.8)$$

---

<sup>3</sup> Algunos sostienen que lo mejor en la vida es no tomar decisiones. Que las cosas pasen y asumirlas.

y  $t_0$  sería un valor observado de una variable aleatoria con distribución  $t$  con  $n-1$  grados de libertad.

Supongamos que queremos (es una elección del decisor que somos nosotros) un error tipo I que sea menor o igual a  $\alpha$  (habitualmente 0.05, 0.01 o 0.1) entonces la regla de decisión es:

- Rechazamos  $H_0$  si  $T > t_{n-1,1-\alpha}$ , o dicho de otro modo, rechazamos si  $T \in [t_{n-1,1-\alpha}, +\infty)$ .
- En otro caso, no rechazamos  $H_0$ .

Si denotamos  $C = [t_{n-1,1-\alpha}, +\infty)$  entonces estamos rechazando cuando el valor del estadístico está en  $C$ . Rechazamos  $H_0$  si  $T \in C$  y no la rechazamos en otro caso. A este intervalo le llamamos *región crítica*.

Lo que acabamos de hacer se puede hacer de otro modo.

1. Supongamos que calculamos el área a la derecha de  $t_0$  en una  $t$  de Student con  $n - 1$  grados de libertad. Este valor lo vamos a denotar con la letra  $p$  y recibe el nombre de *p-valor*. Es decir, el p-valor viene dado por

$$p = P(T \geq t_0) \text{ donde } T \sim t_{n-1}.$$

El p-valor lo podemos calcular con el siguiente código.

```
(pvalor=1-pt(t0,df=n-1))
## [1] 0.0005965143
```

2. Las dos afirmaciones siguientes son equivalentes:

- (a) Se verifica que  $t_0 > t_{n-1,1-\alpha}$  y por lo tanto rechazamos la hipótesis nula.
- (b) El área a la derecha de  $t_0$  sea menor que  $\alpha$ .

Es equivalente que a que el área a la derecha de  $t_0$  sea menor que el área a la derecha de  $t_{n-1,1-\alpha}$ . El área a la derecha de  $t_0$  es el valor de  $p$  o el p-valor mientras que el área a la derecha de  $t_{n-1,1-\alpha}$  es  $\alpha$ . Por tanto, la misma regla de decisión que estamos empleando se puede formular como: *rechazamos la hipótesis nula  $H_0$  si  $p < \alpha$  y no rechazamos si  $p \geq \alpha$* . En la figura 24.2 el p-valor es el área de la zona en color negro.

```
## pdf
## 2
```

**Ejemplo 24.2 (Función t.test para contrastar)** Vamos a realizar el contraste utilizando la función t.test.

```
t.test(x,mu=1500,alternative="greater")
##
##  One Sample t-test
##
## data: x
## t = 3.3372, df = 99, p-value = 0.0005965
```

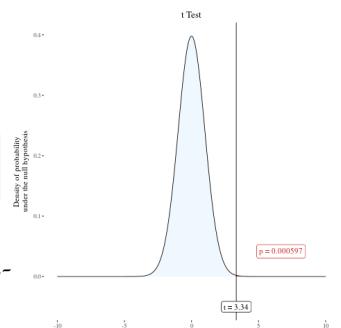


Figura 24.2: El p-valor corresponde con el área sombreada.

```
## alternative hypothesis: true mean is greater than 1500
## 95 percent confidence interval:
## 1548.743      Inf
## sample estimates:
## mean of x
## 1597.009
```

Vemos cómo la función no nos da el valor a partir del cual rechazamos. Simplemente nos indica el *p*-valor y la hipótesis alternativa. ¿Qué hipótesis nula tenemos? La hipótesis nula la obtenemos por negación de la alternativa que nos da la salida. Y la decisión la hemos de tomar nosotros que, al fin y al cabo, somos el decisor. El nivel de significación  $\alpha$  lo hemos elegido a priori. Si trabajamos con un nivel de significación  $\alpha = 0.05$  entonces podemos ver en la salida anterior que dicho *p*-valor es menor que 0.05 y rechazamos la hipótesis nula. Ahí se acaba la historia. Un contraste de hipótesis se acaba cuando se ha tomado una decisión.

**Ejemplo 24.3** Utilizamos los datos tamidata::gse21942. Son datos de expresión con microarrays en los que se compara la expresión entre enfermos de esclerosis múltiple y sanos. Consideramos el gen con identificador Affymetrix y cuyo perfil de expresión aparece en la fila 54651 de la matriz de expresión.

```
data(gse21942, package="tamidata")
y = exprs(gse21942)[54651,]
x = pData(gse21942)[, "FactorValue..DISEASE.STATE."]
```

Nos fijamos en los individuos con la enfermedad (14 primeras muestras).

```
y0 = y[1:14]
```

¿Su media es significativamente menor a 6?

$$\begin{aligned} H_0 : \mu &\geq 6, \\ H_1 : \mu &< 6. \end{aligned}$$

El contraste se puede hacer con dos procedimientos equivalentes.

- Primer procedimiento**
1. En primer lugar elegimos el nivel de significación  $\alpha$ .
  2. Calculamos el estadístico  $T = t_0$  definido en ecuación 24.6.
  3. Rechazamos la hipótesis nula si  $t_0 < t_{n-1,\alpha}$  y no rechazamos en otro caso.

- Segundo procedimiento**
1. Elegimos el nivel de significación  $\alpha$ .
  2. Calculamos el estadístico  $T = t_0$  definido en ecuación 24.6.
  3. Calculamos el área a la izquierda de  $t_0$  en una *t* de Student con  $n-1$  grados de libertad. Este es el valor *p* o *p*-valor.
  4. Rechazamos la hipótesis nula si  $p < \alpha$  y no rechazamos en otro caso.

Hagámoslo con R. Tenemos  $n = 14$ . Tomamos  $\alpha = 0.05$ . El estadístico vale

```
n = 14
mu0 = 7
(t0 = (mean(y0) - mu0) / (sd(y0)/sqrt(n)))

## [1] -3.204991
```

Determinamos el punto a partir del cual rechazamos:  $t_{n-1,\alpha}$

```
alpha = 0.05
qt(alpha,df=n-1)

## [1] -1.770933
```

Como  $t_0 < t_{n-1,\alpha}$  entonces rechazamos la hipótesis nula.  
El segundo procedimiento es más simple de aplicar y es el que usaremos.

```
t.test(y0,mu=mu0,alternative="less")

##
##  One Sample t-test
##
## data: y0
## t = -3.205, df = 13, p-value = 0.00345
## alternative hypothesis: true mean is less than 7
## 95 percent confidence interval:
##       -Inf 6.069111
## sample estimates:
## mean of x
## 4.919546
```

Como el  $p$ -valor es notablemente menor que 0.05 rechazamos la hipótesis nula.

### 24.2.2 Y, finalmente, el contraste bilateral

Suponemos que queremos valorar si la media de la variable de interés podemos considerar que toma un valor próximo a  $\mu_0$ . ¿Cómo formulamos el contraste? El contraste lo formularíamos como:

$$\begin{aligned} H_0 : \mu &= \mu_0, \\ H_1 : \mu &\neq \mu_0. \end{aligned}$$

El contraste se puede hacer con dos procedimientos equivalentes.

- Primer procedimiento**
1. En primer lugar elegimos el nivel de significación  $\alpha$ .
  2. Calculamos el estadístico  $T = t_0$  (definido en ecuación 24.6).
  3. Rechazamos la hipótesis nula si  $t_0 < 1 - t_{n-1,1-\alpha/2}$  o bien si  $t_0 > t_{n-1,1-\alpha/2}$  y no rechazamos en otro caso.

- Segundo procedimiento**
1. Elegimos el nivel de significación  $\alpha$ .
  2. Calculamos el estadístico  $T = t_0$  (definido en ecuación 24.6).

3. Calculamos el área a la izquierda de  $-|t_0|$  más a la derecha de  $|t_0|$  en una  $t$  de Student con  $n-1$  grados de libertad. Este es el valor  $p$  o **p-valor**.
4. Rechazamos la hipótesis nula si  $p < \alpha$  y no rechazamos en otro caso.

**Ejemplo 24.4** Seguimos con los datos del ejemplo 24.3. Nos planteamos saber Supongamos que nos planteamos contrastar si estamos alrededor de una concentración de 5. El contraste es pues el siguiente:

$$\begin{aligned} H_0 : \mu &= 5, \\ H_1 : \mu &\neq 5. \end{aligned}$$

```
t.test(y0,alternative="two.sided",mu=5)

##
##  One Sample t-test
##
## data: y0
## t = -0.12394, df = 13, p-value = 0.9033
## alternative hypothesis: true mean is not equal to 5
## 95 percent confidence interval:
##  3.517188 6.321905
## sample estimates:
## mean of x
## 4.919546
```

Con el  $p$ -valor 0.9033 no podemos rechazar la hipótesis nula a ninguno de los niveles de significación habituales de 0.01 o 0.05 o 0.1. ¿Y de 7? El contraste es ahora:

$$\begin{aligned} H_0 : \mu &= 7, \\ H_1 : \mu &\neq 7. \end{aligned}$$

Realizamos el contrato.

```
t.test(y0,alternative="two.sided",mu=7)

##
##  One Sample t-test
##
## data: y0
## t = -3.205, df = 13, p-value = 0.006901
## alternative hypothesis: true mean is not equal to 7
## 95 percent confidence interval:
##  3.517188 6.321905
## sample estimates:
## mean of x
## 4.919546
```

El  $p$ -valor 0.0069 es menor que el nivel de significación y por lo tanto rechazamos la hipótesis nula con un nivel de significación de 0.05. Notemos que el argumento `alternative` toma el valor `two.sided` por defecto. @

## 24.3 Intervalo de confianza y contraste de hipótesis

Hemos estudiado cómo estimar, mediante un intervalo de confianza, la media de una variable normal. El intervalo de confianza *con un nivel de confianza*  $1 - \alpha$  viene dado por  $[\bar{x}_n - t_{n-1,1-\alpha/2}s/\sqrt{n}, \bar{x}_n + t_{n-1,1-\alpha/2}s/\sqrt{n}]$ . Supongamos que tomamos  $\mu_0$  en este intervalo. Nos planteamos el siguiente contraste:

$$\begin{aligned} H_0 : \quad & \mu = \mu_0, \\ H_1 : \quad & \mu \neq \mu_0 \end{aligned}$$

y pretendemos contrastar estas hipótesis *con un nivel de significación*  $\alpha$ . Si calculamos el valor del estadístico  $t_0 = (\bar{x}_n - \mu_0)/(s/\sqrt{n})$  tenemos que se verifica que  $|t_0| \leq t_{n-1,1-\alpha/2}$  y por lo tanto no rechazamos la hipótesis nula consistente en que la verdadera media de la población  $\mu$  toma el valor  $\mu_0$ .

Este resultado lo hemos formulado referido a la media de una población normal. Sin embargo, es un resultado válido en general. Si suponemos que tenemos un parámetro de toda la población (media, varianza, desviación estandar en poblaciones normales o bien la probabilidad de éxito en poblaciones binomiales) que vamos a denotar de un modo genérico como  $\theta$ . Tenemos un intervalo de confianza (con un nivel de confianza  $1 - \alpha$ ) para  $\theta$  que depende de la muestra  $\{x_1, \dots, x_n\}$  y que denotamos por  $[A(x_1, \dots, x_n), B(x_1, \dots, x_n)]$  o simplemente como  $[A, B]$ . Entonces, si  $A \leq \theta_0 \leq B$  y consideramos el contraste bilateral

$$\begin{aligned} H_0 : \quad & \theta = \theta_0, \\ H_1 : \quad & \theta \neq \theta_0 \end{aligned}$$

no rechazaremos la hipótesis nula con un nivel de significación  $\alpha$ . Y viceversa: si consideremos todos los valores  $\theta_0$  para los cuales no rechazamos la hipótesis nula (con un nivel de significación  $\alpha$ ) en el contraste formulado anteriormente tenemos un intervalo de confianza para  $\theta$  con nivel de confianza  $1 - \alpha$ .

**Ejemplo 24.5** Vamos a ilustrar con los datos del ejemplo 24.3. Recuperamos el contraste de si la media sobre los individuos enfermos es 5.

```
t.test(y0, alternative="two.sided", mu=5, conf.level=0.95)

##
##  One Sample t-test
##
## data: y0
## t = -0.12394, df = 13, p-value = 0.9033
## alternative hypothesis: true mean is not equal to 5
## 95 percent confidence interval:
##  3.517188 6.321905
## sample estimates:
## mean of x
## 4.919546
```

*Podemos comprobar que si tomamos como valores para la media  $\mu$  cualquier valor  $\mu_0$  que esté en el intervalo de confianza el p valor que observaremos para el contraste  $H_0 : \mu = \mu_0$  frente a  $H_1 : \mu \neq \mu_0$  será mayor que  $\alpha = 0.05$  y por lo tanto no la rechazamos.*

*Y al revés, si vamos contrastando  $H_0 : \mu = \mu_0$  frente a  $H_1 : \mu \neq \mu_0$  para todos los posibles valores de  $\mu_0$  y nos quedamos con los valores de  $\mu_0$  para los cuales no rechazamos con un nivel de significación  $\alpha$  entonces tenemos un intervalo de confianza con nivel de confianza  $1 - \alpha$ .*

## 24.4 Contraste de normalidad

Hemos visto que, con mucha frecuencia, los procedimientos estadísticos que hemos utilizado (y mucho de lo que sigue) asumen que los datos que estamos manejando pueden considerarse una muestra de una distribución normal. ¿Y esto es así sin más? ¿Hemos de asumirlo y confiar en nuestra suerte? No. Podemos contrastar esta hipótesis. La hipótesis de que los datos que estamos usando siguen una distribución normal es una hipótesis a contrastar. Y estamos en condiciones de poder hacerlo. El contraste lo podemos formular de un modo informal como:

$H_0$ : Tenemos una muestra de una distribución normal.

$H_1$ : No tenemos una muestra de una distribución normal.

Quizás una formulación más formalista del contraste puede ser la siguiente donde  $X$  es el valor aleatorio que estamos observando n veces.

$H_0$ :  $X \sim N(\mu, \sigma^2)$  con  $-\infty < \mu < +\infty$  y  $\sigma^2 > 0$ .

$H_1$ :  $X$  no sigue una distribución normal.

Consideremos unos datos y veamos cómo evaluar si han sido generados según una distribución normal. Realmente vamos a considerar dos conjuntos de datos. Uno de ellos (que denotamos por  $x$ ) sí que sigue una distribución normal mientras que la segunda muestra (que denotamos por  $y$ ) no sigue una distribución normal. Vamos a estudiar la normalidad de estas dos muestras y recordemos que nos ha de salir siempre una valoración afirmativa para  $x$  y negativa para  $y$ .

### 24.4.1 Gráficos para evaluar la normalidad

Parece que lo primero es ver gráficamente cómo son estos datos. En esta sección vemos el uso del dibujo q-q o dibujo cuantil-cuantil.

**Nota 24.1 (Estimando la densidad de puntos)** Una primera opción (bastante natural pero nada aconsejable) para evaluar la normalidad es utilizar un histograma o un estimador kernel de la densidad. Nos dan una idea de cómo se distribuyen los puntos. En las figuras 24.3(a) y 24.3(b) mostramos el histograma y el estimador kernel de la densidad respectivamente para la muestra  $x$ . Las figuras 24.3(c) y 24.3(d) son las análogas para la muestra  $y$ . ¿Qué se espera ver en estas dos figuras si los datos son normales? La mejor respuesta a esta pregunta es otra pregunta: ¿se parecen las figuras a una densidad normal? Yo diría que las figuras 24.3(a) y 24.3(b) correspondientes a la muestra  $x$  sí que recuerdan la forma de la densidad normal mientras que esto no es cierto para las figuras 24.3(c) y 24.3(d).

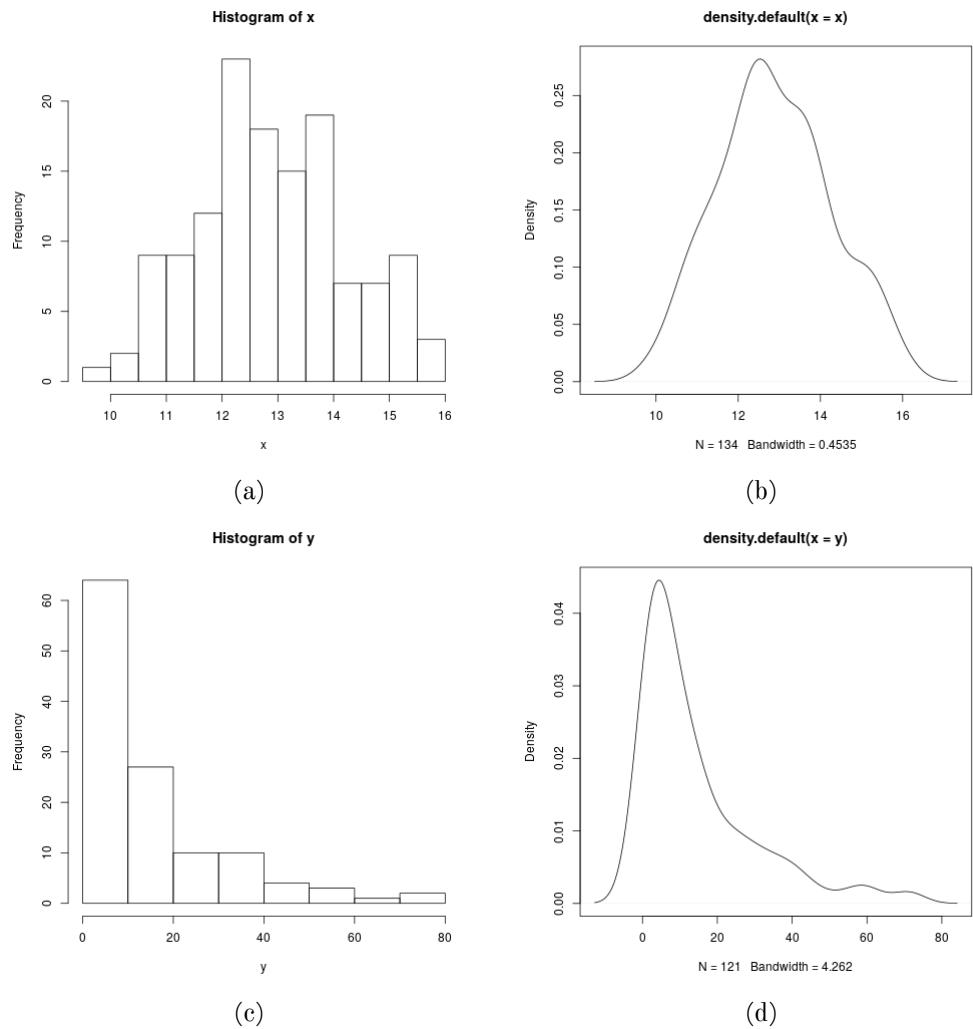


Figura 24.3: Datos x: histograma (a) y estimador kernel de la densidad de x (b). Datos y: histograma (c) y estimador kernel de la densidad de x (d).

No es muy fácil evaluar hasta qué punto es normal la muestra con aproximaciones de la densidad normal que es lo que son el histograma y el estimador kernel de la densidad. De hecho, este tipo de representaciones no son nada aconsejables para este propósito.

Hay una representación gráfica muy popular para este problema concreto: el **dibujo q-q o dibujo cuantil-cuantil**. ¿Qué se pretende evaluar? Si los datos pueden haber sido generados según un modelo normal con la media y varianza que sean. Si  $X$  es la variable que estamos observando ( $n$  veces), pretendemos evaluar hasta qué punto es cierto que

$$X \sim N(\mu, \sigma^2), \quad (24.9)$$

para algún  $\mu$  y algún  $\sigma^2$ . Supongamos que es cierta la afirmación, suponemos cierta que la variable sigue una distribución normal. Elegimos una serie de probabilidades  $p_i$ .<sup>4</sup> En concreto estos valores tienen la forma

$$p_i = \frac{i - \alpha}{n - 2\alpha + 1}, \quad (24.10)$$

donde  $i = 1, \dots, n$ . Dos son los valores de  $\alpha$  que suelen utilizarse<sup>5</sup>

$$\alpha = 0.375, \quad (24.11)$$

o bien

$$\alpha = 0.5. \quad (24.12)$$

Una vez hemos elegido estos valores  $p_i$  hemos de determinar los valores de la abscisa y la ordenada del  $i$ -ésimo punto. Si  $x_i$  con  $i = 1, \dots, n$  son los datos entonces los ordenamos obteniendo los estadísticos ordenados  $x_{(i)}$  que verifican

$$x_{(1)} \leq \dots \leq x_{(n)}.$$

Notemos que el  $i$ -ésimo estadístico ordenado  $x_{(i)}$  es aproximadamente el percentil de orden  $p_i$ . Ahora consideramos una distribución normal estándar y consideramos el valor  $q_i$  tal que si  $Z$  es una variable aleatoria con distribución normal estándar entonces

$$p_i = P(Z \leq q_i) = \int_{-\infty}^{q_i} \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx.$$

De hecho, es habitual denotar

$$\Phi(y) = \int_{-\infty}^y \frac{1}{\sqrt{2\pi}} e^{-\frac{x^2}{2}} dx.$$

Entonces

$$p_i = \Phi(q_i),$$

es decir,

$$q_i = \Phi^{-1}(p_i).$$

En el dibujo q-q representamos los puntos  $(q_i, x_{(i)})$  con  $i = 1, \dots, n$ .

La nota 24.6 nos muestra cómo realizar el dibujo de un modo simple utilizando las funciones `qqnorm` y `qqline`.

---

<sup>4</sup> Una explicación detallada de cómo elegir estos valores lo podemos encontrar en [o](#) en [Thode2002]

<sup>5</sup> La función `ppoints` nos indica los valores que realmente se utilizan.

**Ejemplo 24.6 (Dibujo q-q con las funciones *qnorm* y *qqline*)** La función *qqnorm* nos proporciona el dibujo q-q fácilmente. En la figura 24.4(a) tenemos la representación gráfica para la muestra  $x$ .

`qqnorm(x)`

La figura 24.4(c) muestra el mismo dibujo para la segunda muestra. ¿Podemos considerar que los puntos de la figura 24.4(a) están sobre una línea recta, están alineados? Yo diría que sí. ¿Y los puntos de la figura 24.4(c)? Creo que coincidiríamos que no.

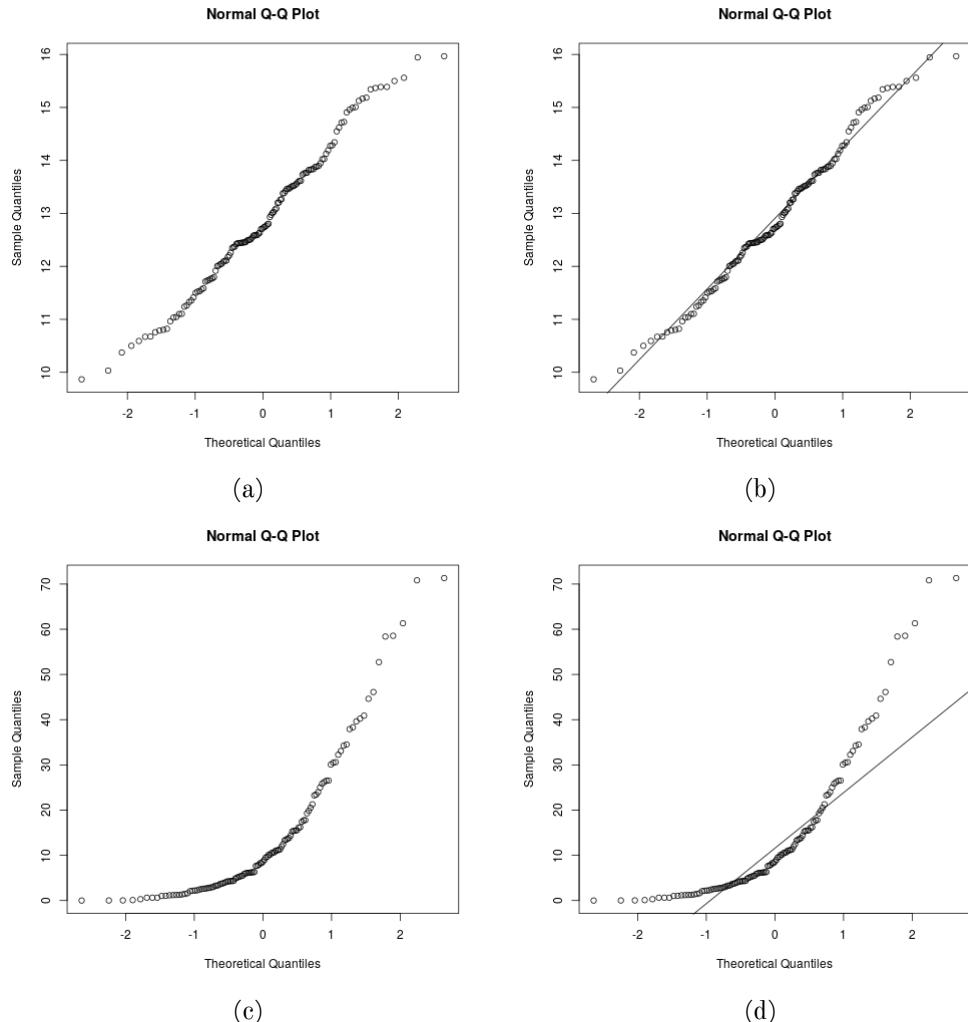


Figura 24.4: (a) Dibujo q-q o cuantil-cuantil para datos  $x$ . (b) Dibujo q-q o cuantil-cuantil para la muestra  $x$  añadiendo la línea que pasa por el primer y tercer cuartil. Vemos cómo los puntos están muy próximos a la línea. No podemos rechazar la normalidad de los datos utilizando este dibujo. (c) Dibujo q-q o cuantil-cuantil para datos  $y$ . (d) Dibujo q-q o cuantil-cuantil para la muestra  $y$  añadiendo la línea que pasa por el primer y tercer cuartil. Los puntos están alejados de la línea. Parece razonable rechazar la normalidad de los datos utilizando este gráfico.

*En las dos figuras estamos intentando ver si podemos superponer una línea recta a los puntos que estamos representando. No viene mal una ayuda visual que nos coloque una buena línea y sobre ella veamos si los puntos están cercanos a la línea. Se pueden superponer muchas líneas. Existe una práctica a la hora de elegir esta línea. Para tra-*

*zar una línea necesitamos dos puntos. Se eligen dos buenos puntos y consideramos la línea que pasa por ellos. El primero es el punto correspondiente a los percentiles de orden 0.25, esto es, el cuartil inferior. Determinamos este cuartil inferior en los datos observados (y es el valor de la ordenada) y en una distribución normal estándar (y es el valor de la abscisa). El segundo punto es el correspondiente al cuartil superior o percentil de orden 0.75. Su abscisa y ordenada son los percentiles de orden 0.75 en la distribución normal estándar y el observado en los datos. En las figuras 24.4(b) y 24.4(d) tenemos los dibujos q-q añadiendo la línea que pasa por los cuartiles inferior y superior. Para los datos  $x$  se obtiene con el siguiente código.*

```
qqnorm(x)
qqline(x)
```

*¿Están sobre una línea recta los puntos en cada una de las gráficas? Podemos ver que para la figura 24.4(b) correspondiente a la muestra  $x$  los datos parecen bien alineados. Esto no parece tan cierto para los datos de la muestra  $y$  que aparecen en la figura 24.4(d). Rechazaríamos gráficamente la normalidad de la muestra  $y$  mientras que no la rechazaríamos para la muestra  $x$ .*

*En [http://en.wikipedia.org/wiki/Q-Q\\_plot](http://en.wikipedia.org/wiki/Q-Q_plot) se tiene una explicación muy completa de este gráfico.*

## 24.5 Contrastes de normalidad

Un procedimiento gráfico siempre ayuda a descartar situaciones claras. En la sección anterior hemos visto cómo podíamos rechazar la normalidad de los datos y mediante un dibujo q-q. Para los datos  $x$  no hemos podido rechazarla. ¿Nos quedamos con esto? ¿Admitimos que son datos normales y en paz? No. El siguiente paso a dar consiste en utilizar un contraste de hipótesis. Vamos a comentar rápidamente los contrastes más utilizados. En este sección utilizamos el paquete de R *nortest* [65] para los test ji-cuadrado y Kolmogorov-Smirnov. El test de Shapiro-Wilk lo aplicamos con la función *shapiro.test* de [113, stats].

### 24.5.1 Test de Shapiro–Wilk

Es otro test para determinar si podemos considerar unos datos normales. Apliquémoslo a nuestros datos. Para la muestra  $x$

```
shapiro.test(x)

##
##  Shapiro-Wilk normality test
##
## data: x
## W = 0.98668, p-value = 0.2204
```

No rechazamos con un nivel de significación de 0.05. Para la segunda muestra,

```
shapiro.test(y)

##
##  Shapiro-Wilk normality test
##
## data:  y
## W = 0.799992, p-value = 1.516e-11
```

Vemos como rechazamos claramente.

Una buena explicación de este test se puede encontrar en [http://en.wikipedia.org/wiki/Shapiro%20%93Wilk\\_test](http://en.wikipedia.org/wiki/Shapiro%20%93Wilk_test)

### 24.5.2 Test ji-cuadrado

Lo podemos hacer con la función *pearson.test* del paquete [65, nortest]. La aplicamos a ambos conjuntos de datos. Primero a los datos x.

```
library(nortest)
pearson.test(x)

##
##  Pearson chi-square normality test
##
## data:  x
## P = 18.239, p-value = 0.1086
```

Y después a la muestra y.

```
pearson.test(y)

##
##  Pearson chi-square normality test
##
## data:  y
## P = 97.099, p-value = 6.69e-16
```

Vemos cómo rechazamos la normalidad de la muestra y mientras que no la rechazamos para la muestra x a un nivel de significación  $\alpha = 0.05$ .

### 24.5.3 Test de Kolmogorov-Smirnov

Para aplicar este test utilizamos la función *lillie.test* del paquete [65, nortest]. Empezamos con los datos x.

```
lillie.test(x)

##
##  Lilliefors (Kolmogorov-Smirnov) normality
##  test
##
## data:  x
## D = 0.060824, p-value = 0.2606
```

Y ahora para los datos y.

```
lillie.test(y)

##
##  Lilliefors (Kolmogorov-Smirnov) normality
##  test
##
## data:  y
## D = 0.18224, p-value = 1.273e-10
```

Vemos cómo rechazamos la normalidad de la muestra y mientras que no la rechazamos para la muestra x a un nivel de significación  $\alpha = 0.05$ .

# Capítulo 25

## Comparación de dos poblaciones

### 25.1 Introducción

Distintos problemas relativos a comparar dos poblaciones se tratan en este tema. Empezamos abordando el problema de la comparación mediante herramientas puramente descriptivas de las dos muestras de que disponemos, una por población en estudio. Seguimos con la comparación de dos poblaciones normales, en particular, la comparación de sus medias y varianzas. Continuamos con la comparación mediante el test de Kolmogorov-Smirnov para dos muestras. Terminamos con un test de Montecarlo para comparar las medias de dos poblaciones.

### 25.2 Comparación descriptiva de las muestras

Pretendemos comparar dos poblaciones. De cada una de ellas disponemos de una muestra. Hablamos de dos **muestras independientes** porque proceden de poblaciones distintas.

Por ejemplo, podemos tener la expresión de un gen dado observado bajo dos condiciones (salvaje y modificado) y pretendemos compararlas. Para ilustrar tomamos muestras concretas. Tenemos dos muestras x e y, una por población. Lo primero es la comparación descriptiva de estas muestras.

La primera muestra es

```
## [1] 23.3 23.4 26.5 25.8 19.1 22.1 26.3 21.9 21.8  
## [10] 22.5 19.5 23.1 22.5 26.9 22.4 27.3 24.1 27.8  
## [19] 25.1 22.7 24.7 24.6 16.4 23.2 24.8 27.9 19.2  
## [28] 26.4 22.3 23.8 19.7 27.6 22.6 20.3 22.3 21.4  
## [37] 25.2 19.1 22.5 23.4 20.7 21.6 21.1 24.4 22.1
```

mientras que la segunda muestra es

```
round(y,1)  
  
## [1] 29.2 28.3 27.9 28.0 30.5 26.0 32.5 28.4 29.0  
## [10] 31.8 24.7 22.8 26.5 28.8 23.8 29.9 28.8 31.4  
## [19] 30.9 32.0 31.0 33.4 29.0 23.7 33.8 26.6 31.9
```

```
## [28] 30.7 27.3 28.2 36.6 29.8 25.3 31.2 29.6 30.1
## [37] 32.8 28.0 27.9 33.3 26.6 29.3 27.8 30.3 26.3
## [46] 28.5 29.1 29.0 32.9 27.1 32.7 33.4 25.7 28.8
```

¿Qué significa comparar las dos muestras? Desde un punto de vista estadístico las dos muestras son realizaciones de variables aleatorias. Si repetimos el muestreo obtendremos valores distintos en cada una de las dos zonas observadas. Por tanto, la comparación no puede ser comparar los valores numéricos uno a uno. Tampoco puede ser comparar sin más algún resumen de los datos como la media y la varianza muestrales. En nuestro caso, los valores observados son para la primera muestra

```
mean(x)
## [1] 23.14726
sd(x)
## [1] 2.630329
```

y para la segunda

```
mean(y)
## [1] 29.24535
sd(y)
## [1] 2.850981
```

Casi tiene un mayor interés la comparación gráfica de ambas muestras. En la figura 25.1 tenemos el histograma y el estimador kernel de la densidad de las muestras x e y. ¿Podemos considerar que ambas muestras han sido obtenidas de una misma población? O, por el contrario: ¿proceden de distintas poblaciones?

Una primera cuestión es que *pretendemos comparar muestras obtenidas de posiblemente dos poblaciones distintas*. No parece una buena opción utilizar figuras distintas (bien histogramas bien estimadores kernel). Ambos dibujos se configuran para su propia muestra. Por ejemplo, el número de clases de cada histograma depende del número de datos y por lo tanto es distinta. La anchura de banda de los estimadores kernel también son distintos. En resumen, si queremos comparar lo lógico es representar conjuntamente ambas muestras. ¿Cómo? Utilizar histogramas conjuntos no es especialmente útil o de interpretación fácil. Es preferible utilizar colocar en una misma gráfica los dos estimadores kernel. Un detalle, el nivel de suavizado *debe* de ser el mismo para los dos estimadores, o dicho de otro modo, la anchura de banda ha de ser la misma. En la figura 25.2 se muestra una representación conjunta de ambos estimadores kernel de la densidad. Los datos muestran una forma simétrica y vemos que la muestra y toma valores mayores que la muestra x. Sin embargo, es una pura interpretación gráfica. Hemos de estimar las diferencias entre las muestras y contrastar si ambas muestras proceden de distintas poblaciones con contrastes de hipótesis formales. Es lo que vamos a hacer.

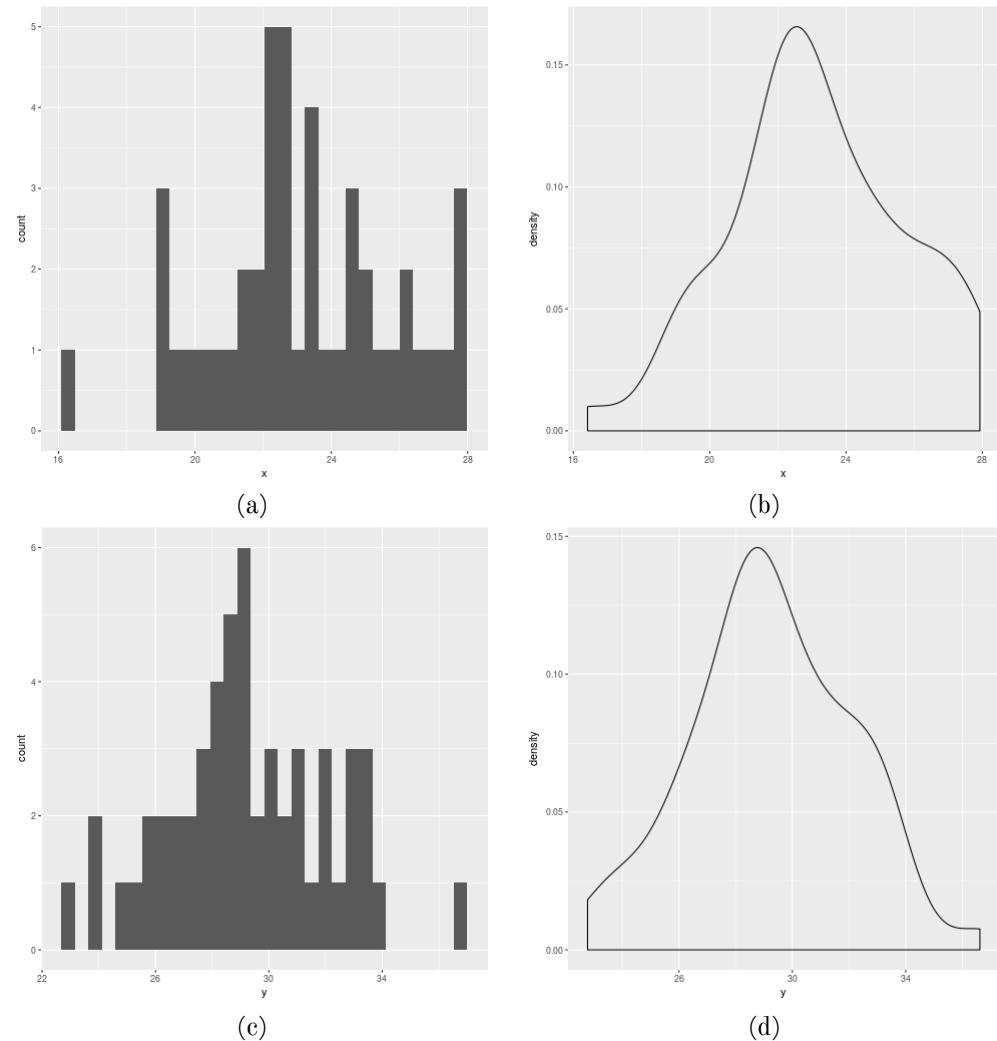


Figura 25.1: Datos  $x$ : histograma (a) y estimador kernel de la densidad de  $x$  (b). Datos  $y$ : histograma (c) y estimador kernel de la densidad de  $x$  (d).

```

z = data.frame(u = c(x,y), v=factor(rep(c(1,2),c(length(x),length(y))), 
                                         levels=1:2,labels=c("x","y")))
png(paste0(dirTamiFigures,"CompDosPob7.png"))
ggplot(z,aes(x=u,group=v,colour=v))+geom_density()
dev.off()

## pdf
## 2

```

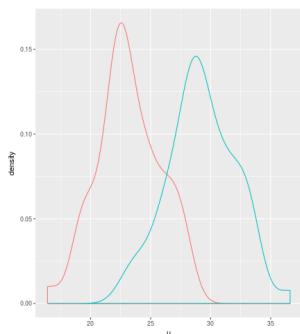


Figura 25.2: Estimadores kernel de la densidad de  $x$  e  $y$ . Los datos de la muestra  $y$  tienden a tomar valores mayores que los de  $x$ .

Cuando comparamos dos poblaciones hay que hacerse varias preguntas y en función de las respuestas que obtengamos planteamos la comparación. ¿Son los datos normales? Es la primera pregunta que debemos responder: ¿Podemos considerar que los datos proceden de dos poblaciones normales? De otro modo: ¿la primera muestra procede de una población normal con media y varianza  $\mu_X$  y  $\sigma_X^2$  desconocidos y la segunda muestra procede de una población normal con media y varianza  $\mu_Y$  y  $\sigma_Y^2$  desconocidos? Obviamente esto supone pasar test de normalidad a nuestros datos. Cada una de las dos muestras ha de pasar el test de normalidad. Supongamos que la respuesta es afirmativa. El problema de comparar las poblaciones se simplifica. ¿Cómo es la densidad de una normal? Si la variable aleatoria  $X \sim N(\mu_X, \sigma_Y^2)$  entonces su función de densidad es

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma_X} e^{-\frac{1}{2}\frac{(x-\mu_X)^2}{\sigma_X^2}} \quad (25.1)$$

La de la variable  $Y$  con distribución  $Y \sim N(\mu_Y, \sigma_Y^2)$  tiene la misma expresión en donde cambiamos  $\mu_X$  por  $\mu_Y$  y  $\sigma_X$  por  $\sigma_Y$ . En resumen si las medias son la misma,  $\mu_X = \mu_Y$  y las varianzas son iguales  $\sigma_X^2 = \sigma_Y^2$  entonces las densidades son la misma. Tenemos la misma población normal. En resumen, si asumimos que las dos poblaciones son normales entonces comparar las poblaciones se reduce a comparar las medias y las varianzas. Además cuando no sean iguales podremos saber a qué se debe. Las situaciones en que nos podemos encontrar son las siguientes:

1. La misma media y varianza.
2. Distinta media y la misma varianza. En la figura ?? mostramos dos densidades normales verificándolo.
3. La misma media y distinta varianza. En la figura ?? vemos las funciones de densidad.
4. Distinta media y varianza. En la figura ?? tenemos un ejemplo de densidades normales verificando esto.

En la situación 1 de la enumeración anterior *no* tenemos dos poblaciones. Tenemos una sola población. Sin embargo, en los casos 2, 3 y 4 tenemos dos poblaciones distintas bien porque la media, la varianza o ambas son distintas. Podemos evaluar (contrastar) si la diferencia entre las poblaciones se da en la variabilidad (en las varianzas) en las medias o en ambas cosas. Esto nos da una visión clara de las diferencias entre las poblaciones.

En la sección § 25.4.2 planteamos el correspondiente contraste de hipótesis en donde comparamos las varianzas de dos poblaciones normales. Obviamente si rechazamos la hipótesis de igualdad de las varianzas tenemos dos poblaciones distintas ya que sus varianzas lo son. Lo que

no sabemos si sus medias son o no iguales. Esto va después. Si no hemos rechazado que las varianzas sean iguales entonces comparamos las medias asumiendo una misma varianza en las dos poblaciones, es decir, asumimos que la primera muestra aleatoria es de una población normal con media  $\mu_X$  y varianza  $\sigma^2$  mientras que la segunda muestra es de una normal con media  $\mu_Y$  y varianza  $\sigma^2$ . Finalmente, también podemos realizar el contraste de hipótesis para comparar las medias cuando las varianzas son distintas.

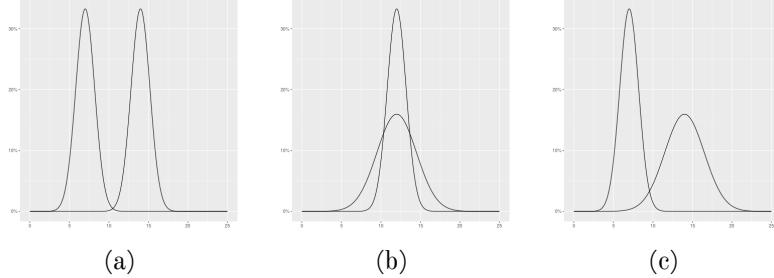


Figura 25.3: Densidades normales: (a) distinta media y la misma varianza; (b) misma media y distinta varianza; (c) distintas medias y varianzas.

## 25.3 Comparando las medias de dos poblaciones normales

Asumimos, en esta sección, que las muestras son de poblaciones normales. Desde el punto de vista del usuario el interés suele estar en comparar medias independientemente de si las varianzas son iguales o no. Esto es una visión algo miope porque la diferencia de la varianza también indica que tenemos poblaciones distintas.

Tenemos una muestra  $X_1, \dots, X_n$  de una variable  $X \sim N(\mu_X, \sigma_X^2)$  y otra muestra  $Y_1, \dots, Y_m$  de  $Y \sim N(\mu_Y, \sigma_Y^2)$ .

### 25.3.1 Estimación de la diferencia de medias

Nos interesa estudiar si  $\mu_X$  y  $\mu_Y$  son iguales o no. Esto se puede formular de varias formas. En primer lugar podemos plantearnos estimar la diferencia de los dos valores. El estimador de esta cantidad es

$$\widehat{\mu_X - \mu_Y} = \hat{\mu}_X - \hat{\mu}_Y = \bar{X}_n - \bar{Y}_m.$$

La expresión anterior significa que el estimador de la diferencia  $\mu_X - \mu_Y$ , que denotamos como  $\widehat{\mu_X - \mu_Y}$ , tomamos la diferencia de los estimadores de  $\mu_X$  y  $\mu_Y$  que son respectivamente  $\bar{X}_n$  y  $\bar{Y}_m$ . El valor a estimar  $\mu_X - \mu_Y$  es un número desconocido, lo que llamamos parámetro, que estimamos mediante el estimador puntual  $\bar{X}_n - \bar{Y}_m$ .

Hemos visto antes como construir intervalos de confianza para cada una de las medias  $\mu_X$  y  $\mu_Y$ . Estos intervalos (asumiendo normalidad) tienen la expresión  $\bar{x}_n \pm t_{n-1, 1-\alpha/2} s_X / \sqrt{n}$  y  $\bar{y}_m \pm t_{m-1, 1-\alpha/2} s_Y / \sqrt{m}$ .

**Ejemplo 25.1** Supongamos que tomamos un nivel de confianza  $1-\alpha = 0.95$ . Con los datos que tenemos los intervalos son, para la primera muestra,

```
t.test(x)$conf.int
## [1] 22.35703 23.93750
## attr(,"conf.level")
## [1] 0.95
```

y para la segunda

```
t.test(y)$conf.int
## [1] 28.46718 30.02352
## attr(,"conf.level")
## [1] 0.95
```

El primer intervalo contiene a  $\mu_X$  con un nivel de confianza  $1 - \alpha$  y el segundo a  $\mu_Y$  con el mismo nivel de confianza. *Pero esto ya lo sabía y no me resuelve nada.* Queremos un intervalo para la diferencia de medias  $\mu_X - \mu_Y$  con un nivel de confianza previamente especificado  $1 - \alpha$ . Para ello se utiliza la siguiente cantidad.

$$T = \frac{(\bar{X}_n - \bar{Y}_m) - (\mu_X - \mu_Y)}{SE(\bar{X}_n - \bar{Y}_m)} \quad (25.2)$$

donde  $SE(\bar{X}_n - \bar{Y}_m)$  denota la desviación estándar de la variable  $\bar{X}_n - \bar{Y}_m$ . Esta cantidad no tiene una expresión única. De hecho, depende de si las varianzas son iguales o distintas. Además el comportamiento aleatorio de  $T$  definido en la ecuación 25.2 también depende de si son la misma varianza o son distintas. En cualquier caso  $T$  va a tener una distribución de probabilidad que es una  $t$  de Student donde serán distintos los grados de libertad. Si de momento denotamos como  $\nu$  estos grados (tomará dos valores posibles) entonces el intervalo de confianza para  $\mu_X - \mu_Y$  con un nivel de confianza  $1 - \alpha$  tendrá la expresión general

$$\bar{X}_n - \bar{Y}_m \pm t_{\nu, 1-\alpha/2} \times SE(\bar{X}_n - \bar{Y}_m).$$

Como vemos no es muy distinto al caso de una sola muestra en que estimamos una sola media.

**Ejemplo 25.2** En nuestro caso, y antes de ver las expresiones, podemos obtener el intervalo de confianza asumiendo varianzas iguales con

```
t.test(x,y,var.equal=TRUE)$conf.int
## [1] -7.200980 -4.995192
## attr(,"conf.level")
## [1] 0.95
```

y asumiendo que son distintas (que es la opción por defecto) con

```
t.test(x,y,var.equal=FALSE)$conf.int
## [1] -7.193018 -5.003155
## attr(,"conf.level")
## [1] 0.95
```

Como vemos son bastante parecidos los dos intervalos. Posiblemente no deben de ser muy distintas las varianzas de las dos poblaciones.

Si asumimos que las dos varianzas son iguales, esto es, asumimos la hipótesis de que  $\sigma_X^2 = \sigma_Y^2$ , denotaremos por  $\sigma^2$  el valor común:  $\sigma^2 = \sigma_X^2 = \sigma_Y^2$ . El valor común  $\sigma^2$  de la varianza se puede estimar con

$$S_p^2 = \frac{(n-1)S_X^2 + (m-1)S_Y^2}{n+m-2}.$$

De hecho, lo que tenemos es que  $SE(\bar{X}_n - \bar{Y}_m) = S_p \sqrt{\frac{1}{n} + \frac{1}{m}}$  y

$$T = \frac{\bar{X} - \bar{Y} - (\mu_X - \mu_Y)}{S_p \sqrt{\frac{1}{n} + \frac{1}{m}}} \sim t_{n+m-2}, \quad (25.3)$$

tiene una distribución t de Student con  $n+m-2$  grados de libertad. El intervalo de confianza lo podemos construir utilizando el resultado dado en 25.3 y vendría dado por

$$\bar{X}_n - \bar{Y}_m \pm t_{n+m-2, 1-\alpha/2} \frac{S_p}{\sqrt{n+m-2}}.$$

**Ejemplo 25.3** Este intervalo viene dado, para un nivel de  $1-\alpha = 0.95$  por

```
t.test(x,y,var.equal=TRUE,conf.level=0.95)$conf.int
## [1] -7.200980 -4.995192
## attr(,"conf.level")
## [1] 0.95
```

y, para un nivel de  $1-\alpha = 0.99$  por

```
t.test(x,y,var.equal=TRUE,conf.level=0.99)$conf.int
## [1] -7.558148 -4.638024
## attr(,"conf.level")
## [1] 0.99
```

Como vemos un mayor nivel de confianza supone un intervalo más grande. Tenemos más confianza en que el valor verdadero de  $\mu_X - \mu_Y$  esté en el intervalo pero el intervalo al ser mayor nos estima la diferencia de medias con una precisión menor.

Y ahora consideremos el caso con varianzas posiblemente distintas, esto es, suponemos que  $\sigma_X^2 \neq \sigma_Y^2$ . No tiene ahora sentido estimar una varianza común que no existe. Estimamos cada una de ellas con su estimador natural:  $\sigma_X^2$  con  $S_X^2$  y  $\sigma_Y^2$  con  $S_Y^2$ . El error estándar de  $\bar{X}_n - \bar{Y}_m$ ,  $SE(\bar{X}_n - \bar{Y}_m)$ , viene dado por  $SE(\bar{X}_n - \bar{Y}_m) = \sqrt{\frac{S_X^2}{n} + \frac{S_Y^2}{m}}$  y la expresión que adopta 25.2 y su distribución **aproximada** es

$$T = \frac{\bar{X}_n - \bar{Y}_m - (\mu_X - \mu_Y)}{\sqrt{\frac{S_X^2}{n} + \frac{S_Y^2}{m}}} \sim t_{\nu_0} \quad (25.4)$$

con

$$\nu_0 = \frac{\left(\frac{S_X^2}{n} + \frac{S_Y^2}{m}\right)}{\frac{(S_X^2/n)^2}{n-1} + \frac{(S_Y^2/m)^2}{m-1}}.$$

Es decir, que  $T$  se distribuye **aproximadamente** como una distribución t de Student  $\nu_0$  grados de libertad donde  $\nu_0$  no tiene porqué ser un número entero. El intervalo de confianza con nivel de confianza  $1 - \alpha$  viene dado por

$$\bar{X}_n - \bar{Y}_m \pm t_{\nu_0, 1-\alpha/2} \sqrt{\frac{S_X^2}{n} + \frac{S_y^2}{m}}.$$

**Ejemplo 25.4** Con los datos que estamos analizando los intervalos para la diferencia de medias con niveles de confianza 0.95 y 0.99 son

```
t.test(x,y,var.equal=FALSE,conf.level=0.95)$conf.int
```

```
## [1] -7.193018 -5.003155
## attr(,"conf.level")
## [1] 0.95
```

*y*

```
t.test(x,y,var.equal=FALSE,conf.level=0.99)$conf.int
```

```
## [1] -7.547722 -4.648450
## attr(,"conf.level")
## [1] 0.99
```

### 25.3.2 Contraste de hipótesis

Otra manera de comparar las medias de dos poblaciones es contrastar si sus medias son iguales o bien si alguna de ellas es mayor que la otra. Vamos a considerar el contraste de igualdad de medias frente a la desigualdad lo que también se llama contraste bilateral, bidireccional o de dos colas.

$$\begin{aligned} H_0 : \quad & \mu_X = \mu_Y, \\ H_1 : \quad & \mu_X \neq \mu_Y. \end{aligned}$$

La región crítica, es decir, los valores para los cuales rechazamos la hipótesis nula es:

$$|T_0| > t_{\nu, 1-\alpha/2}$$

siendo

$$T_0 = \frac{\bar{X} - \bar{Y}}{S_p \sqrt{\frac{1}{n} + \frac{1}{m}}}, \quad (25.5)$$

y  $\nu = n + m - 2$  si las varianzas se asumen iguales. Para varianzas desiguales:

$$T_0 = \frac{\bar{X}_n - \bar{Y}_m}{\sqrt{\frac{S_X^2}{n} + \frac{S_y^2}{m}}} \quad (25.6)$$

y  $\nu = \nu_0$ . El p-valor viene dado por

$$p = P(T_0 \geq |t_0|)$$

siendo  $t_0$  el valor observado de  $T_0$  en cada caso.

**Ejemplo 25.5 (Test de la t para comparar medias)** Supongamos que el nivel de significación elegido es  $\alpha = 0.01$ . Vamos a contrastar la igualdad de medias en los dos casos. Con varianzas iguales tendremos:

```
t.test(x,y,var.equal=TRUE)

##
##  Two Sample t-test
##
## data: x and y
## t = -10.974, df = 97, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -7.200980 -4.995192
## sample estimates:
## mean of x mean of y
## 23.14726 29.24535
```

Vemos que el  $p$ -valor 0 es muy pequeño (menor que  $\alpha$ ) y por ello rechazamos la igualdad de las medias. ¿Qué pasa si no asumimos la igualdad de las varianzas? ¿Cambia el resultado del test? Para varianzas desiguales:

$$T = \frac{\bar{X} - \bar{Y}}{\sqrt{\frac{S_x^2}{n} + \frac{S_y^2}{m}}} \quad (25.7)$$

$y$

$$\nu = \frac{\left(\frac{S_x^2}{n} + \frac{S_y^2}{m}\right)}{\frac{(S_x^2/n)^2}{n-1} + \frac{(S_y^2/m)^2}{m-1}}.$$

El  $p$ -valor viene dado por

$$p = P(|T| \geq t_0)$$

siendo  $t_0$  el valor observado de  $T$  en cada caso.

```
t.test(x,y,var.equal=FALSE)

##
## Welch Two Sample t-test
##
## data: x and y
## t = -11.055, df = 95.965, p-value < 2.2e-16
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -7.193018 -5.003155
## sample estimates:
## mean of x mean of y
## 23.14726 29.24535
```

El  $p$ -valor 0 es muy pequeño por lo que rechazamos la hipótesis nula.

### 25.3.3 Los contrastes unilaterales o direccionales

Los otros dos contrastes serían los correspondientes unilaterales, o direccionales o de una cola.

$$\begin{aligned} H_0 : \mu_X &\leq \mu_Y, \\ H_1 : \mu_X &> \mu_Y. \end{aligned}$$

$$\begin{aligned} H_0 : \mu_X &\geq \mu_Y, \\ H_1 : \mu_X &< \mu_Y. \end{aligned}$$

**Ejemplo 25.6** Vamos a comparar las expresiones medias del gen cuyo perfil aparece en la fila 54651 de la matriz de expresión de tamida: gse21942. En principio vamos a suponer que las varianzas de dichas concentraciones las podemos considerar iguales. Primero leemos los datos.

```
library(Bioconductor)
data(gse21942, package="tamidata")
y = exprs(gse21942)[54651,]
x = pData(gse21942)[, "FactorValue..DISEASE.STATE."]
```

Los intervalos de confianza para la diferencia de medias y el contraste de hipótesis para la igualdad de las medias frente a la alternativa de medias distintas asumiendo que las varianzas son la misma lo podemos obtener con

```
t.test(y~x, var.equal=TRUE)

##
## Two Sample t-test
##
## data: y by x
## t = 6.2573, df = 27, p-value = 1.077e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## 2.675284 5.285788
## sample estimates:
##               mean in group healthy
##                         8.900083
## mean in group multiple sclerosis
##                         4.919546
```

Estimamos la diferencia de medias como 3.981 y el intervalo de confianza con nivel 0.95 es [2.675, 5.286]. Además el p-valor observado es  $1.076984 \times 10^{-6}$  menor que 0.01 por lo que rechazamos la hipótesis nula de igualdad de medias.

Vamos a repetir el estudio sin asumir que la varianza es la misma en las dos poblaciones.

```
t.test(y~x, var.equal=TRUE)
```

```

## 
## Two Sample t-test
## 
## data: y by x
## t = 6.2573, df = 27, p-value = 1.077e-06
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## 2.675284 5.285788
## sample estimates:
##          mean in group healthy
##                  8.900083
## mean in group multiple sclerosis
##                  4.919546

```

*El estimador puntual de la diferencia de medias no cambia, 3.981. Sin embargo, el intervalo de confianza con nivel 0.95 sí que cambia. Ahora es [2.567,5.395]. Además el p-valor observado es  $3 \times 10^{-5}$  sigue siendo menor que 0.01 por lo que rechazamos la hipótesis nula de igualdad de medias. La pregunta obvia es: ¿qué opción elegir? Desde el punto de vista aplicado lo lógico es asumir varianzas desiguales (que por otra parte suele ser el caso). Por ello la opción por defecto de t.test es precisamente asumir varianzas distintas. Además cuando las varianzas son realmente iguales o casi iguales los resultados que se obtienen asumiendo que las varianzas son la misma o sin asumirlo son prácticamente las mismas.*

## 25.4 Inferencia sobre las varianzas de dos poblaciones normales

Hasta ahora no nos hemos preocupado de las varianzas de las poblaciones normales con las que estamos trabajando. Ya es hora de hacerlo. Además es enormemente importante estimar y contrastar cosas sobre las varianzas. Tenemos dos muestras correspondientes a dos poblaciones. Tenemos, como siempre, un doble interés: estimar y contrastar.

### 25.4.1 Estimación del cociente de varianzas

**Ejemplo 25.7** *Vamos a generar unos datos con distribución normal de los cuales sabemos sus medias y varianzas.*

```

n = 45
m = 54
x = rnorm(n,mean=23,sd=2.45)
y = rnorm(m,mean=30,sd=3.45)

```

*Si llamamos a la función var.test tenemos lo siguiente:*

*Si miramos la última línea vemos que no nos está estimando cada varianza separadamente. En lugar de ello, como nos interesa compararlas, lo que estimamos es el cociente  $\sigma_X^2/\sigma_Y^2$ .*

Las varianzas de las dos poblaciones se comparan utilizando el cociente  $\sigma_X^2/\sigma_Y^2$ . Podemos estimarlo o bien contrastar hipótesis sobre el cociente.

El estimador puntual de esta cantidad es

$$\frac{S_X^2}{S_Y^2} \quad (25.8)$$

que con nuestros datos vale 0.711 indicando que la primera varianza es menor que la segunda. Bien, nos hemos centrado en la estimación de  $\sigma_X^2/\sigma_Y^2$ . Siempre que estimamos damos una estimación puntual (que acabamos de ver en 25.8) y un intervalo de confianza. Para obtenerlo utilizamos la cantidad pivotal siguiente:

$$\frac{S_X^2/\sigma_X^2}{S_Y^2/\sigma_Y^2} \sim F(n-1, m-1) \quad (25.9)$$

que tiene una distribución  $F$  (de Fisher) con  $n-1$  y  $m-1$  grados de libertad.

Los tamaños muestrales de nuestros datos son  $n=45$  y  $m=54$ . La función de densidad de  $\frac{S_X^2/\sigma_X^2}{S_Y^2/\sigma_Y^2}$  aparece en la figura 25.4.

Denotemos por  $F_p(n-1, m-1)$  el percentil de orden  $p$  de la distribución  $F(n-1, m-1)$ , es decir, el punto que a su izquierda tiene un área  $p$ . Por ejemplo, si tomamos los valores de  $n$  y  $m$  anteriores y  $p = 0.975$  entonces el percentil viene dado como

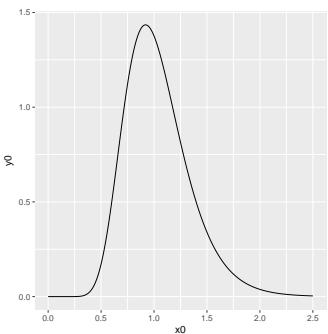


Figura 25.4

```
qf(0.975, df1=n-1, df2=m-1)
```

```
## [1] 1.75846
```

Teniendo en cuenta el resultado 25.9 el intervalo de confianza para  $\frac{S_X^2/\sigma_X^2}{S_Y^2/\sigma_Y^2}$  es el siguiente:

$$\left[ \frac{S_Y^2}{S_X^2} \frac{1}{F_{1-\alpha/2}(n-1, m-1)}, \frac{S_Y^2}{S_X^2} \frac{1}{F_{\alpha/2}(n-1, m-1)} \right]$$

### 25.4.2 Contraste de hipótesis para el cociente de varianzas

Nuestro interés fundamental es valorar si podemos considerar que las varianzas son iguales o que son distintas. Tenemos un problema de contraste de hipótesis. De hecho, estamos interesados en el siguiente contraste:

$$\begin{aligned} H_0 : \quad & \sigma_X^2 = \sigma_Y^2, \\ H_1 : \quad & \sigma_X^2 \neq \sigma_Y^2. \end{aligned}$$

que lo reformulamos como

$$\begin{aligned} H_0 : \quad & \frac{\sigma_X^2}{\sigma_Y^2} = 1, \\ H_1 : \quad & \frac{\sigma_X^2}{\sigma_Y^2} \neq 1. \end{aligned}$$

Bajo la hipótesis de que  $H_0 : \sigma_X^2 = \sigma_Y^2$  (o  $H_0 : \frac{\sigma_X^2}{\sigma_Y^2} = 1$ ) tenemos que

$$F = \frac{S_X^2}{S_Y^2} \sim F(n-1, m-1) \quad (25.10)$$

y podemos contrastar que las varianzas sean la misma rechazando la hipótesis nula de igualdad con un nivel de significación  $\alpha$  si

$$F < F_{\alpha/2}(n - 1, m - 1) \text{ o } F > F_{1-\alpha/2}(n - 1, m - 1).$$

**Ejemplo 25.8** Vamos a plantearnos si podemos considerar que la varianza de la expresión del gen en fila 54651 (de la matriz de expresión de tamidata::gse21942) es la misma en dos condiciones.

```
data(gse21942, package="tamidata")
y = exprs(gse21942)[54651,]
x = pData(gse21942) [, "FactorValue..DISEASE.STATE."]
```

En primer lugar podemos observar las varianzas muestrales de ambas muestras

```
aggregate(y, by=list(x), FUN=var)

##                   Group.1          x
## 1             healthy 0.1736921
## 2 multiple sclerosis 5.8991638
```

Comparamos las varianzas.

```
var.test(y~x)

##
##  F test to compare two variances
##
##  data:  y by x
##  F = 0.029444, num df = 14, denom df = 13,
##  p-value = 5.258e-08
##  alternative hypothesis: true ratio of variances is not equal to 1
##  95 percent confidence interval:
##  0.009553828 0.088680708
##  sample estimates:
##  ratio of variances
##  0.02944351
```

El cociente lo estimamos como 0.029. El intervalo de confianza para el cociente de las varianzas es [0.01, 0.089]. Como vemos el valor uno no está en el intervalo de confianza y no podemos considerar que la variabilidad de las medidas sean semejantes.

## 25.5 Comparación de medias con muestras apareadas

Volvemos al problema de comparar dos muestras. Ahora ya no hablamos de muestras independientes. No son dos zonas distintas en las que medimos una concentración y tomamos mediciones en cada una de ellas. O dos grupos de enfermos distintos de modo que en un grupo administraremos una medicación y en el otro grupo administraremos la otra. Ahora vamos a suponer que tenemos muestras *apareadas* o *emparejadas* (paired es la expresión inglesa). Por ejemplo, si medimos la

humedad en una localización un día y repetimos la medición otro día pero en el mismo lugar entonces tenemos dos observaciones apareadas. Son el mismo punto y lo que cambia es el momento en que observamos. Esto lo hacemos en  $n$  localizaciones distintas y tenemos:  $(x_i, y_i)$  con  $i = 1, \dots, n$ . El factor que empareja es la localización en que medimos.

Un ejemplo médico habitual es medir algo en un enfermo antes y después de tomar una medicación. Los datos son apareados pues corresponden a la misma persona. El factor que empareja es la persona.

Hablando con (un poco) de precisión lo que tenemos son  $n$  observaciones *independientes* de dos variables aleatorias que denotamos  $(X_i, Y_i)$  con  $i = 1, \dots, n$ . Notemos que las dos variables aleatorias  $X_i$  e  $Y_i$  no son *independientes*, están relacionadas porque bien corresponden a la misma localización o a una misma persona. Siendo  $\mu_X$  y  $\mu_Y$  las medias de las variables  $X$  e  $Y$  nos interesa (lo que no es mucha novedad) conocer la diferencia de medias  $\mu_X - \mu_Y$ . Como es sabido la media de la variable diferencia es la diferencia de las medias de cada variable, esto es,  $\mu_{X-Y} = \mu_X - \mu_Y$ . Teniendo en cuenta este resultado lo que vamos a hacer es olvidarnos de las variables  $X$  e  $Y$  y trabajar con la variable  $D = X - Y$  y contrastar si la media de la variable  $D$  es nula.

**Ejemplo 25.9** Supongamos las siguientes mediciones de humedad en  $n$  localizaciones con una diferencia de un día en la observación.

Los valores de las muestras  $x$  e  $y$  (mostramos los 10 primeros solamente por razones de espacio) son:

```
cbind(x[1:10],y[1:10])

##          [,1]      [,2]
## [1,] 47.65658 38.90872
## [2,] 24.43842 31.14831
## [3,] 37.64699 30.06900
## [4,] 39.42556 18.29267
## [5,] 25.56568 41.57664
## [6,] 30.50563 40.43124
## [7,] 40.20254 50.22449
## [8,] 28.01582 38.39682
## [9,] 33.39100 32.93366
## [10,] 40.88822 35.99563
```

Si hacemos lo indicado. Empezamos calculando las diferencias y vemos los 10 primeros valores.

```
d = x -y
d[1:10]

## [1]  8.7478665 -6.7098870  7.5779865
## [4] 21.1328860 -16.0109616 -9.9256049
## [7] -10.0219548 -10.3810040  0.4573352
## [10]  4.8925915
```

Ahora podemos obtener el estimador puntual de la diferencia de medias, el intervalo de confianza y el contraste de si la diferencia de medias vale cero utilizando t.test sobre las diferencias.

```
t.test(d)

##
##  One Sample t-test
##
## data: d
## t = -5.8304, df = 144, p-value = 3.489e-08
## alternative hypothesis: true mean is not equal to 0
## 95 percent confidence interval:
## -5.487038 -2.708603
## sample estimates:
## mean of x
## -4.097821
```

*Otra opción en la que no necesitamos calcular previamente las diferencias es la siguiente:*

```
t.test(x,y,paired=TRUE)

##
##  Paired t-test
##
## data: x and y
## t = -5.8304, df = 144, p-value = 3.489e-08
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## -5.487038 -2.708603
## sample estimates:
## mean of the differences
## -4.097821
```

*Como vemos lo que sale es lo mismo.*

**Ejemplo 25.10** Se trata de evaluar el efecto de la dieta y el ejercicio en el nivel de colesterol. Se midió la concentración antes y después de un programa de ejercicio aeróbico y cambio a una dieta baja en grasas. Los datos son los siguientes

```
(x = c(265,240,258,295,251,245,287,314,260,279,283,240,238,225,247))
## [1] 265 240 258 295 251 245 287 314 260 279 283
## [12] 240 238 225 247
(y = c(229,231,227,240,238,241,234,256,247,239,246,218,219,226,233))
## [1] 229 231 227 240 238 241 234 256 247 239 246
## [12] 218 219 226 233)
```

*donde x corresponde a los niveles antes e y a los niveles después. Podemos contrastar la igualdad de medias frente a la desigualdad con*

```
t.test(x,y,paired=T)
##
```

```
## Paired t-test
##
## data: x and y
## t = 5.4659, df = 14, p-value = 8.316e-05
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
## 16.32430 37.40904
## sample estimates:
## mean of the differences
## 26.86667
```

Podemos observar que el valor 0 no está en el intervalo de confianza. Por el tipo de dato que tenemos parece más natural contrastar las hipótesis  $H_0 : \mu_X \leq \mu_Y$  frente a  $H_1 : \mu_X > \mu_Y$ .

```
t.test(x,y,paired=T,alternative = "greater")

##
## Paired t-test
##
## data: x and y
## t = 5.4659, df = 14, p-value = 4.158e-05
## alternative hypothesis: true difference in means is greater than 0
## 95 percent confidence interval:
## 18.20922      Inf
## sample estimates:
## mean of the differences
## 26.86667
```

Vemos que el  $p$ -valor 0 es menor que  $\alpha = 0.05$  y por lo tanto rechazamos la hipótesis nula. De hecho, también rechazamos con un nivel de significación  $\alpha = 0.01$ .

## 25.6 Test de Kolmogorov-Smirnov para dos muestras

<sup>264</sup> Es aconsejable consultar [http://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov\\_test](http://en.wikipedia.org/wiki/Kolmogorov%E2%80%93Smirnov_test).

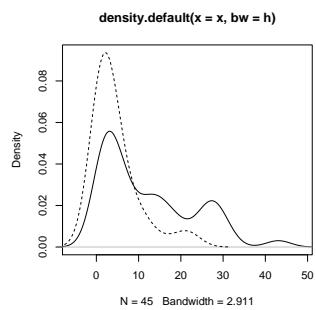


Figura 25.5: Estimadores kernel de la densidad de x (trazo continuo) e y (trazo discontinuo).

En esta sección nos ocupamos del test no paramétrico conocido como test de Kolmogorov-Smirnov para dos muestras.<sup>264</sup>

Hasta ahora hemos comparado poblaciones asumiendo que ambas poblaciones tienen una distribución normal. Las variables que observábamos seguían una distribución normal. ¿Qué ocurre cuando no lo podemos asumir? ¿Qué ocurre cuando nuestros datos son *marcadamente* no normales?

En la figura 25.5 tenemos los estimadores kernel de las densidades de dos muestras que pretendemos comparar. Vemos claramente que las formas de las densidades estimadas no se parecen en nada a los de una normal.

Un dibujo q-q para la muestra x aparece en la figura 25.6 y para la muestra y en la figura ???. Vemos que en ambas figuras los puntos se alejan de la línea. Ninguna de las dos muestras puede considerarse normal. Finalmente, si aplicamos un test de normalidad a las muestras (en concreto, vamos a utilizar el test de Shapiro-Wilk) obtenemos los siguientes resultados.

```

shapiro.test(x)

##
##  Shapiro-Wilk normality test
##
## data: x
## W = 0.88088, p-value = 0.0002536

shapiro.test(y)

##
##  Shapiro-Wilk normality test
##
## data: y
## W = 0.75051, p-value = 3.23e-08

```

Vemos cómo el test de normalidad rechaza claramente que podamos considerarlas muestras normales. No parece muy razonable utilizar un test de comparación de medias basada en la distribución t. ¿Hay otras opciones? Sí que las hay. Además no son malas soluciones. En esta sección nos ocupamos del test de Kolmogorov-Smirnov para dos muestras. Es un procedimiento *no paramétrico*. Un procedimiento se dice no paramétrico cuando no asume ningún modelo particular para los datos. No suponemos que son normales o que son binomiales o que son exponenciales, etc. A veces se les llama de distribución libre indicando que no estamos sujetos a un modelo específico. Esto suena bien. De hecho es bueno pero también tiene su pago. Asumimos menos hipótesis pero también son procedimientos con menos potencia. Les cuesta más rechazar la hipótesis nula.

```

png(paste0(dirTamiFigures, "CompDosPob44.png"))
qqnorm(x)
qqline(x)
dev.off()

## pdf
## 2

## pdf
## 2

```

La hipótesis nula que pretendemos contrastar la podemos formular como:

$H_0$ : Las muestras han sido extraídas de una misma población.

$H_1$ : Las muestras han sido extraídas de poblaciones distintas.

Para contrastar, como siempre, necesitamos un estadístico del contraste, un estadístico que compare ambas muestras. El estadístico no se basa en la comparación de las medias y varianzas muestrales como en los test anteriores. Denotamos las muestras como  $x_1, \dots, x_n$  e  $y_1, \dots, y_m$  de tamaños respectivos  $n$  y  $m$ . Consideramos las funciones de distribución empíricas o muestrales que denotamos  $F_n$  para muestra de las  $x$ 's y por  $G_m$  para la muestra de las  $y$ 's. Es decir:

$$F_n(z) = \frac{|\{x_i : x_i \leq z\}|}{n}.$$

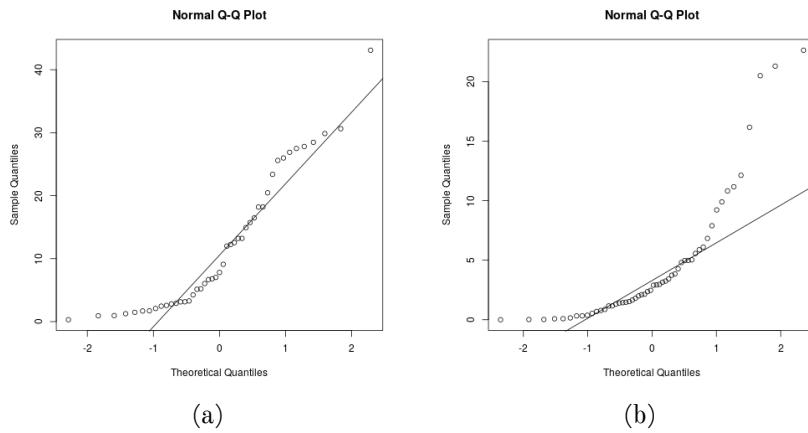


Figura 25.6: (a) Dibujo q-q de la muestra x. Vemos que los puntos se alejan de la lnea indicando que no hay normalidad en los datos. (b) Dibujo q-q de la muestra y. Vemos que los puntos se alejan de la lnea indicando que no hay normalidad en los datos.

Donde  $|\cdot|$  denota el cardinal del conjunto. En resumen,  $F_n(z)$  está contando el número de valores en la muestra  $x$  que son menores o iguales que  $z$ . La función  $G_m$  se define de un modo análogo con la segunda muestra. En la figura 25.7 mostramos las dos funciones  $F_n$  y  $G_m$ .

El estadístico del test es

$$D = \max_z |F_n(z) - G_m(z)|, \quad (25.11)$$

es decir,  $D$  nos da la máxima diferencia que observamos entre las funciones de distribución muestrales  $F_n$  y  $G_m$ . En la figura 25.7 representamos con un segmento vertical la máxima diferencia entre ambas funciones, esto es, el valor del estadístico  $D$ . Por la definición del estadístico  $D$  es claro que rechazamos para valores grandes de  $D$ . Si  $d$  es el valor observado entonces el p-valor vendría dado por

$$p = P(D \geq d),$$

donde en la probabilidad anterior asumimos la hipótesis nula.

```

library(Hmisc)
xy = c(x,y)
xcdf = ecdf(x)
ycdf = ecdf(y)
xy1 = xy[which.max(abs(xcdf(xy) - ycdf(xy)))]
png(paste0(dirTamiFigures, "CompDosPob46.png"))
Ecdf(xy, group= c(rep(1,length(x)),rep(2,length(y))), xlab="z")
segments(xy1, xcdf(xy1), xy1, ycdf(xy1), lty=2)
dev.off()

## pdf
## ?

```

**Ejemplo 25.11 (Función ks.test)** El test de Kolmogorov-Smirnov para dos muestras lo podemos aplicar con la función ks.test del siguiente modo:

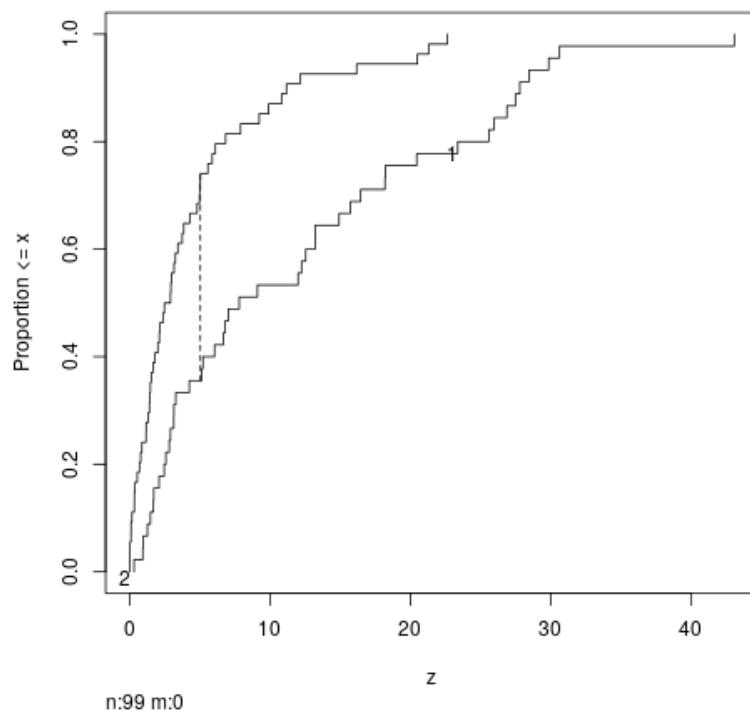


Figura 25.7: Funciones de distribución empíricas de ambas muestras. Vemos que la función de distribución de la segunda muestra (la muestra y indicada en la gráfica con el número 2) es mayor que la función de distribución empírica de la primera muestra (muestra x indicada con 1 en la gráfica). La longitud de la línea punteada nos muestra la máxima diferencia entre ambas funciones de distribución. Esta longitud es el valor del estadístico de

```
ks.test(x,y)

##
##  Two-sample Kolmogorov-Smirnov test
##
## data: x and y
## D = 0.38519, p-value = 0.000928
## alternative hypothesis: two-sided
```

*La salida se autoexplica. Observamos un  $p$ -valor de  $9 \times 10^{-4}$ . Si trabajamos con un nivel de significación de  $\alpha = 0.05$  entonces como el  $p$ -valor es menor que este nivel rechazamos la hipótesis nula. Podemos decir que hay diferencias significativas en la distribución de los datos a un nivel de significación 0.05.*

## 25.7 Test de Wilcoxon

<sup>265</sup> No asumimos ningún modelo paramétrico para las variables aleatorias. En particular, no asumimos normalidad.

Es un procedimiento no paramétrico<sup>265</sup> Pretendemos comparar dos poblaciones (dos condiciones experimentales por ejemplo). El valor aleatorio correspondiente a la primera población corresponde a una variable aleatoria  $X$  mientras que  $Y$  es la variable que nos describe a una observación de la segunda población. Tenemos dos muestras aleatorias:  $X_1, \dots, X_n$  una muestra de la primera población e  $Y_1, \dots, Y_m$  correspondiente a la segunda población.

## Capítulo 26

# Datos categóricos

### 26.1 Estimación de una proporción

Supongamos que denotamos por  $p$  la proporción a estimar (proporción de aprobados en un examen, proporción de votantes de un cierto partido político). Realmente disponemos (antes de tomar los datos) de una muestra aleatoria  $X_1, \dots, X_n$  donde cada valor aleatorio  $X_i$  puede tomar los valores 1 y cero con probabilidades  $p$  y  $1 - p$ . El estimador de  $p$  viene dado por

$$\hat{p} = \sum_{i=1}^n \frac{X_i}{n}.$$

Por el teorema central del límite sabemos que aproximadamente (cuando el número de datos  $n$  es grande)  $\hat{p}$  tiene una distribución normal con media  $p$  y varianza  $p(1 - p)/n$ , es decir,

$$\hat{p} \sim N(p, p(1 - p)/n).$$

Esto lo podemos escribir como

$$\frac{\hat{p} - p}{\sqrt{p(1 - p)/n}} \sim N(0, 1).$$

Sin embargo, esto no es utilizable para determinar un intervalo de confianza. Una opción es estimar la varianza mediante  $\hat{p}(1 - \hat{p})/n$  y considerar que aproximadamente (lo cual quiere decir que hemos de tener una muestra grande) se verifica

$$\frac{\hat{p} - p}{\sqrt{\hat{p}(1 - \hat{p})/n}} \sim N(0, 1).$$

Si denotamos (como hemos indicado antes) el percentil de orden  $\gamma$  ( $0 < \gamma < 1$ ) de una normal estándar como  $Z_\gamma$  entonces se verifica para un  $\alpha$  dado

$$P\left(-Z_{1-\alpha/2} \leq \frac{\hat{p} - p}{\sqrt{\hat{p}(1 - \hat{p})/n}} \leq Z_{1-\alpha/2}\right) = 1 - \alpha.$$

o, escrito de otro modo, que

$$P\left(\hat{p} - Z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1 - \hat{p})}{n}} \leq p \leq \hat{p} + Z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1 - \hat{p})}{n}}\right) = 1 - \alpha.$$

Tabla 26.1: Efecto preventivo de la aspirina

	Ataque fatal y no fatal	No ataque
Placebo	189	10845
Aspirina	104	10933

**Teorema 26.1 (Intervalo de confianza para una proporción)** Si  $X_1, \dots, X_n$  son una muestra aleatoria de variables binomiales con una prueba y probabilidad de éxito  $p$  entonces el intervalo

$$\hat{p} \pm Z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}$$

es un intervalo de confianza para la proporción  $p$  siendo

$$\hat{p} = \frac{\sum_{i=1}^n X_i}{n}.$$

**Nota de R 26.1 (R y lo buena que es la aspirina)** Vamos a obtener este intervalo de confianza utilizando R. Los datos con los que vamos a trabajar consideran si una persona toma aspirina o placebo y si ha sufrido un ataque cardíaco o no. Aparecen en la tabla 26.1.

Vamos a estimar la proporción de personas que tomando placebo tienen un ataque cardíaco.

```
library(Hmisc,T)
binconf(x=189, n=11034, method="asymptotic")

##      PointEst      Lower      Upper
##  0.01712887 0.01470788 0.01954987
```

Del mismo modo podemos estimar la proporción de los que, tomando aspirina, tienen ataque cardíaco. La estimación puntual y el intervalo de confianza vienen dados en la siguiente salida.

```
binconf(x=104, n=11037, method="asymptotic")

##      PointEst      Lower      Upper
##  0.00942285 0.007620423 0.01122528
```

En principio parece que si tomas aspirina tienes una probabilidad menor de tener el ataque. En cualquier caso en lo que sigue veremos el problema de comparar las proporciones.

### 26.1.1 Ejercicios

#### Ejercicio 60

Para los datos de la tabla 23.1 se pide:

1. Estimar la proporción de aptos para cada una de las universidades y para todo el sistema universitario valenciano.

### Ejercicio 61

[151, pág. 120, problema 21] Se observan los siguientes éxitos y fracasos: 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0. Calcule un intervalo de confianza con nivel 0,95 para la probabilidad de éxito  $p$ .

### Ejercicio 62

[151, pág. 120, problema 22] Teniendo en cuenta los siguientes resultados para una muestra de una distribución binomial, calcule el error estándar de  $\hat{p}$  cuando:

1.  $n = 25, X = 5.$
2.  $n = 48, X = 12.$
3.  $n = 100, X = 80.$
4.  $n = 300, X = 160.$

### Ejercicio 63

[151, pág. 120, problema 23] Entre los 100 adultos seleccionados al azar, 10 se encontraron desempleados. Dar una intervalo de confianza con un nivel de 0,99 para el porcentaje de adultos desempleados.

### Ejercicio 64

[151, pág. 121, problema 31] Una compañía de cosméticos encontró que 180 de cada 1000 mujeres seleccionadas al azar en Nueva York han visto el anuncio de televisión de la empresa. Calcule un intervalo de confianza al 0,95 para el porcentaje de mujeres en la ciudad de Nueva York que han visto el anuncio.

## 26.2 Tamaño de la muestra en la estimación de una proporción

En la sección anterior hemos visto un intervalo de confianza para estimar una proporción. Es el intervalo  $\left[ \hat{p} - Z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \hat{p} + Z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \right]$ . Este intervalo tiene nivel de confianza  $1 - \alpha$ . Dados unos datos tenemos el intervalo. Supongamos que la cosa ha ido bien: El intervalo que calculamos cubre al valor verdadero de  $p$ . Tenemos una confianza  $1 - \alpha$  de que esto sea cierto. Bien. El intervalo cubre a  $p$  pero nuestra estimación puntual de  $p$  es  $\hat{p}$ . Si nos piden que demos un valor para  $p$  responderemos dando la estimación puntual, dando el valor  $\hat{p}$ . La diferencia entre la estimación puntual que damos y el valor real de  $p$  (que desconocemos y siempre desconoceremos) es como mucho la mitad de la longitud del intervalo de confianza (siempre asumiendo que  $p$  está dentro de este intervalo). En otras palabras:

$$|\hat{p} - p| \leq Z_{1-\alpha/2} \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}.$$

Un intervalo que acierte mucho es bueno, esto es, un nivel de confianza alto es bueno. De hecho, menos de un 90% no se suele utilizar.

Pero no sólo importa el nivel de confianza. Por ejemplo, el intervalo  $[0, 1]$  tiene un nivel de confianza de 1. ¿Cuál es la proporción de personas que sufren ataque de corazón tomando aspirina? Si respondemos que esta proporción está entre 0 y 1 estamos haciendo una afirmación con una confianza de uno. No nos equivocamos en absoluto. *Pero también la afirmación es absolutamente inútil.* No nos dice nada que no supieramos previamente. En resumen el intervalo que damos para estimar  $p$  tiene que ser tal que *confiemos* que contiene el valor que queremos conocer pero también tiene que ser preciso, tan estrecho como podamos para que sea informativo. Es fácil suponer que precisión supone más muestra, más datos.

El problema se plantea normalmente del siguiente modo. Quiero estimar la proporción  $p$  y quiero que hacerlo con un error máximo dado  $\delta$  (por ejemplo,  $\delta = 0.02$ ). Necesitamos una primera muestra. ¿Para qué? Para tener una estimación inicial de  $p$ . Denotemos el tamaño de esta primera muestra como  $n_0$  y la estimación puntual de  $p$  obtenida como  $\hat{p}_0$ . Entonces podemos plantearnos qué valor de  $n$  verifica que

$$Z_{1-\alpha/2} \sqrt{\frac{\hat{p}_0(1 - \hat{p}_0)}{n}} \leq \delta$$

Esto es simple. Despejamos en la desigualdad anterior el valor de  $n$  y tenemos que

$$n \geq Z_{1-\alpha/2}^2 \frac{\hat{p}_0(1 - \hat{p}_0)}{\delta^2}$$

La estimación inicial de la proporción no necesariamente la tenemos que obtener de una muestra inicial. Puede que tengamos alguna muestra previa de esa población o alguna estimación que obtengamos de alguna publicación.

### 26.2.1 Exercises

#### Ejercicio 65

Pretendemos estimar la proporción de palmeras afectadas por el pi-cudo. Se ha tomado una primera muestra de 100 palmeras al azar. Se han observado 23 palmeras afectadas. Se pide:

1. ¿Cuál es el error máximo observado con un nivel de confianza de 0.95?
2. Tomamos como estimación inicial de  $p$  el valor observado con la primera muestra de 100 palmeras. Supongamos que nos planteamos estimar la proporción de palmeras afectadas con un error máximo de 0.04 y un nivel de confianza de 0.95. ¿Cuál ha de ser el número total de palmeras a observar?
3. Responde a la pregunta del apartado 2 suponiendo que deseamos un nivel de confianza en el error máximo de 0.99. Mantenemos el error máximo en 0.04.
4. Responde a la pregunta del apartado 2 suponiendo que deseamos un nivel de confianza en el error máximo de 0.95 pero queremos un error máximo de 0.02.
5. ¿Más nivel de confianza supone más muestra? Responde la pregunta considerando los apartados anteriores.

6. ¿Menor error supone una mayor muestra? Responde la pregunta utilizando los apartados anteriores?.

## 26.3 Varias variables categóricas

En esta sección vemos algunas cuestiones básicas cuando trabajamos con datos categóricos. Sin duda, una magnífica referencia bibliográfica para saber más (muchísimo más y muchísimo mejor) es [3]. Todo este material está tomado (notación incluida) de este texto. Es un breve y, a veces, críptico resumen de algunas partes de los temas 2 y 3 de [3]. Fundamentalmente utilizamos el test exacto de Fisher.

### 26.3.1 Probabilidad y tabla de contingencia

Supongamos que tenemos dos variables aleatorias  $X$  e  $Y$  con  $I$  y  $J$  categorías. Su **distribución conjunta** viene dada por

$$\pi_{ij} = P(X = i, Y = j),$$

con  $i = 1, \dots, I$  y  $j = 1, \dots, J$ . Es decir, la probabilidad de que simultáneamente  $X$  tome el valor  $i$  y la variable  $Y$  tome el valor  $j$ . También podemos considerar el comportamiento *marginal* de estas variables y preguntarnos por las **distribuciones marginales** que se definen como

$$\begin{aligned}\pi_{i\cdot} &= P(X = i) = \sum_{j=1}^J P(X = i, Y = j) = \sum_{j=1}^J \pi_{ij} \\ \pi_{\cdot j} &= P(Y = j) = \sum_{i=1}^I P(X = i, Y = j) = \sum_{i=1}^I \pi_{ij}\end{aligned}$$

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 y sí de **distribución condicionada** que se define como

$$P(Y = j|X = i) = \pi_{j|i} = \frac{\pi_{ij}}{\pi_{i\cdot}}$$

Dos variables aleatorias categóricas se dicen independientes si

$$P(X = i, Y = j) = P(X = i)P(Y = j)$$

esto es

$$\pi_{ij} = \pi_{i\cdot}\pi_{\cdot j}.$$

En particular, la condicionada es igual a la marginal.

$$\pi_{j|i} = \pi_{\cdot j} \text{ con } j = 1, \dots, J.$$

Si  $X$  e  $Y$  son variables respuesta entonces hablamos de *independencia*. Si  $Y$  es respuesta y  $X$  explicativa hablamos de *homogeneidad*.

Supongamos que tomamos una muestra  $(x_i, y_i)$  con  $i = 1, \dots, n$ . ¿Cómo estimamos la distribución conjunta, las marginales y las condicionadas? Un ejemplo puede ser tomar una muestra de hombres en un cierto grupo de edad. Asignamos a cada individuo en el estudio a uno de dos posibles grupos: toma una aspirina diaria (grupo aspirina)

Tabla 26.2: Tabla de contingencia para  $X =$  toma aspirina o placebo ( $I = 2$ ) e  $Y =$  sufre ataque cardíaco o no ( $J = 2$ ).

	Ataque fatal	Ataque no fatal	No ataque
Placebo	18	171	10845
Aspirina	5	99	10933

Tabla 26.3: Tabla de contingencia  $2 \times 2$ .

	Test positivo	Test negativo	Total
Enfermo	$n_{11}$	$n_{12}$	$n_{1\cdot}$
No enfermo	$n_{21}$	$n_{22}$	$n_{2\cdot}$
Total	$n_{\cdot 1}$	$n_{\cdot 2}$	$n$

o bien le damos diariamente un placebo (grupo placebo). Se toma una gran muestra y los resultados los tenemos en la tabla 26.2 donde el número que aparece en cada celda de la tabla indica el número de veces que se da simultáneamente el valor correspondiente de la fila y columna consideradas. Esta tabla recibe el nombre de *tabla de contingencia*<sup>1</sup> o *tabla de clasificación cruzada*. De un modo genérico podemos considerar la tabla 26.3 donde denotamos por  $n_{ij}$  el número de ocurrencias del valor  $i$  de  $X$  y del valor  $j$  de  $Y$ . Por  $n_{i\cdot}$  denotamos la suma de los conteos de la fila, esto es, del total de veces que  $X$  toma el valor  $i$ . Análogamente por  $n_{\cdot j}$  denotamos el total de la columna  $j$  y nos da el número total de veces que la variable  $Y$  ha tomado el valor  $j$ . Notemos que estimamos la distribución conjunta (ver tabla 26.4) con

$$\hat{P}(X = i, Y = j) = \hat{\pi}_{ij} = \frac{n_{ij}}{n} \quad (26.1)$$

La distribución condicionada de la variable  $Y$  a la variable  $X$  (tabla 26.5) la estimaríamos con

$$\hat{P}(Y = j|X = i) = \hat{\pi}_{j|i} = \frac{n_{ij}}{n_{i\cdot}}. \quad (26.2)$$

De un modo similar podemos estimar la distribución condicionada de la variable  $X$  a la variable  $Y$ .

$$\hat{P}(X = i|Y = j) = \hat{\pi}_{i|j} = \frac{n_{ij}}{n_{\cdot j}}. \quad (26.3)$$

De un modo genérico la tabla 26.2 correspondería a la tabla 26.3. A partir de estos conteos tendremos la distribución conjunta estimada que mostramos en la tabla 26.4. La distribución condicional estimada de  $Y$  a  $X$  (de la columna a la fila) la mostramos en la tabla 26.5.

### 26.3.2 Comparación de proporciones

Muchos estudios se diseñan para comparar grupos basándonos en una respuesta binaria,  $Y$ . Con dos grupos tenemos una tabla de contin-

---

<sup>1</sup> Nombre debido a Karl Pearson.

Tabla 26.4: Distribución conjunta estimada

$\hat{\pi}_{ij}$	Test positivo	Test negativo	Total
Enfermo	$n_{11}/n$	$n_{11}/n$	$n_{1\cdot}/n$
No enfermo	$n_{21}/n$	$n_{22}/n$	$n_{2\cdot}/n$
Total	$n_{\cdot 1}/n$	$n_{\cdot 2}/n$	1

Tabla 26.5: Distribución condicionada estimada de Y a X.

$\hat{\pi}_{j i}$	Test positivo	Test negativo	Total
Enfermo	$n_{11}/n_{1\cdot}$	$n_{11}/n_{1\cdot}$	1
No enfermo	$n_{21}/n_{2\cdot}$	$n_{22}/n_{2\cdot}$	1

gencia  $2 \times 2$ . Supongamos que denotamos las probabilidades condicionadas del siguiente modo:

$$P(Y = 1|X = i) = \pi_i, \quad (26.4)$$

$$P(Y = 2|X = i) = 1 - \pi_i. \quad (26.5)$$

Estudiar la posible asociación entre las variables  $X$  e  $Y$  es equivalente a comparar las proporciones  $\pi_1$  y  $\pi_2$ . ¿Cómo comparamos las proporciones? Podemos estimar la diferencia de las proporciones<sup>2</sup>  $\pi_1 - \pi_2$ . Otra opción es trabajar con odds. ¿Qué son los odds?

**Definición 26.1 (Odds)** Para un suceso aleatorio  $A$  con probabilidad  $P(A)$  los odds se definen como el cociente

$$\frac{P(A)}{1 - P(A)}.$$

En concreto se trata de considerar los odds del suceso  $Y = 1$  en la distribución condicionada a  $X = 1$  frente a los odds del suceso  $Y = 1$  en la distribución condicionada a  $X = 2$ . Estamos considerando el siguiente cociente

$$\theta = \frac{\pi_1/(1 - \pi_1)}{\pi_2/(1 - \pi_2)}.$$

Este *cociente* de odds (odds ratio en inglés) es muy útil como medida de asociación en una tabla  $2 \times 2$ . Se comprueba fácilmente<sup>3</sup> que

$$\theta = \frac{\pi_{11}\pi_{22}}{\pi_{12}\pi_{21}}.$$

Por ello también se le llama el *cociente de los productos cruzados*. El cociente de los odds tiene las siguientes propiedades:

- Puede ser cualquier valor positivo.

<sup>2</sup>Por su simplicidad suele la opción más habitual en los textos introductorios de Estadística.

<sup>3</sup>Tenemos que

$$\theta = \frac{\pi_1/(1 - \pi_1)}{\pi_2/(1 - \pi_2)} = \frac{\pi_{1|1}/(1 - \pi_{1|1})}{\pi_{1|2}/(1 - \pi_{1|2})}, \quad (26.6)$$

pero  $\pi_{j|i} = \pi_{ij}/\pi_i$ , y sustituyendo se sigue inmediatamente el resultado.

- $\theta = 1$  significa que no hay asociación entre  $X$  e  $Y$ , que  $X$  e  $Y$  son independientes.
- Valores de  $\theta$  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.

### 26.3.3 Test de Fisher

Consideramos una tabla  $2 \times 2$ . 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

$$p(t) = P(n_{11} = t) = \frac{\binom{n_1}{t} \binom{n_{.1}-t}{n_{.1}}}{\binom{n}{n_{.1}}}$$

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  $2 \times 2$  lo podemos formular en términos del cociente de odds,  $\theta$ . Recordemos que la independencia de las variables binarias o dicotómicas es equivalente a que el cociente de odds sea igual a 1.

$$H_0 : \theta = 1$$

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 26.1** *Muriel Bristol* indicó en una ocasión a *R.A. Fisher* que era capaz de diferenciar si en un té con leche se había servido primero el té y luego la leche o bien primero la leche y luego el té.<sup>4</sup> Se realiza la experiencia consistente en darle a probar 8 tazas de té a dicha señora. Ella sabe que hay cuatro tazas preparadas en el orden té y luego leche y cuatro tazas preparadas en el orden leche y luego té. ¿Sabía esta señora diferenciar el orden o respondía de un modo aleatorio? Este es el problema conocido como *la señora probando el té*. En la tabla 26.6 mostramos un posible resultado de la experiencia.

En este experimento 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  $n_{11}$ .

---

<sup>4</sup> Realmente Fisher lo comenta en su libro The Design of Experiments y más que probablemente es una bonita leyenda pero que le da interés al problema.

Tabla 26.6: Fisher y el té.

Primer servicio	Predicción primer servicio		Total
	Leche	Té	
Leche	3	1	4
Té	1	3	4
Total	4	4	



## Capítulo 27

# Modelos lineales

El problema que se trata en este tema es básico. Estudiar relaciones entre una variable que llamaremos *variable respuesta* y una (o más de una) variable que llamaremos *variables predictoras*. También se utilizan las denominaciones de *variable dependiente* para la variable respuesta y *variables independientes* en lugar en predictoras.

Si denotamos a la variable respuesta por  $Y$  (al valor observado lo denotaremos por  $y$ ) y a la variable predictora por  $X$  nuestro problema es intentar conocer el valor de  $Y$  cuando conocemos el valor de la variable  $X$ .

**Ejemplo 27.1** Vamos a utilizar los datos tamidata:gse44456.

```
library(BioBase)
data(gse44456, package="tamidata")
```

Nos fijamos en el gen que ocupa la fila 19254 en la matriz de expresión.

```
y = exprs(gse44456)[19254,]
```

¿Qué gen es?

```
featureNames(gse44456)[19254]
## [1] "8047097"
```

Empezamos representando en figura 27.1 la expresión del gen como función de la tasa de crecimiento de la célula.

```
## pdf
## 2
```

Nuestro problema es intentar predecir la expresión del gen a partir de la tasa de crecimiento.

Nada de lo que sigue tiene ningún sentido biológico. Es simplemente un ejemplo para ilustrar.

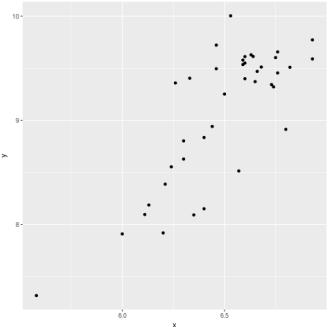


Figura 27.1: Expresión del gen 19254 frente al pH.

### 27.1 Coeficiente de correlación de Pearson

Tenemos la matriz de expresión y consideramos el perfil de expresión de dos genes. Supongamos que nos planteamos la siguiente pregunta: ¿están relacionadas linealmente las expresiones de los genes que

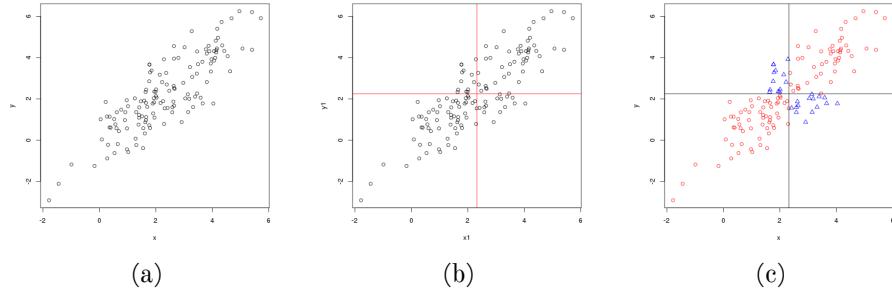


Figura 27.2: a) Datos. b) Los datos con dos líneas: la línea horizontal que corta al eje de ordenadas en  $\bar{y}$  y la línea vertical que corta al eje de abscisas en  $\bar{x}$ . c) Los datos, la línea horizontal que corta al eje de ordenadas en  $\bar{y}$  y la línea vertical que corta al eje de abscisas en  $\bar{x}$ . Representamos en rojo aquellos puntos donde el producto cruzado  $(x_i - \bar{x})(y_i - \bar{y})$  es positivo y en azul aquellos puntos donde el producto toma un valor negativo.

estamos considerando? Denotemos los niveles de expresión del primer y segundo gen mediante  $(x_i, y_i)$  con  $i = 1, \dots, n$ . Es decir, en la muestra i el gen 1 se expresa  $x_i$  y el gen 2 se expresa  $y_i$ . Supongamos que los datos observados son los que aparecen en la figura ??.

Un valor que se utiliza frecuentemente para responder esta pregunta fué propuesto por [Karl Pearson](#) es el *coeficiente de correlación de Pearson*. Se define del siguiente modo.

### Definición 27.1 (Coeficiente de correlación de Pearson)

$$r = \frac{\sum_{i=1}^n (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^n (x_i - \bar{x})^2} \sqrt{\sum_{i=1}^n (y_i - \bar{y})^2}}.$$

Veamos una ilustración gráfica del concepto. En la figura 27.2(a) mostramos los datos con los que vamos a trabajar.

En la figura 27.2(b) añadimos al diagrama de puntos mostrado en la figura 27.2(a) un par de líneas que se cortan en el punto  $(\bar{x}, \bar{y})$ : la línea horizontal que corta al eje de ordenadas en  $\bar{y}$  y la línea vertical que corta al eje de abscisas en  $\bar{x}$ .

Consideraremos los productos cruzados  $(x_i - \bar{x})(y_i - \bar{y})$ . En la figura 27.2(c) reproducimos el dibujo de la figura ??(b). Simplemente, representamos en rojo aquellos puntos donde el producto cruzado anterior es positivo y en azul aquellos puntos donde el producto toma un valor negativo.

Tenemos una relación razonablemente lineal entre las dos variables y por ello vemos muchos más puntos rojos que azules. El coeficiente de correlación lineal de Pearson vale

```
cor(x1, y1)
## [1] 0.7769814
```

Vamos a repetir el mismo análisis con otros dos conjuntos de datos. En uno de ellos la asociación lineal es mayor mientras que en el otro buscaremos que no haya prácticamente asociación lineal entre las variables.

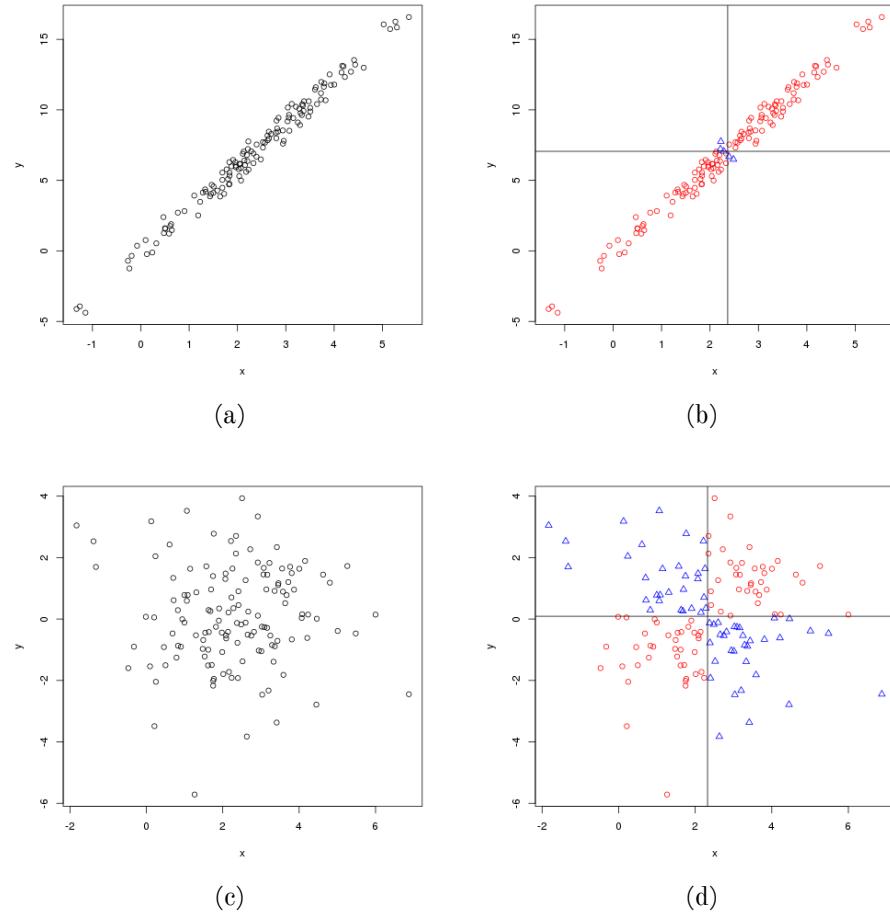


Figura 27.3: Ejemplo con fuerte asociación lineal: datos (a) y los datos diferenciando el signo del producto cruzado. Las figuras (c) y (d) son los dibujos análogos con datos en los que apenas hay asociación lineal entre las abscisas y las ordenadas.

En la figura 27.3(a) mostramos los datos con fuerte asociación lineal. Distinguimos por el signo del producto cruzado en la figura 27.3(b). Vemos cómo hay muchos puntos azules. El nuevo coeficiente de correlación lineal de Pearson es

```
cor(x2,y2)
## [1] 0.9889923
```

Mayor que en el caso anterior. Y un tercer ejemplo donde la asociación lineal es casi inexistente. En las figuras 27.3(c) y (d) mostramos los dibujos análogos al caso anterior. Vemos cómo hay muchos más puntos azules que en los dos casos anteriores. El nuevo coeficiente de correlación lineal de Pearson es

```
cor(x3,y3)
## [1] 0.08869201
```

Esta es la ilustración gráfica de la idea que subyace a este concepto. Cuanto más se aproximan los puntos a una línea recta mayor es el valor absoluto del coeficiente de correlación, más próximo a uno. Si cuando x crece y crece entonces la recta el valor de coeficiente de correlación es positivo. Si cuando x crece y decrece entonces el coeficiente de correlación es negativo.

Si vemos la definición 27.1 que acabamos de dar se tiene que

$$r = \frac{s_{xy}}{s_x s_y}.$$

La covarianza muestral

$$s_{xy} = \frac{\sum_{i=1}^n (x_i - \bar{x}_n)(y_i - \bar{y}_n)}{n-1},$$

está estimando la cantidad

$$E(X - \mu_X)(Y - \mu_Y),$$

mientras que las desviaciones típicas muestrales,  $s_x$  y  $s_y$ , no son más que estimaciones de las desviaciones típicas poblacionales  $\sigma_X$  y  $\sigma_Y$ . En resumen que  $r = s_{xy}/(s_x s_y)$  no es más que un estimador del coeficiente de correlación de Pearson que damos en la siguiente definición.

**Definición 27.2** *Dadas dos variables aleatorias X e Y definimos su coeficiente de correlación de Pearson como*

$$\rho = \frac{E(X - \mu_X)(Y - \mu_Y)}{\sigma_X \sigma_Y}.$$

Esta cantidad es una cuantificación de la asociación lineal entre X e Y. Es importante notar que aquí no hay predictor y respuesta. Ambas variables son tratadas de un modo simétrico. De hecho si intercambiamos las variables el valor del coeficiente de correlación no se modifica, es el mismo. Además se tiene que

$$-1 \leq \rho \leq 1.$$

Se puede probar que si  $\rho = 1$  entonces existe dos constantes  $a$  y  $b$  (con  $b$  positiva) tales que  $Y = a + bX$  (con probabilidad uno). Si  $\rho = -1$  entonces existen  $a$  y  $b$  (con  $b$  negativa) tales que  $Y = a + bX$  (con probabilidad uno). En resumen que si el coeficiente de correlación es 1 o -1 entonces una variable es función lineal de la otra. La recta es creciente si la correlación es positiva y decreciente si es negativa.

## 27.2 Matriz de covarianzas y matriz de correlaciones

Ahora consideramos una muestra de dimensión  $d$ ,  $y_1, \dots, y_n$ . Si

$$y_i = \begin{bmatrix} y_{i1} \\ \vdots \\ y_{ip} \end{bmatrix}$$

Entonces podemos representar toda la muestra como la siguiente matriz

$$Y = \begin{bmatrix} y'_1 \\ \vdots \\ y'_n \end{bmatrix} = \begin{bmatrix} y_{11} & \dots & y_{1d} \\ \vdots & \vdots & \vdots \\ y_{n1} & \dots & y_{nd} \end{bmatrix}$$

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

$$\bar{y} = \frac{1}{n} \sum_{i=1}^n y_i = \frac{1}{n} Y' 1_n. \quad (27.1)$$

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

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

La matriz de covarianzas muestral es aquella matriz que tiene en la posición  $(j, k)$  la covarianza muestral entre las componentes  $j$  y  $k$ , es decir,

$$s_{jk} = \frac{1}{n-1} \sum_{i=1}^n (y_{ij} - \bar{y}_{\cdot j})(y_{ik} - \bar{y}_{\cdot k}),$$

de modo que

$$S = \begin{bmatrix} s_{11} & \dots & s_{1d} \\ \vdots & \vdots & \vdots \\ s_{d1} & \dots & s_{dd} \end{bmatrix} = \frac{1}{n-1} \sum_{i=1}^n (y_i - \bar{y})(y_i - \bar{y})'.$$

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

$$r_{jk} = \frac{\sum_{i=1}^n (y_{ij} - \bar{y}_{\cdot j})(y_{ik} - \bar{y}_{\cdot k})}{\sqrt{\sum_{i=1}^n (y_{ij} - \bar{y}_{\cdot j})^2 \sum_{i=1}^n (y_{ik} - \bar{y}_{\cdot k})^2}} = \frac{s_{jk}}{\sqrt{s_{jj}s_{kk}}} \quad (27.2)$$

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

**Ejemplo 27.2 (Matrices de covarianza y correlación muestrales)** ¿Cómo calculamos las matrices de covarianzas y de correlaciones con *R*.

Supongamos la siguiente matriz de expresión con 9 genes y cuatro muestras.

```
M
##      muestra1 muestra2 muestra3 muestra4
## gen1 25.43529 23.05873 26.00079 25.52829
## gen2 22.10927 23.71074 22.34906 25.56481
## gen3 19.98967 23.50403 22.84677 21.50219
## gen4 18.08352 22.33291 25.65621 22.68493
## gen5 19.15375 24.79281 22.24346 23.45789
## gen6 28.75482 22.20723 26.29208 23.49364
## gen7 25.09714 22.19309 25.49735 25.48533
## gen8 26.32781 25.10382 24.12959 26.25590
## gen9 20.37034 18.94016 25.54662 25.15155
```

La matriz de covarianzas entre las muestras viene dada por

```
cov(M)
```

y las covarianzas entre los genes por

```
cov(t(M))
```

Análogamente la matriz de correlación entre las muestras es

```
cor(M)
```

y entre los genes viene dada por

```
cor(t(M))
```

### 27.3 Regresión lineal simple

En todos los ejemplos antes comentados el problema común es determinar el valor de  $Y$  a partir del valor de  $X$ . Obviamente la respuesta más simple sería buscar una función que podemos denotar por  $f$  de modo que para un valor dado  $x$  simplemente calculamos  $y = f(x)$ . Un poco de imaginación y conocimiento de la posible relación entre  $x$  e  $y$  podrían darnos una idea de qué función  $f$  buscar. Este planteamiento es de base muy restrictivo. ¿Por qué? Pues en primer lugar porque estamos asumiendo que, para un valor de  $x$ , existe un único valor de  $y$  asociado. Y esto nunca (o casi) es así. Un detalle, a veces  $X$  es una variable aleatoria que observamos simultáneamente con  $Y$ , en otras ocasiones es un valor que nosotros prefijamos (dosis de medicación, tratamiento en un problema de diseño de experimentos). Sin embargo, desde el punto de vista de la regresión  $X$  siempre lo consideramos fijo y estudiamos cómo se comporta  $Y$  dado el valor de  $X = x$ . Es decir, de la distribución condicionada de  $Y$  al valor de  $X = x$ .

Un ejemplo muy famoso de Galton. Se tomaba como variable predictora la estatura del padre y como variable respuesta o a predecir, la

estatura de un hijo. Es claro que para un mismo padre la estatura de sus hijos es variable. No todos los hijos de un mismo padre miden lo mismo. No tiene ningún sentido asumir una relación funcional entre la estatura de un padre y la de un hijo.

Tan tontos no son los estadísticos (que no estadistas). De hecho, lo que se modeliza es la relación entre el valor  $x$  y *el valor medio* de la variable  $Y$  dado ese valor  $x$ . Siguiendo con el ejemplo de Galton. Si consideramos un padre de estatura  $X = 178$  centímetros. Supondremos que la media de la variable  $Y$  que nos da la estatura aleatoria de un hijo es la que se relaciona con  $x$ . Denotemos por  $E[Y | x]$  esta media (estatura media de todos los hijos de un padre con estatura 178 centímetros). Hemos de admitir que además de lo que mide el padre, algo tendrá que decir la madre, y también otros muchos factores que todos podemos imaginar. De modo que  $Y$ , conocida la estatura del padre, sigue siendo una cantidad aleatoria. De hecho, se asume que la distribución de  $Y$  es normal cuya media depende de  $x$ ,  $E[Y | x]$ , pero cuya varianza no depende de  $x$ , es decir, es una cantidad constante que denotaremos por  $\sigma^2$ . En resumen, estamos asumiendo que

$$Y \sim N(E[Y | x], \sigma^2). \quad (27.3)$$

En el modelo de regresión más simple con el que se trabaja se asume que la media condicionada  $E[Y | x]$  es una función lineal de  $x$ , en otras palabras, se asume que

$$E[Y | x] = \beta_0 + \beta_1 x. \quad (27.4)$$

Las hipótesis asumidas en 27.3 y 27.4, podemos expresarlas conjuntamente diciendo que la variable respuesta  $Y$  se puede expresar como

$$Y = \beta_0 + \beta_1 x + \epsilon, \quad (27.5)$$

donde

$$\epsilon \sim N(0, \sigma^2). \quad (27.6)$$

En la formulación de 27.5 expresamos el valor aleatorio de  $Y$  como suma de una parte que *sistemáticamente* depende de  $x$  (la componente sistemática del modelo) y un término aleatorio con distribución normal, un término de error o desajuste del modelo. En esta variable normal con media cero y varianza constante  $\sigma^2$  estamos incluyendo todas las posibles causas que influyen el valor de  $Y$  y que no vienen dadas por la variable predictora.

No consideramos un solo valor aleatorio de  $Y$  dado un valor fijo de  $x$ . Realmente, tenemos  $n$  valores observados cuyos valores son *independientes entre sí pero no tienen la misma distribución*. Hemos de pensar que cada  $Y_i$  tiene una variable predictora distinta que influye en la distribución de  $Y_i$ . Tenemos pares  $(x_i, Y_i)$  donde la  $x_i$  viene dada y consideramos la distribución de  $Y_i$  condicionada a  $x_i$ , es decir,  $Y_i | x_i$ .

Resumiendo, estamos asumiendo que  $Y_i \sim N(\beta_0 + \beta_1 x_i, \sigma^2)$  y que los distintos  $Y_i$  son independientes entre si. Utilizando propiedades de la distribución normal multivariante tenemos que estas hipótesis las podemos expresar conjuntamente diciendo que

$$Y \sim N_n(X\beta, \sigma^2 I_{n \times n}), \quad (27.7)$$

donde

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

Si consideramos que

$$\epsilon = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

donde los  $\epsilon_i \sim N(0, \sigma^2)$  e independientes entre si. Entonces el modelo dado en 27.7 lo podemos reescribir como

$$Y = X\beta + \epsilon, \quad (27.8)$$

con  $\epsilon \sim N_n(0, \sigma^2 I_{n \times n})$ .

Este modelo probabilístico es conocido como el **modelo de regresión lineal simple**. No lo estudiaremos en más detalle porque nos vamos a ocupar de la situación más general en que tenemos más de una variable predictor. No es más que un caso particular y sin mucha dificultad adicional se puede estudiar el situación general de *regresión lineal múltiple*.

## 27.4 Regresión lineal múltiple

Pretendemos determinar la relación que liga a una variable respuesta  $Y$  como función de  $p-1$  variables predictoras,  $x_1, \dots, x_{p-1}$ . Siguiendo el razonamiento anterior podemos plantearnos un modelo muy general como el que sigue.

$$Y = f(x_1, \dots, x_n) + \epsilon, \quad (27.9)$$

donde  $f$  es una función desconocida y  $\epsilon$  es el término del error. Vamos a asumir una situación más simple. En concreto que la función  $f$  es lineal de modo que la relación sería  $Y = \beta_0 + \beta_1 x_1 + \dots + \beta_{p-1} x_{p-1}$ . Realmente observamos  $n$  vectores  $(y_i, x_{i1}, \dots, x_{ip-1})$  en consecuencia nuestro modelo estocástico ha de considerar el modelo para los  $n$  valores aleatorios  $Y_i$ , donde cada  $Y_i$  tiene asociado un vector  $x_i$ . Vamos a suponer que para una combinación de valores  $(x_{i1}, \dots, x_{ip-1})$  vamos a observar un valor aleatorio  $Y_i$  con distribución normal cuya media es  $\beta_0 + \beta_1 x_{i1} + \dots + \beta_{p-1} x_{ip-1}$  y cuya varianza va a ser constante e igual a  $\sigma^2$ . Además los distintos  $Y_i$  son independientes entre si.

Vamos a formular conjuntamente estas hipótesis. Denotaremos

$$Y = \begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix} \quad X = \begin{bmatrix} 1 & x_{11} & \dots & x_{1,p-1} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{n1} & \dots & x_{n,p-1} \end{bmatrix} \quad \beta = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_{p-1} \end{bmatrix} \quad \epsilon = \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

El modelo estocástico básico es el que sigue

$$Y = X\beta + \epsilon. \quad (27.10)$$

En este modelo a  $X\beta$  le llamaremos la parte sistemática mientras que  $\epsilon$  es la componente aleatoria del modelo. Estamos expresando los datos

como la suma de una parte sistemática más una parte aleatoria. La parte sistemática es la media del vector  $Y$ . Notemos que la dimensión del espacio en que estamos trabajando es  $n$ , el número de observaciones. El vector de medias o parte sistemática del modelo tiene dimensión  $p$  por lo que la parte no explicada, el residuo, tiene dimensión  $n - p$ .

## 27.5 Estimación de $\beta$

¿Cómo estimamos los parámetros  $\beta$ ? Nuestros datos son  $(y_i, x_{i1}, \dots, x_{ip})$  con  $i = 1, \dots, n$ . Nuestro objetivo es estimar los coeficientes  $\beta$  de modo que  $X\beta$  esté próximo a  $y$ . En concreto vamos a minimizar

$$\sum_{i=1}^n \epsilon_i^2 = \epsilon' \epsilon = (y - X\beta)'(y - X\beta). \quad (27.11)$$

Si desarrollamos la expresión anterior tendremos que

$$(y - X\beta)'(y - X\beta) = y'y - 2\beta'X'y + \beta'X'X\beta.$$

Diferenciando respecto de los distintos  $\beta_j$  e igualando a cero nos da el siguiente sistema de ecuaciones normales:

$$X'X\hat{\beta} = X'y. \quad (27.12)$$

Si asumimos que la matriz  $X'X$  es una matriz no singular entonces tendremos que

$$\hat{\beta} = (X'X)^{-1}X'y, \quad (27.13)$$

y en consecuencia se sigue que

$$X\hat{\beta} = X(X'X)^{-1}X'y = Hy. \quad (27.14)$$

La matriz  $H = X(X'X)^{-1}X'y = Hy$  es la matriz de proyección de  $y$  sobre el espacio engendrado por los  $p$  vectores columna de la matriz  $X$ . Es una matriz  $n \times n$ .

Utilizando la matriz  $H$  podemos calcular las predicciones para cada una de los  $x_i$  originales. Vienen dadas por

$$\hat{y} = Hy = X\hat{\beta}. \quad (27.15)$$

También tenemos los residuos, esto es, las diferencias entre los valores observados originalmente y las predicciones que de ellos hacemos. Los residuos en términos de la matriz  $H$  vienen dados por

$$\hat{\epsilon} = y - Hy = (I - H)y. \quad (27.16)$$

Finalmente, hemos determinado los coeficientes que nos minimizaban la suma de cuadrados. El valor mínimo que hemos obtenido que recibe el nombre de **suma de cuadrados residual** o **suma de cuadrados del error** que viene dada por

$$SS(Error) = \sum_{i=1}^n (y_i - \hat{y}_i)^2, \quad (27.17)$$

y sería, como función de la matriz  $H$ ,

$$\hat{\epsilon}'\hat{\epsilon} = y'(I - H)(I - H)y = y'(I - H)y. \quad (27.18)$$

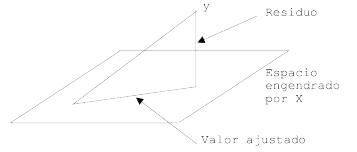


Figura 27.4: Expresamos la observación  $y$  como suma ortogonal de una parte sistemática más un residuo.

Veamos una interpretación geométrica que nos ayude a entender qué son los estimadores mínimo cuadráticos que utilizamos. Estamos minimizando  $(y - X\beta)'(y - X\beta)$ . Si vemos la figura 27.4 el valor de  $\beta$  que nos da el mínimo coincide con el punto que nos da la proyección ortogonal de  $y$  sobre el plano que viene engendrado por las columnas de la matriz  $X$ . De este modo es claro que

$$(y - \bar{y}1_n)'(y - \bar{y}1_n) = (y - X\hat{\beta})'(y - X\hat{\beta}) + (\hat{y} - \bar{y}1_n)'(\hat{y} - \bar{y}1_n). \quad (27.19)$$

o de otro modo la ecuación anterior la podemos expresar como

$$\sum_{i=1}^n (y_i - \bar{y})^2 = \sum_{i=1}^n (y_i - \hat{y}_i)^2 + \sum_{i=1}^n (\hat{y}_i - \bar{y})^2. \quad (27.20)$$

Las sumas de cuadrados que acabamos de considerar reciben la siguiente denominación:

#### Suma de cuadrados total

$$SS(Total) = (y - \bar{y}1_n)'(y - \bar{y}1_n) = \sum_{i=1}^n (y_i - \bar{y})^2. \quad (27.21)$$

#### Suma de cuadrados del error

$$SS(Error) = (y - X\hat{\beta})'(y - X\hat{\beta}) = \sum_{i=1}^n (y_i - \hat{y}_i)^2. \quad (27.22)$$

#### Suma de cuadrados de la regresión

$$SS(Regression) = (\hat{y} - \bar{y}1_n)'(\hat{y} - \bar{y}1_n) = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2. \quad (27.23)$$

## 27.6 Algunos casos particulares

Supongamos en primer lugar la situación en que no tenemos ningún predictor. Esto es, nuestro modelo es  $Y = \mu + \epsilon$ . En este caso se tiene que la matriz de diseño  $X = 1_n = (1, \dots, 1)', X'X = n$  y, finalmente,  $\hat{\beta} = (X'X)^{-1}X'y = \frac{1}{n}1_n'y = \hat{y}$ .

El segundo ejemplo que podemos considerar sería con una sola variable predictora o lo que es lo mismo, el modelo de regresión lineal simple. En este caso, tendremos

$$\begin{bmatrix} Y_1 \\ \vdots \\ Y_n \end{bmatrix} = \begin{bmatrix} 1 & x_1 \\ \vdots & \vdots \\ 1 & x_n \end{bmatrix} \begin{bmatrix} \beta_0 \\ \beta_1 \end{bmatrix} + \begin{bmatrix} \epsilon_1 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

Notemos que podemos hacer  $Y_i = \beta_0 + \beta_1\bar{x} + \beta_1(x_i - \bar{x}) + \epsilon_i$ . Nuestra matriz de diseño sería

$$X = \begin{bmatrix} 1 & x_1 - \bar{x} \\ \vdots & \vdots \\ 1 & x_n - \bar{x} \end{bmatrix}$$

con

$$X'X = \begin{bmatrix} n & 0 \\ 0 & \sum_{i=1}^n (x_i - \bar{x})^2 \end{bmatrix}$$

Finalmente se comprueba sin dificultad que

$$\hat{\beta} = \frac{\sum_{i=1}^n (x_i - \hat{x})y_i}{\sum_{i=1}^n (x_i - \hat{x})^2} \quad (27.24)$$

## 27.7 Verosimilitud

Dados los datos  $(x_i, y_i)$  con  $i = 1, \dots, n$  la verosimilitud de  $y = (y_1, \dots, y_n)'$  vendría dada por

$$L(\beta, \sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma^n} \exp\left\{-\frac{1}{2\sigma^2} (y - X\beta)'(y - X\beta)\right\} \quad (27.25)$$

y la logverosimilitud sería

$$l(\beta, \sigma) = \frac{n}{2} \log(2\pi) - n \log \sigma - \frac{1}{2\sigma^2} (y - X\beta)'(y - X\beta). \quad (27.26)$$

El estimador máximo verosímil de  $\beta$  se obtiene maximizando cualquiera de las dos funciones anteriores. Es obvio que el máximo respecto de  $\beta$  se obtiene como el valor que minimiza  $(y - X\beta)'(y - X\beta)$ , en definitiva, que los estimadores máximo verosímiles no son más que los estimadores mínimo cuadráticos.

## 27.8 Distribución muestral de $\hat{\beta}$

Hemos visto que  $\hat{\beta} = (X'X)^{-1}X'Y$ . Aplicando propiedades simples de la media tenemos que

$$E\hat{\beta} = (X'X)^{-1}X'(EY) = \beta, \quad (27.27)$$

o, lo que es lo mismo, que  $\hat{\beta}$  es un estimador **insesgado** de  $\beta$ , el estimador tiene por vector de medias el vector de parámetros que estima. Es una buena propiedad. La matriz de covarianzas del error se obtiene fácilmente como

$$var(\hat{\beta}) = (X'X)^{-1}X'(\sigma^2 I)X(X'X)^{-1} = (X'X)^{-1}\sigma^2. \quad (27.28)$$

Esta matriz de covarianzas depende de la varianza desconocida del error  $\sigma^2$ . Si estimamos esta varianza tendremos un estimador de dicha matriz de covarianzas. Se puede probar que

$$E[SS(Error)] = (n - p)\sigma^2 \quad (27.29)$$

de donde, un estimador insesgado para la varianza  $\sigma^2$  viene dado por

$$\hat{\sigma}^2 = \frac{\hat{\epsilon}'\hat{\epsilon}}{n - p}. \quad (27.30)$$

Ya podemos estimar  $var(\hat{\beta})$ . Este estimador sería

$$\widehat{var}(\hat{\beta}) = (X'X)^{-1}\hat{\sigma}^2. \quad (27.31)$$

Si  $(X'X)^{-1} = [a_{ij}]_{i,j=1,\dots,p}$  entonces el estimador de la varianza de  $\hat{\beta}_i$ ,  $var(\hat{\beta}_i)$ , sería  $a_{ii}\hat{\sigma}^2$ . Recordemos que  $\hat{\beta}_i$  es un estimador insesgado de  $\beta_i$  y por lo tanto su varianza coincide con su error cuadrático medio. Finalmente el error estándar de  $\hat{\beta}_i$ , es decir, su desviación típica (raíz cuadrada de su varianza) sería

$$\widehat{SE}(\hat{\beta}_i) = \sqrt{a_{ii}}\hat{\sigma}. \quad (27.32)$$

Realmente de  $\hat{\beta}$  sabemos más cosas:  $\hat{\beta} = (X'X)^{-1}X'Y$  y puesto que  $Y \sim N_n(X\beta, \sigma^2 I_{n \times n})$  entonces, por propiedades básicas de la distribución normal multivariante se tiene que

$$\hat{\beta} \sim N_p(\beta, (X'X)^{-1}\sigma^2). \quad (27.33)$$

Una vez realizado el ajuste con la función *summary* podemos observar los valores estimados de  $\sigma$  (etiquetado como *Residual standard error*) y los errores estándar de  $\hat{\beta}_i$ .

## 27.9 Bondad de ajuste

Hemos supuesto una relación lineal entre la media de la variable respuesta y las variables predictoras. Asumiendo la relación hemos considerado una función objetivo y minimizando dicha función hemos obtenido los estimadores mínimo cuadráticos. Sin embargo, la primera pregunta que hay que responder es: ¿tenemos un ajuste razonable? La respuesta se da utilizando medidas que comparan los valores observados con las predicciones asumiendo el modelo, es decir, comparando  $y_i$  con  $\hat{y}_i$  para los distintos datos. En concreto, con diferencia la más utilizada es el **coeficiente de determinación** que se denota por  $R^2$  y se define como

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y}_i)^2} = 1 - \frac{SS(Error)}{SS(Total)}. \quad (27.34)$$

Es habitual denominar a  $\sum_{i=1}^n (y_i - \hat{y}_i)^2$ , **suma de cuadrados del error** mientras que a  $SS(Regresion) = \sum_{i=1}^n (\hat{y}_i - \bar{y})^2$  se le llama **suma de cuadrados de la regresión**. Tenemos pues que

$$R^2 = \frac{\sum_{i=1}^n (\hat{y}_i - \bar{y})^2}{\sum_{i=1}^n (y_i - \bar{y}_i)^2} = \frac{SS(Regresion)}{SS(Total)}. \quad (27.35)$$

El ajuste que estamos realizando se supone que será tanto mejor cuanto más pequeña sea  $SS(Error)$ . Tampoco sería natural que  $SS(Error)$  fuera nula pues sería tanto como asumir que los distintos valores aleatorios son iguales a su media. Notemos que  $SS(Total)$  es una cuantificación de la variabilidad de los distintos  $y_i$  sin tener en cuenta las variables predictoras mientras que  $SS(Error)$  nos cuantifica la variación residual después de utilizar las variables predictoras. Es de esperar que un mejor ajuste vaya acompañado de un valor de  $SS(Error)$  pequeño en relación con  $SS(Total)$ . Esa es la idea del coeficiente de determinación. Toma valores entre 0 y 1 y cuanto más cerca de 1 mejor es el ajuste.

Tiene un pequeño inconveniente y es que no tiene en cuenta el número de variables predictoras que estamos utilizando para predecir la variable respuesta. Una pequeña modificación de  $R^2$  para incorporar esta información es el **coeficiente de determinación ajustado** que podemos

denotar  $R^2$ -ajustado y se define como

$$R^2 - \text{ajustado} = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2 / (n - p)}{\sum_{i=1}^n (y_i - \bar{y})^2 / (n - 1)}, \quad (27.36)$$

donde suponemos que tenemos  $p - 1$  variables predictoras.

## 27.10 Valoración de las hipótesis del modelo

Un modelo de regresión lineal múltiple supone, como hemos visto, varias hipótesis. Es necesario valorar lo razonables, lo asumibles que son estas hipótesis. Las hipótesis del modelo que vamos a valorar son las siguientes:

1. ¿Tenemos errores independientes, con la misma varianza y con distribución normal? Esto es, nos preguntamos si es asumible la hipótesis  $\epsilon \sim N_n(0, \sigma^2 I_{n \times n})$ .
2. Asumimos que  $E[Y_i | x_i] = \beta_0 + \beta_1 x_{i1} + \dots + \beta_{p-1} x_{ip-1}$ .

Los errores  $\epsilon$  no son directamente observables. Observamos los residuos  $\hat{\epsilon} = y - \hat{y}$  que no es lo mismo. Las propiedades de ambos vectores son distintas. En particular, estamos asumiendo que  $\text{var}(\epsilon) = \sigma^2 I_{n \times n}$ . Sin embargo, esta afirmación no es cierta para los residuos observados  $\hat{\epsilon}$ . Notemos que

$$\hat{y} = X(X'X)^{-1}X'y = Hy.$$

De modo que

$$\hat{\epsilon} = y - \hat{y} = (I - H)y = (I - H)X\beta + (I - H)\epsilon = (I - H)\epsilon.$$

La tercera igualdad anterior es consecuencia de que  $HX\beta = X\beta$  porque  $H$  es la matriz de proyección sobre el espacio engendrado por las columnas de  $X$  y  $X\beta$  está en este espacio por lo que la proyección es el propio punto. Notemos que  $I - H$  también es una matriz de proyección (sobre el espacio ortogonal al engendrado por las columnas de  $X$ ) de modo que  $(I - H)^2 = I - H$ . Aplicando esta propiedad se tiene que

$$\text{var}(\hat{\epsilon}) = \text{var}(I - H)\epsilon = (I - H)\sigma^2, \quad (27.37)$$

ya que  $\text{var}(\epsilon) = \sigma^2 I_{n \times n}$ . Vemos pues que, aunque asumimos que los errores  $\epsilon$  son incorrelados y con la misma varianza, esto no es cierto para los residuos  $\hat{\epsilon}$ .

### 27.10.1 Homogeneidad de la varianza

La mera observación de los residuos sin considerar su posible asociación con otra variable no nos proporciona información sobre si la varianza de los mismos es constante. Hemos de considerarlos en relación con otras variables. Es habitual considerar un diagrama de puntos de los residuos  $\hat{\epsilon}$  como función de las predicciones  $\hat{y}$ . Cuando la varianza es constante debemos de observar los residuos dispersos de un modo aleatorio respecto del eje de abscisas. También podemos ver un comportamiento no aleatorio alrededor del eje de abscisas cuando la parte estructural del modelo no es lineal, es decir, cuando no se verifica que  $EY = X\beta$ .

**Ejemplo 27.3** Los datos que vamos a utilizar para valorar las hipótesis del modelo son los datos tamidata::gse44456. Se pretende estudiar la relación que liga el nivel de expresión observado del gen el pH. El siguiente diagrama de puntos muestra en abscisas las predicciones y en ordenadas los residuos. No parece en principio que no podamos asumir una varianza constante.

```
y = exprs(gse44456)[19254,]
x = pData(gse44456)[, "pH"]
```

Construimos un *data.frame* para su uso posterior.

```
gse44456_19254 = data.frame(y, x)
```

Hacemos el ajuste lineal.

```
gen19254.lm = lm(y ~ x, data=gse44456_19254)
```

Notemos que elegimos el primer dibujo (figura 27.5).

```
## pdf
## 2
```

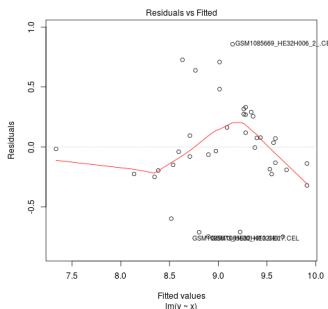


Figura 27.5: Predicciones frente a residuos del ajuste.

Cuando no tenemos una varianza constante una opción es transformar las variables. Si  $y$  es la variable original y  $h(y)$  la transformada queremos determinar  $h$  de modo que la transformada tenga varianza constante. Transformaciones habituales que podemos valorar son la raíz cuadrada o el logaritmo de la variable respuesta.

¿Qué tipo de representación cabe esperar cuando la varianza no es constante? Veamos distintas representaciones de los residuos frente a los valores ajustados correspondiendo a varianza constante, varianzas no constantes y situaciones en que no hay linealidad.

**Ejemplo 27.4 (Hipótesis del modelo de regresión)** Ilustramos cómo observaríamos residuos cuya varianza es constante. En la figura 27.6 tenemos 50 posibles residuos en donde la varianza es constante.

```
## pdf
## 2
```

En la figura 27.7 tenemos como abscisas valores que van de 1 a 50 en incrementos unitarios. El valor de residuo que generamos tienen varianza creciente. En concreto vamos multiplicando el valor con distribución  $N(0, 1)$  por una constante  $c$ . La varianza es del orden del cuadrado del valor por el que multiplicamos.

```
## pdf
## 2
```

En la figura 27.8 los valores por los que multiplicamos son  $1, \sqrt{2}, \sqrt{3}, \dots, \sqrt{50}$  por lo que las varianzas se incrementan mucho menos de una abscisa a la siguiente.

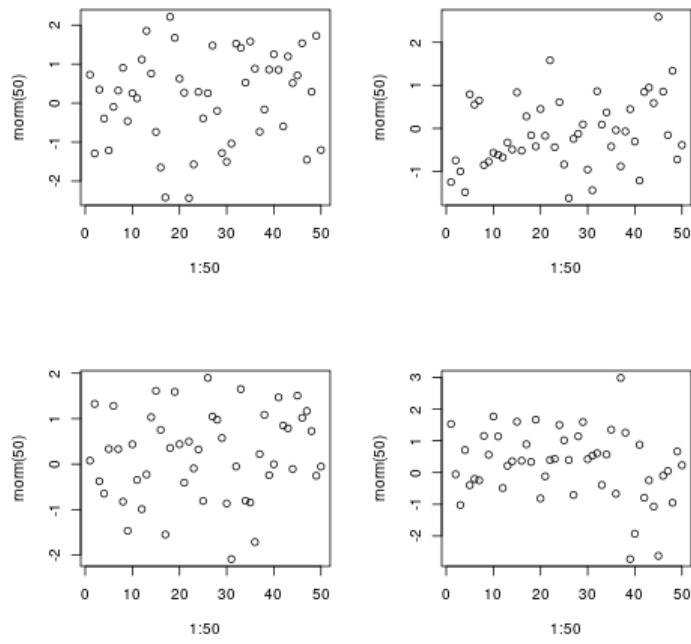


Figura 27.6: Residuos cuando se verifica la homogeneidad de la varianza.

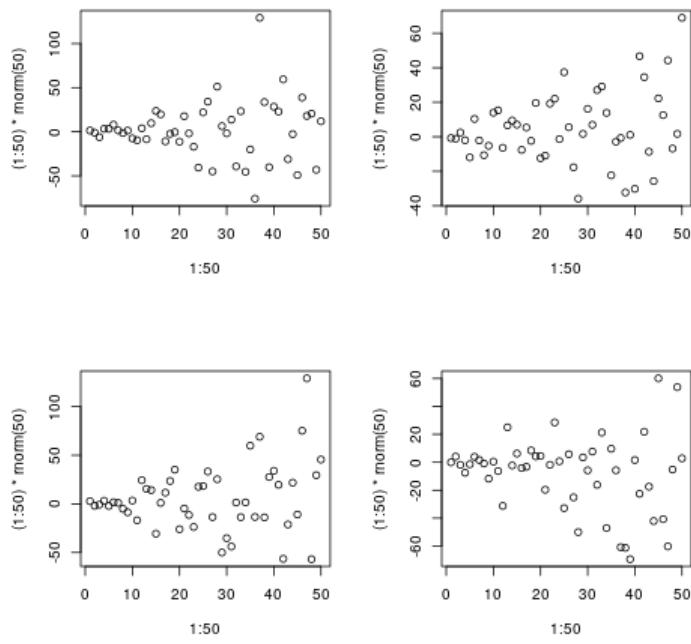


Figura 27.7: Residuos con varianza creciente con la variable predictora.

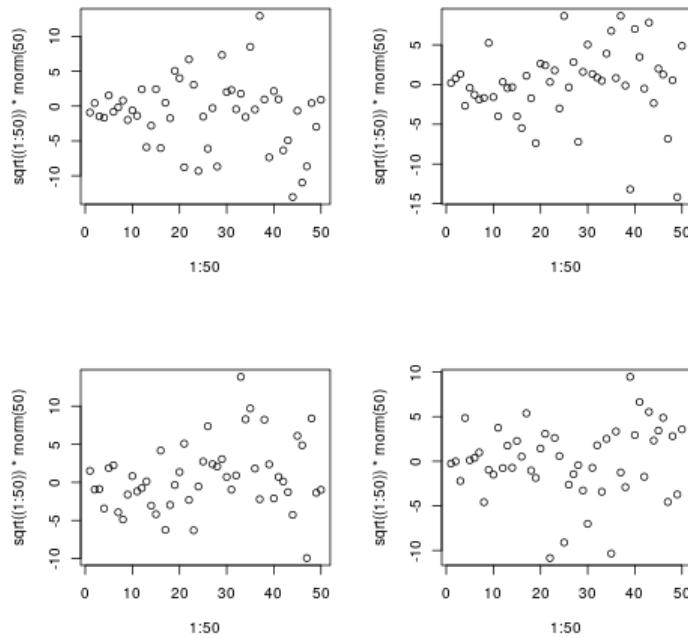


Figura 27.8: Residuos con varianza creciente con la variable predictora.

```
## pdf
## 2
```

Finalmente en la figura 27.9 mostramos un ejemplo de un residuo no lineal. Es un caso en que no podemos suponer que la media es una función lineal de los predictores.

```
## pdf
## 2
```

### 27.10.2 Normalidad

La siguiente hipótesis a valorar es la normalidad de los errores. La herramienta gráfica más habitual es el *dibujo q-q* o la representación cuantil-cuantil. Ordenamos los residuos y representamos los residuos ordenados en función de  $\Phi^{-1}(\frac{i}{n+1})$  para  $i = 1, \dots, n$ . Si los residuos tienen una distribución normal entonces deben estar alineados. El histograma no es muy adecuado para ver la normalidad de los residuos.

**Ejemplo 27.5 (Dibujos cuantil-cuantil)** Para los datos representamos un dibujo q-q. En la figura que sigue mostramos cómo hacerlo después de ajustar el modelo y utilizando la función plot (realmente estamos utilizando plot.lm).

```
## pdf
## 2
```

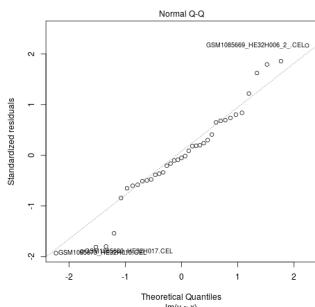


Figura 27.10: Dibujo cuantil-cuantil.

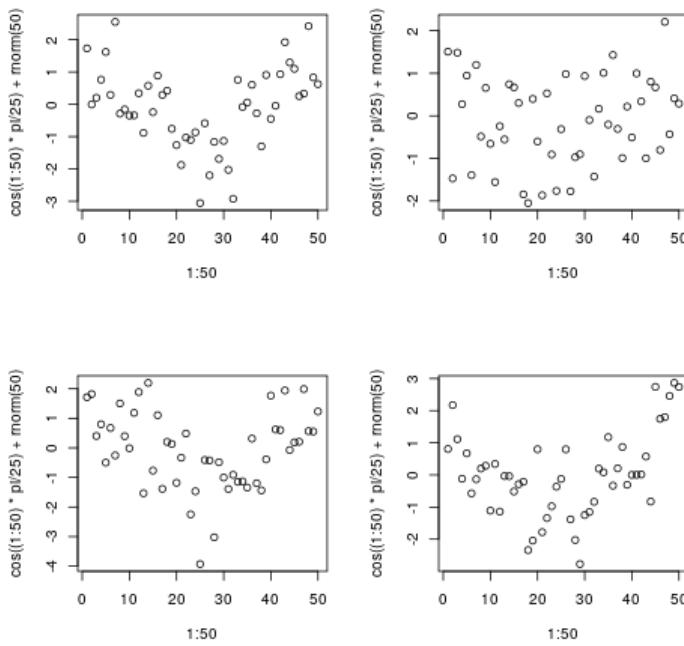


Figura 27.9: Residuo no lineal con la variable predictora.

En la figura 27.11 aparece un dibujo q-q utilizando las funciones qqnorm que construye el dibujo y qqline que añade una línea uniendo el primer y tercer cuartil. Como vemos es el mismo dibujo.

```
## pdf
## 2
```

Hemos visto cómo es una dibujo q-q cuando tenemos normalidad. Pero: ¿y qué pasa cuando no tenemos normalidad. Esto es lo interesante saber qué tenemos que buscar para detectar que los residuos no siguen una distribución normal.

**Ejemplo 27.6 (Dibujos cuantil-cuantil y error no normal)** Veamos cómo se modifican los dibujos q-q con otras distribuciones de probabilidad. En esta nota consideraremos cuatro distribuciones de probabilidad. En la siguiente aparecen las densidades de los modelos considerados. La primera es la correspondiente a la distribución normal. Luego tenemos la lognormal, la densidad de una Cauchy y la densidad uniforme. La lognormal se toma como ejemplo de distribución asimétrica, la Cauchy como ejemplo de una distribución con las colas más largas que la normal y finalmente la uniforme como ejemplo de distribución con las colas más cortas. Las funciones de densidad correspondientes aparecen en la figura 27.12.

```
## pdf
## 2
```

En la figura 27.13 tenemos nueve dibujos q-q realizados con datos simulados con una distribución normal.

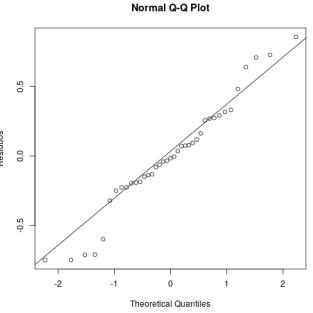


Figura 27.11: Dibujo cuantil-cuantil utilizando qqnorm y qqline.

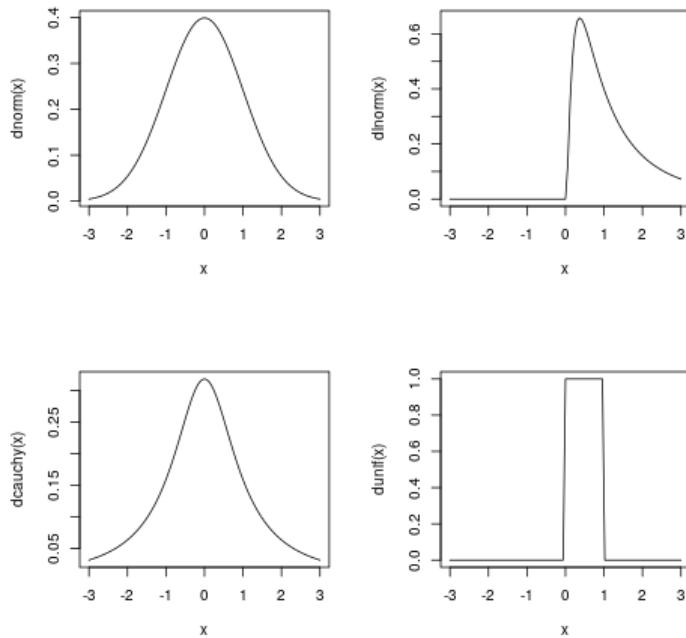


Figura 27.12: Densidades normal, lognormal, Cauchy y uniforme.

```
## pdf
## 2
```

*La figura 27.14 que sigue muestra el dibujo q-q con datos simulados correspondientes a la distribución lognormal.*

```
## pdf
## 2
```

*La figura 27.15 que sigue muestra el dibujo q-q con datos simulados correspondientes a la distribución de Cauchy.*

```
## pdf
## 2
```

*La figura 27.16 que sigue muestra el dibujo q-q con datos simulados correspondientes a la distribución de uniforme.*

```
## pdf
## 2
```

Como vemos no es simple la interpretación de estos dibujos. Por ejemplo, no es fácil diferenciar entre una distribución con colas más largas de una situación en donde tenemos observaciones anómalas. ¿Cómo hacerlo? Si quitamos las observaciones extremas y aparecen otras estamos en una distribución con colas largas.

Si tenemos muestras grandes el problema de la no normalidad se alivia. El problema de la no normalidad es mayor para colas largas que

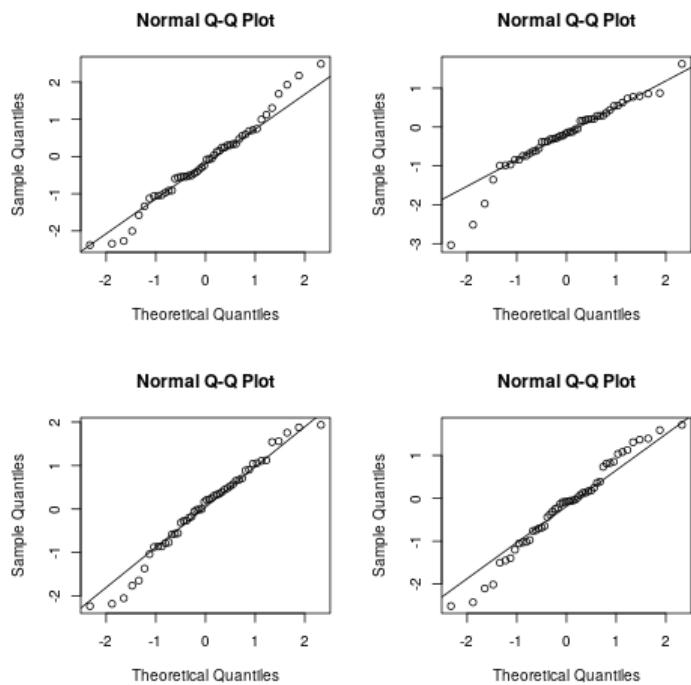


Figura 27.13: Densidad normal de los residuos.

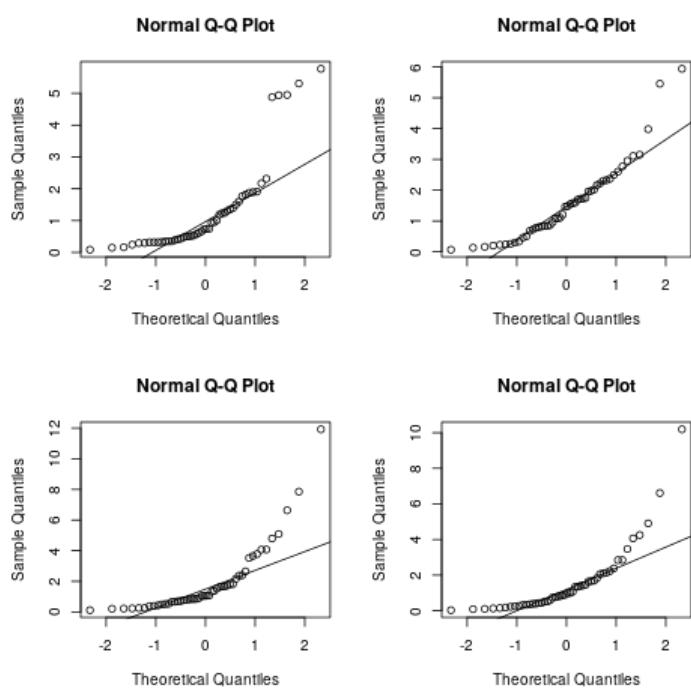


Figura 27.14: Densidad lognormal de los residuos.

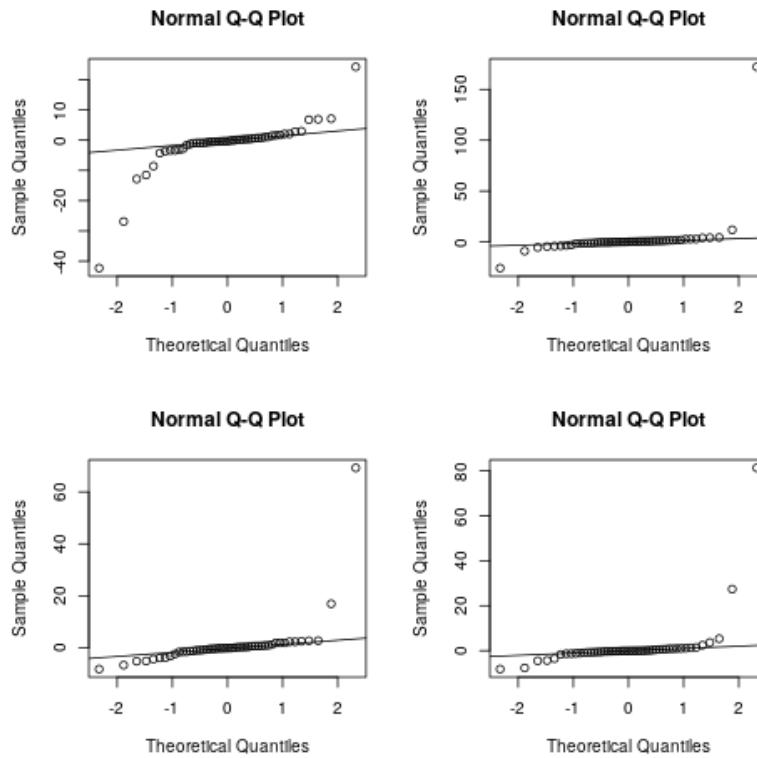


Figura 27.15: Densidad Cauchy de los residuos.

para colas más cortas. La asimetría se resolvería transformando los datos.

También podemos usar un test de normalidad. Un test de normalidad es un cualquier test estadístico donde la hipótesis nula es que los datos observados (en nuestro caso los residuos observados) proceden de una distribución normal. Una opción que viene implementada con la función *shapiro.test* es el test de Shapiro-Wilk.

**Ejemplo 27.7 (Residuos y test de normalidad de Shapiro-Wilks)** *Aplicamos un test de Shapiro-Wilks a los residuos observados en el ajuste de los datos gen19254. Vemos que el p-valor observado es muy grande y no podemos rechazar la normalidad de los residuos.*

```
head(residuals(gen19254.lm)) #Mostramos los primeros

##      GSM1085665_HE32H001.CEL
##                  0.31615334
##      GSM1085666_HE32H003.CEL
##                 -0.01649683
##      GSM1085667_HE32H004.CEL
##                  0.07792104
##      GSM1085668_HE32H005.CEL
##                  0.27348758
##  GSM1085669_HE32H006_2_.CEL
##                  0.85598123
##  GSM1085670_HE32H007_2_.CEL
```

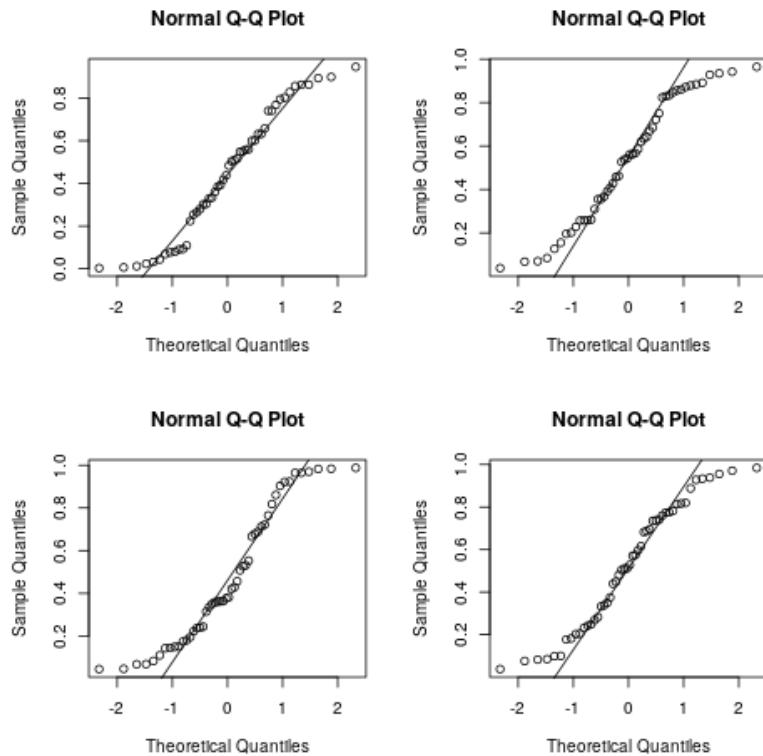


Figura 27.16: Densidad uniforme de los residuos.

```
##          -0.70969598
shapiro.test(residuals(gen19254.lm))
##
##  Shapiro-Wilk normality test
##
## data:  residuals(gen19254.lm)
## W = 0.96264, p-value = 0.2182
```

### 27.10.3 Incorrelación de los errores

Estamos asumiendo que **los errores son incorrelados**. La correlación entre los datos pueden venir de que han sido observados próximos bien en el tiempo bien en el espacio. ¿Cómo contrastar si los residuos son incorrelados?

Un test de la hipótesis de incorrelación es el test de Durbin-Watson que utiliza el siguiente estadístico

$$DW = \frac{\sum_{i=2}^n (\hat{\epsilon}_i - \hat{\epsilon}_{i-1})^2}{\sum_{i=1}^n \hat{\epsilon}_i^2}, \quad (27.38)$$

bajo la hipótesis nula de incorrelación la distribución es una combinación lineal de distintas distribuciones  $\chi^2$ .

**Ejemplo 27.8** Vamos a aplicar el test de Durbin-Watson de correlación serial.

```
library(lmtest)
```

Aplicamos el test de Durbin-Watson.

```
dwtest(gen19254.lm)

##
##  Durbin-Watson test
##
## data: gen19254.lm
## DW = 2.3323, p-value = 0.8575
## alternative hypothesis: true autocorrelation is greater than 0
```

Vemos que no tenemos autocorrelación.

#### 27.10.4 Observaciones anómalas y observaciones influyentes

Algunas observaciones no ajustan bien al modelo y son llamadas **observaciones anómalas**. Otras observaciones influyen mucho en el ajuste y lo modifican de un modo substantivo. Estas últimas reciben el nombre de **observaciones influyentes**. Una observación dada puede ser o bien anómala o bien influyente o ambas cosas.

Empecemos estudiando qué se entiende por **influencia** de una observación. Llamaremos a  $h_i = H_{ii}$ , esto es, el valor en la posición  $(i, i)$  de la matriz  $H$  la **influencia** (*leverage* en inglés). Notemos que  $\text{var}(\hat{\epsilon}_i) = \sigma^2(1 - h_i)$ . En consecuencia, una influencia alta supone una varianza del correspondiente residuo baja. Forzamos al ajuste a que esté próximo a  $y_i$ . Se tiene que  $\sum_{i=1}^p h_i = p$ . En consecuencia el valor medio de los  $h_i$  es  $p/n$ . Como una regla simple de aplicar si la influencia  $h_i$  es mayor que  $2p/n$  debemos de observar el dato con atención. Buscamos valores grandes de las influencias.

**Ejemplo 27.9** Calculamos las influencias el ajuste *gen19254.lm*. En la siguiente figura aparece una representación de las influencias respecto de los residuos estandarizados. Calculamos las influencias.

```
gen19254.inf = influence(gen19254.lm)
head(gen19254.inf$hat)

##      GSM1085665_HE32H001.CEL
##                      0.02890237
##      GSM1085666_HE32H003.CEL
##                      0.31508019
##      GSM1085667_HE32H004.CEL
##                      0.03777124
##      GSM1085668_HE32H005.CEL
##                      0.02890237
##      GSM1085669_HE32H006_2_.CEL
##                      0.02611433
##      GSM1085670_HE32H007_2_.CEL
##                      0.02769529
```

Y las representamos en la figura 27.17.

```
## pdf
## 2
```

Otra posibilidad es para encontrar observaciones anómalas consiste en trabajar con los residuos estudiantizados. Veamos su definición. Notemos que  $\text{var}(\hat{\epsilon}_i) = \sigma^2(1 - h_i)$  lo que sugiere tomar

$$r_i = \frac{\hat{\epsilon}_i}{\hat{\sigma}\sqrt{1 - h_i}} \quad (27.39)$$

Estos son los residuos estudiantizados. Si el modelo que asumimos es correcto entonces la varianza de estos residuos es uno y son aproximadamente incorrelados. Notemos que la estudiantización corrige las varianzas desiguales de los residuos cuando las varianzas de los errores son iguales entre si. En otro caso esto no es cierto. Si las varianzas de los errores no son iguales (tenemos heterocedasticidad) entonces la estudiantización no hace homogéneas las varianzas de los residuos. Algunos autores (y programas) tienden a usar en lugar de los residuos originales, los residuos estudiantizados.

**Una observación anómala** es una observación que no se ajusta a nuestro modelo. Hemos de protegernos frente a este tipo de puntos. Un test para observaciones anómalas es útil si nos permite distinguir entre observaciones que son realmente anómala y aquellas que simplemente tienen un residuo grande aunque no excepcionalmente grande.

Para detectar este tipo de puntos, lo que hacemos es excluir el punto  $i$ -ésimo y ajustamos el modelo sin ese punto obteniendo los estimadores  $\hat{\beta}_{(i)}$  y  $\hat{\sigma}_{(i)}^2$ . Tendremos para el punto  $i$ -ésimo la siguiente estimación

$$\hat{y}_{(i)} = x_i' \hat{\beta}_{(i)}. \quad (27.40)$$

Si el valor  $y_i - \hat{y}_{(i)}$  es grande entonces el caso  $i$  es una observación anómala. Con objeto de valorar si estos nuevos residuos son anormales hemos de estandarizarlos. Notemos que

$$\widehat{\text{var}}(y_i - \hat{y}_{(i)}) = \hat{\sigma}_{(i)}^2 (1 + x_i'(X_{(i)}' X_{(i)})^{-1} x_i)$$

de modo que podemos definir los residuos jackknife como

$$t_i = \frac{y_i - \hat{y}_{(i)}}{\sqrt{\widehat{\text{var}}(y_i - \hat{y}_{(i)})}} = \frac{y_i - \hat{y}_{(i)}}{\hat{\sigma}_{(i)} \sqrt{1 + x_i'(X_{(i)}' X_{(i)})^{-1} x_i}} \sim t_{n-p-1},$$

asumiendo que el modelo es correcto. Se tiene la siguiente expresión

$$t_i = \frac{\hat{\epsilon}_i}{\hat{\sigma}_{(i)} \sqrt{1 - h_i}} = r_i \left( \frac{n - p - 1}{n - p - r_i^2} \right)^{1/2}.$$

Puesto que cada uno de los  $t_i$  sigue una distribución conocida podemos contrastar si tenemos una observación anómala. En principio el procedimiento sería simplemente fijar un nivel de significación  $\alpha$  y determinar el percentil  $1 - \frac{\alpha}{2}$  de una distribución t de Student con  $n - p - 1$  grados de libertad. Si denotamos el percentil como  $t_{n-p-1, 1-\frac{\alpha}{2}}$  entonces residuos que no estén en el intervalo  $[-t_{n-p-1, 1-\frac{\alpha}{2}}, t_{n-p-1, 1-\frac{\alpha}{2}}]$  serían sospechosos. Esto no es adecuado. Porque estamos analizando  $n$  residuos

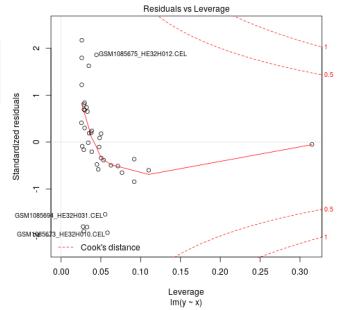


Figura 27.17: Influencias para el ajuste gen19254.lm.

estudentizados. Con uno sólo sí que sería aplicable el razonamiento. Tenemos un problema de muchos tests simultáneamente considerados. Si aplicamos la corrección de Bonferroni tendríamos que corregir el nivel de significación  $\alpha$  y trabajar con  $\alpha/n$ . Por tanto, consideramos como sospechosos aquellos residuos estandarizados que estén fuera del intervalo  $[-t_{n-p-1,1-\frac{\alpha}{2n}}, t_{n-p-1,1-\frac{\alpha}{2n}}]$ .

**Ejemplo 27.10 (Residuos estudiantizados)** Calculamos para los datos gen19254 los residuos estudiantizados con la función `rstudent`. La pregunta que nos hacemos es si alguno de estos residuos es muy grande o muy pequeño. Comparamos el módulo de los residuos estandarizados con  $t_{n-p-1,1-\frac{\alpha}{2n}}$ .

```
gen19254.rs = rstudent(gen19254.lm)
```

Comparamos los residuos con el percentil correspondiente.

```
table(abs(gen19254.rs) > qt(1-.05/(50*2), 44))

## 
## FALSE
##      39
```

Un punto influyente es aquel que cuando lo quitamos causa una modificación importante del ajuste. Un punto influyente puede ser o no una observación anómala, puede tener o no una alta influencia pero tenderá a tener al menos una de las dos propiedades. El estadístico de Cook es una de las maneras de medir la influencia. Se define como

$$D_i = \frac{(\hat{y}_i - \hat{y}_{(i)})'(\hat{y}_i - \hat{y}_{(i)})}{p\hat{\sigma}^2} = \frac{1}{p} r_i^2 \frac{h_i}{1-h_i}. \quad (27.41)$$

**Ejemplo 27.11** Con los datos gen19254 representamos las distancias de Cook y las obtenemos utilizando la función `Cooks.distance`. Lo tenemos en la figura 27.18.

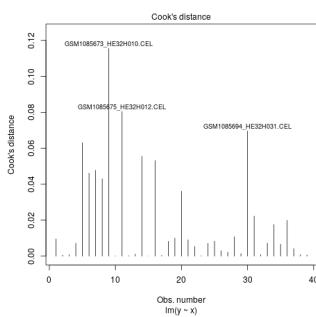


Figura 27.18: Distancias de Cook para el ajuste gen19254.lm.

```
png(paste0(dirTamiFigures, "ModLin49.png"))
plot(gen19254.lm, which = 4)
dev.off()

## pdf
## 2
```

También podemos ver los valores de la distancia de Cook.

```
head(cooks.distance(gen19254.lm))
```

```
##      GSM1085665_HE32H001.CEL
##      0.0095804543
##      GSM1085666_HE32H003.CEL
##      0.0005716436
##      GSM1085667_HE32H004.CEL
##      0.0007746346
##      GSM1085668_HE32H005.CEL
##      0.0071691187
```

```
## GSM1085669_HE32H006_2_.CEL
##          0.0630919910
## GSM1085670_HE32H007_2_.CEL
##          0.0461454249
```

## 27.11 Inferencia sobre el modelo

Hemos formulado un modelo probabilístico en donde relacionamos la variable respuesta con una serie de variables predictoras. Es claro que el experimentador introduce en el modelo como variables predictoras variables que a priori sospecha que pueden ser relevantes a la hora de predecir. Esto no quiere decir que luego podamos prescindir de alguna o algunas de ellas. Bien porque se demuestra que dicha variable no es relevante o bien porque la información que contiene esa variable predictora está contenida en las otras.

Supongamos que nos planteamos el siguiente contraste de hipótesis:  $H_0 : \beta_{i_1} = \dots = \beta_{i_r} = 0$  frente a la alternativa  $H_1 : \text{No } H_0$ . Si un coeficiente determinado  $\beta_i$  es nulo entonces la variable respuesta  $Y$  no dependería de la variable asociada a dicho coeficiente. En definitiva, la hipótesis nula considerada se podría formular diciendo que la variable  $Y$  no depende de las variables  $x_{i_1}, \dots, x_{i_r}$ . ¿Cómo contrastar las hipótesis indicadas? Se puede hacer mediante el test del cociente de verosimilitudes, es decir, utilizando el estadístico

$$\Lambda = \frac{\max_{(\beta, \sigma) \in \Omega_0} L(\beta, \sigma)}{\max_{(\beta, \sigma) \in \Omega} L(\beta, \sigma)} \quad (27.42)$$

siendo  $L(\beta, \sigma) = \frac{1}{(2\pi)^{\frac{n}{2}} \sigma^n} \exp\left\{-\frac{1}{2\sigma^2} (y - X\beta)'(y - X\beta)\right\}$ ,  $\Omega_0 = \{(\beta, \sigma) \in \mathbb{R}^p \times (0, +\infty) : \beta_{i_1} = \dots = \beta_{i_r} = 0\}$  y  $\Omega = \mathbb{R}^p \times (0, +\infty)$ . Como es habitual en un test del cociente de verosimilitudes rechazaríamos la hipótesis nula si  $\Lambda$  es pequeño (menor que una cierta constante) que se prueba es equivalente a que

$$\frac{SS(\text{Error})_{\Omega_0} - SS(\text{Error})_{\Omega}}{SS(\text{Error})_{\Omega}}$$

sea grande (mayor que una cierta constante). Denotamos por  $SS(\text{Error})_{\Omega_0}$  la suma de cuadrados del error bajo la hipótesis nula y  $SS(\text{Error})_{\Omega}$  la suma de cuadrados sobre todo el espacio paramétrico. Bajo la hipótesis nula se verifica que

$$\frac{SS(\text{Error})_{\Omega_0} - SS(\text{Error})_{\Omega}}{\sigma^2 r} \sim \chi_r^2$$

y

$$\frac{SS(\text{Error})_{\Omega}}{\sigma^2(n-p)} \sim \chi_{n-p}^2$$

además ambas cantidades son independientes. Por ello se verifica que

$$F = \frac{(SS(\text{Error})_{\Omega_0} - SS(\text{Error})_{\Omega})/r}{(SS(\text{Error})_{\Omega})/(n-p)} \sim F_{r, n-p}.$$

De modo que rechazaremos la hipótesis nula de que  $H_0 : \beta_{i_1} = \dots = \beta_{i_r} = 0$  si

$$F > F_{r, n-p, 1-\alpha}$$

donde  $F_{r,n-p,1-\alpha}$  es el percentil  $1 - \alpha$  de una  $F$  con  $r$  y  $n - p$  grados de libertad.

¿Realmente depende la variable respuesta de alguna de las variables predictoras? Realmente nos estamos planteando la hipótesis de que todos los coeficientes, salvo el término constante  $\beta_0$ , valen cero, es decir, la hipótesis nula  $H_0 : \beta_1 = \dots = \beta_{p-1} = 0$ . En este caso tendremos que

$$F = \frac{((y - \mathbf{1}_n \bar{y})'(y - \mathbf{1}_n \bar{y}) - (y - X\hat{\beta})'(y - X\hat{\beta}))/(p-1)}{(y - X\hat{\beta})'(y - X\hat{\beta})/(n-p)} \sim F_{p-1,n-p}.$$

Como segundo caso tendríamos la situación en que contrastamos que un solo coeficiente vale cero, es decir, la hipótesis nula  $H_0 : \beta_i = 0$  frente a la alternativa  $H_1 : \beta_i \neq 0$ . Tenemos que bajo la hipótesis nula indicada

$$t_i = \frac{\hat{\beta}_i}{SE(\hat{\beta}_i)} \sim t_{n-p}$$

donde  $SE(\hat{\beta}_i)$  es el error estándar de  $\hat{\beta}_i$  y viene dado en ecuación 27.32. Se tiene, de hecho, que

$$F = t_i^2.$$

Rechazaremos la hipótesis nula si

$$|t_i| > t_{n-p,1-\frac{\alpha}{2}}$$

o bien si

$$F = t_i^2 > F_{1,n-p,1-\frac{\alpha}{2}}.$$

Ambos procedimientos son equivalentes como se puede ver fácilmente.

**Ejemplo 27.12 (Contrastes sobre los coeficientes)** *Cuando contrastamos la hipótesis de que cada coeficiente valga cero vemos que podemos rechazarlo para las dos variables predictoras. Realizamos un summary del modelo de regresión ajustado.*

```
summary(gen19254.lm)

##
## Call:
## lm(formula = y ~ x, data = gse44456_19254)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.7492 -0.1943 -0.0165  0.2621  0.8560
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -3.3143    1.5311 -2.165   0.0369
## x            1.9084    0.2356  8.100 1.02e-09
##
## (Intercept) *
## x            ***
## ---
## Signif. codes:
## 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3998 on 37 degrees of freedom
## Multiple R-squared:  0.6394, Adjusted R-squared:  0.6297
## F-statistic: 65.62 on 1 and 37 DF,  p-value: 1.024e-09
```

Podemos tener intervalos de confianza para cada uno de los coeficientes  $\beta_i$ . Teniendo en cuenta que  $\frac{\hat{\beta}_i - \beta_i}{SE(\hat{\beta}_i)} \sim t_{n-p}$  entonces el intervalo de confianza al nivel  $1 - \alpha$  para el coeficiente  $\beta_i$  sería

$$\hat{\beta}_i \pm t_{n-p, 1 - \frac{\alpha}{2}} SE(\hat{\beta}_i).$$

**Ejemplo 27.13 (Intervalos de confianza)** Calculamos los intervalos de confianza para los coeficientes de gen19254 con la función confint.

```
confint(gen19254.lm)
```

```
##           2.5 %      97.5 %
## (Intercept) -6.416593 -0.2121019
## x           1.431029  2.3857477
```

Supongamos que consideramos un vector de predictores  $x_0$  y pretendemos predecir la correspondiente media de la variable respuesta. Esta media viene dada por

$$E[Y|x_0] = x_0' \beta.$$

La estimación de esta media es

$$x_0' \hat{\beta},$$

que tiene varianza

$$var(x_0' \hat{\beta}) = x_0' (X' X)^{-1} x_0 \sigma^2.$$

Esta varianza la estimamos mediante

$$\widehat{var}(x_0' \hat{\beta}) = x_0' (X' X)^{-1} x_0 \hat{\sigma}^2.$$

Y el intervalo de confianza sería

$$x_0' \hat{\beta} \pm t_{n-p, 1 - \frac{\alpha}{2}} \hat{\sigma} \sqrt{x_0' (X' X)^{-1} x_0}.$$

Supongamos que, en lugar de predecir la media de la variable respuesta para un conjunto de predictores dados, pretendemos predecir la propia variable respuesta. Recordemos que según nuestro modelo tenemos

$$Y = x_0' \beta + \epsilon.$$

En consecuencia la predicción de la propia observación sería

$$x_0' \hat{\beta}$$

pero hay que considerar la varianza añadida por el error  $\epsilon$  de modo que la varianza al predecir  $Y$  dado  $x_0$  sería

$$var(x_0' \hat{\beta}) + \sigma^2$$

que estimaríamos como

$$\widehat{var}(x_0' \hat{\beta}) + \hat{\sigma}^2 = x_0' (X' X)^{-1} x_0 \hat{\sigma}^2 + \hat{\sigma}^2 = (x_0' (X' X)^{-1} x_0 + 1) \hat{\sigma}^2.$$

Finalmente tendremos que el intervalo de confianza para una observación  $Y$  que tiene predictores  $x_0$  es el siguiente

$$x_0' \hat{\beta} \pm t_{n-p, 1 - \frac{\alpha}{2}} \hat{\sigma} \sqrt{x_0' (X' X)^{-1} x_0 + 1}$$

**Ejemplo 27.14** Con los datos gen19254 consideramos cómo obtener las predicciones, 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 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 las predicciones para los propios datos.

```
head(predict(gen19254.lm))

##      GSM1085665_HE32H001.CEL
##              9.261931
##      GSM1085666_HE32H003.CEL
##              7.334459
##      GSM1085667_HE32H004.CEL
##              9.433686
##      GSM1085668_HE32H005.CEL
##              9.261931
##      GSM1085669_HE32H006_2_.CEL
##              9.147428
##      GSM1085670_HE32H007_2_.CEL
##              9.223763
```

En segundo lugar, obtenemos los intervalos de confianza para la predicción de la media.

```
head(predict(gen19254.lm, interval = "confidence"))

##                      fit      lwr
## GSM1085665_HE32H001.CEL 9.261931 9.124197
## GSM1085666_HE32H003.CEL 7.334459 6.879696
## GSM1085667_HE32H004.CEL 9.433686 9.276231
## GSM1085668_HE32H005.CEL 9.261931 9.124197
## GSM1085669_HE32H006_2_.CEL 9.147428 9.016505
## GSM1085670_HE32H007_2_.CEL 9.223763 9.088936
##                      upr
## GSM1085665_HE32H001.CEL 9.399665
## GSM1085666_HE32H003.CEL 7.789221
## GSM1085667_HE32H004.CEL 9.591140
## GSM1085668_HE32H005.CEL 9.399665
## GSM1085669_HE32H006_2_.CEL 9.278350
## GSM1085670_HE32H007_2_.CEL 9.358590
```

Y finalmente obtenemos los intervalos de confianza para la predicción de la observación.

```
head(predict(gen19254.lm, interval = "prediction"))

## Warning in predict.lm(gen19254.lm, interval = "prediction"):
## predictions on current data refer to _future_ responses

##                      fit      lwr
## GSM1085665_HE32H001.CEL 9.261931 8.440140
## GSM1085666_HE32H003.CEL 7.334459 6.405385
```

```

## GSM1085667_HE32H004.CEL      9.433686 8.608361
## GSM1085668_HE32H005.CEL      9.261931 8.440140
## GSM1085669_HE32H006_2_.CEL   9.147428 8.326751
## GSM1085670_HE32H007_2_.CEL   9.223763 8.402454
##                                     upr
## GSM1085665_HE32H001.CEL      10.083722
## GSM1085666_HE32H003.CEL      8.263533
## GSM1085667_HE32H004.CEL      10.259011
## GSM1085668_HE32H005.CEL      10.083722
## GSM1085669_HE32H006_2_.CEL   9.968104
## GSM1085670_HE32H007_2_.CEL   10.045072

```

## 27.12 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. 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.

Dos son las aproximaciones al problema de la selección de variables. En la primera vamos comparando modelos sucesivos y, básicamente, consideramos si los modelos sucesivos que comparamos difieren significativamente como modelos. En la segunda se adopta una medida de calidad global del ajuste y se plantea el problema de determinar el modelo que optimiza ese criterio global.

### 27.12.1 Procedimientos que comparan modelos

Tres procedimientos se pueden considerar: **selección backward**, **selección forward** y **selección stepwise**. En el primer procedimiento, **selección backward** (selección hacia atrás), empezamos el proceso de selección con un modelo en que tenemos todas las variables consideradas a priori. Elegimos aquella que, cuando la quitamos, el p-valor resultante de comparar ambos modelos (completo y aquel que le falta la variable considerada) es mayor. Si este p-valor es mayor que un valor  $\alpha$  previamente especificado entonces eliminamos la variable del modelo. Y así continuamos hasta que al eliminar una variable el p-valor correspondiente sea menor que el  $\alpha$  elegido.

El procedimiento **selección forward** (o hacia adelante) empezamos con un modelo en el cual solamente tenemos la constante. Elegimos para entrar la variable tal que cuando se incorpora el modelo cambia lo más posible. Esto es, el p-valor es el más significativo. Si el p-valor de la comparación de los modelos es menor que  $\alpha$  (siempre elegido antes de iniciar el proceso de selección y no reajustado a lo largo del mismo) entonces la variable entra en el modelo. Continuamos hasta que el modelo no cambia significativamente.

El procedimiento más usado es el **selección stepwise** (hacia adelante y hacia atrás). Vistos los anteriores es de esperar cómo funciona este método. Después de la inclusión de una variable consideramos la posible exclusión de las variables que están en el modelo. Si es posible

eliminar una lo hacemos y nos planteamos la inclusión de la variable que produzca un cambio más significativo.

### 27.12.2 Procedimientos basados en criterios

Estamos ajustando un modelo de regresión lineal múltiple con algún propósito. Podemos fijarnos en una medida global de calidad del ajuste y buscar la selección de variables que nos da el óptimo según ese criterio. Una medida global puede ser AIC (Akaike Information Criterion) definido como

$$AIC = -2 \max \log verosimilitud + 2p.$$

A partir de la definición de esta medida global es claro que un valor pequeño indica un mejor ajuste.

**Ejemplo 27.15 (Selección de variables)** *Notemos que con en la función step podemos indicar con el argumento direction si queremos both (stepwise), o bien backward o bien forward. Utilizamos MASS::stepAIC.*

```
library(MASS)
stepAIC(gen19254.lm)

## Start:  AIC=-69.55
## y ~ x
##
##          Df Sum of Sq    RSS    AIC
## <none>            5.9155 -69.554
## - x     1    10.49 16.4058 -31.771
## 
## Call:
## lm(formula = y ~ x, data = gse44456_19254)
## 
## Coefficients:
## (Intercept)           x
##       -3.314        1.908
```

*Vemos que se conservan ambas variables predictoras.*

## Capítulo 28

# 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 y directo.

### 28.1 Verosimilitud

Sea  $y = (y_1, \dots, y_n)$  una realización del vector aleatorio  $Y = (Y_1, \dots, Y_n)$ . 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

$$\text{Verosimilitud}[f; y] = L(f; y) = f(y). \quad (28.1)$$

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

$$l[f; y] = \log f(y). \quad (28.2)$$

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  $\theta = (\theta_1, \dots, \theta_p)$  de modo que denotaremos  $f(y; \theta)$  o  $f_Y(y; \theta)$ . Al conjunto de valores posibles de  $\theta$  se le llama **espacio paramétrico** y lo denotaremos por  $\Theta$ . En este caso, la logverosimilitud es una función de  $\theta$  y denotaremos

$$\text{Verosimilitud}[\theta; y] = l(\theta; y) = \log f(y; \theta). \quad (28.3)$$

Supongamos una transformación 1-1 de  $Y$  a  $Z$ ,  $Z = g(Y)$ . Las densidades de ambos vectores se relacionan según la siguiente relación

$$f_Z(z) = f_Y(y) \left| \frac{\partial y}{\partial z} \right|,$$

donde  $\left| \frac{\partial y}{\partial z} \right|$  es el jacobiano de la transformación de  $z$  a  $y$ . Se tiene la siguiente relación entre las verosimilitudes

$$L_Z(\theta; z) = \left| \frac{\partial y}{\partial z} \right| L_Y(\theta; y).$$

Esto sugiere que es mejor trabajar con el cociente de las verosimilitudes para dos vectores de parámetros  $\theta_1$  y  $\theta_2$  en lugar de los valores aislados.

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

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

y

$$l_y(\theta; y) = \sum_{i=1}^n \log f_{Y_i}(y_i) = \sum_{i=1}^n L_{Y_i}(\theta; y_i).$$

Veamos algunos ejemplos de verosimilitud.

**Ejemplo 28.1 (Pruebas Bernoulli)**  $Y_1, \dots, Y_n$  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 28.2 (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  $\theta$ . La verosimilitud viene dada por

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

**Ejemplo 28.3 (Muestreo Bernoulli inverso)** 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  $\theta$ . 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 28.1, 28.2 y 28.3. Si consideramos dos valores del parámetro  $\theta_1$  y  $\theta_2$  entonces el cociente de las verosimilitudes calculados en ambos valores tiene el mismo valor en los tres ejemplos.

## 28.2 Estimación

Denotamos por  $\Theta$  el espacio formado por los valores que puede tomar  $\theta$  o espacio paramétrico. Un **estimador** del parámetros o vector paramétrico  $\theta$  es cualquier función de la muestra  $X_1, \dots, X_n$  que toma valores en el espacio paramétrico. Si  $\delta(X_1, \dots, X_n)$  es un estimador del parámetro  $\theta$  entonces se define el **error cuadrático medio** como

$$MSE(\delta) = E[\delta(X_1, \dots, X_n) - \theta]^2 \quad (28.4)$$

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

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

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

$$MSE(\delta) = E[\delta - \theta]^2 = E[\delta - \mu_\delta + \mu_\delta - \theta]^2 = E[\delta - \mu_\delta]^2 + [\mu_\delta - \theta]^2 \quad (28.5)$$

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.

### 28.2.1 Estimación insesgada de media y varianza

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

$$\bar{Y} = \frac{1}{n} \sum_{i=1}^n Y_i. \quad (28.6)$$

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  $\sigma^2$  a partir de una muestra  $Y_1, \dots, Y_n$  se utiliza la varianza muestral dada por

$$S^2 = \frac{1}{n-1} \sum_{i=1}^n (Y_i - \bar{Y})^2. \quad (28.7)$$

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

siguientes igualdades.

$$\begin{aligned} E \sum_{i=1}^n (Y_i - \bar{Y})^2 &= \\ E \sum_{i=1}^n [(Y_i - \mu) - (\bar{Y})^2 - \mu]^2 &= \sum_{i=1}^n E(Y_i - \mu)^2 - nE(\bar{Y} - \mu)^2, \end{aligned} \quad (28.8)$$

pero  $E(Y_i - \mu)^2 = \sigma^2$  y  $E(\bar{Y} - \mu)^2 = \text{var}(\bar{Y}) = \sigma^2/n$ . 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  $\sigma^2$  sin sesgo.

### 28.2.2 Estimación insesgada del vector de medias y la matriz de covarianzas

Ahora consideramos una muestra de un vector de dimensión  $d$ ,  $Y_1, \dots, Y_n$  i.i.d. con vector de medias  $\mu = EY_i$  y matriz de covarianzas  $\Sigma = \text{cov}(Y_i)$ . Los estimadores insesgados de  $\mu$  y  $\Sigma$  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

$$Y = \begin{bmatrix} Y'_1 \\ \vdots \\ Y'_n \end{bmatrix} = \begin{bmatrix} Y_{11} & \dots & Y_{1d} \\ \vdots & \vdots & \vdots \\ Y_{n1} & \dots & Y_{nd} \end{bmatrix}$$

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

$$y = \begin{bmatrix} y'_1 \\ \vdots \\ y'_n \end{bmatrix} = \begin{bmatrix} y_{11} & \dots & y_{1d} \\ \vdots & \vdots & \vdots \\ y_{n1} & \dots & y_{nd} \end{bmatrix}$$

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

$$\bar{Y} = \frac{1}{n} \sum_{i=1}^n Y_i = \frac{1}{n} Y' 1_n. \quad (28.9)$$

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

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

El estimador de la matriz de covarianzas (poblacional)  $\Sigma$  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

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

Es inmediato que  $E\bar{Y} = \mu$  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  $\bar{X} = \bar{X} - \mu$ . Se sigue que

$$\sum_{i=1}^n (Y_i - \bar{Y})(Y_i - \bar{Y})' = \sum_{i=1}^n (X_i - \bar{X})(X_i - \bar{X})' = \sum_{i=1}^n X_i X_i' - n \bar{X} \bar{X}'.$$

Los vectores  $X_1, \dots, X_n$  tienen vector de medias nulo y matriz de covarianzas  $\Sigma$ , la misma que los  $Y_i$ . En consecuencia,  $E\bar{X} \bar{X}' = \Sigma$  y

$$EQ = \sum_{i=1}^n cov(Y_i) - n cov(\bar{Y}) = n\Sigma - n cov(\bar{Y}) = n\Sigma - n \frac{\Sigma}{n} = (n-1)\Sigma.$$

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

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

$$r_{jk} = \frac{\sum_{i=1}^n (Y_{ij} - \bar{Y}_{\cdot j})(Y_{ik} - \bar{Y}_{\cdot k})}{\sqrt{\sum_{i=1}^n (Y_{ij} - \bar{Y}_{\cdot j})^2 \sum_{i=1}^n (Y_{ik} - \bar{Y}_{\cdot k})^2}} = \frac{S_{jk}}{\sqrt{S_{jj} S_{kk}}} \quad (28.10)$$

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

## 28.3 Estimador máximo verosímil

El método de estimación que vamos a utilizar en este curso el **método de máxima verosimilitud**. El estimador máximo verosímil de  $\theta$ , que denotaremos por  $\hat{\theta}$ , se obtienen maximizando 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**.

$$L(\hat{\theta}) = \max_{\theta \in \Theta} L(\theta), \quad (28.11)$$

o también

$$\hat{\theta} = argmax_{\theta \in \Theta} L(\theta), \quad (28.12)$$

**Ejemplo 28.4 (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  $\theta^* = f(\hat{\theta})$  siendo  $f$  una biyección entonces el estimador máximo verosímil de  $\theta^*$  es verifica que

$$\hat{\theta}^* = f(\hat{\theta}). \quad (28.13)$$

**Ejemplo 28.5 (Normal)** En este caso se comprueba que  $\hat{\mu} = \bar{X}_n$  y que  $\hat{\sigma}^2 = \frac{n-1}{n} S^2 = \frac{1}{n} \sum_{i=1}^n (X_i - \bar{X}_n)^2$ . Teniendo en cuenta la propiedad enunciada en 28.13 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 [115], página 364), se verifica que  $cov(\hat{\theta})$  es la inversa de la **matriz de información** cuyo elemento  $(j, k)$  viene dado por

$$-E\left(\frac{\partial^2 l(\theta)}{\partial \theta_j \partial \theta_k}\right) \quad (28.14)$$

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  $\theta$  se aleja de  $\hat{\theta}$ . En resumen, es de esperar que  $\theta$  esté más próximo a  $\hat{\theta}$ .

**Ejemplo 28.6 (Binomial)** Supongamos que una muestra en una población finita y consideremos como valor observado el número de éxitos. Entonces la verosimilitud sería

$$L(p) = \binom{n}{y} p^y (1-p)^{n-y}, \quad (28.15)$$

y la logverosimilitud viene dada como

$$l(p) = \log \binom{n}{y} + y \log p + (n-y) \log(1-p), \quad (28.16)$$

La ecuación de verosimilitud sería

$$\frac{\partial l(p)}{\partial p} = \frac{y}{p} - \frac{n-y}{1-p} = \frac{y-np}{p(1-p)}. \quad (28.17)$$

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 asintótica sería

$$-E\left[\frac{\partial^2 l(p)}{\partial p^2}\right] = E\left[\frac{y}{p^2} + \frac{n-y}{(1-p)^2}\right] = \frac{n}{p(1-p)}. \quad (28.18)$$

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  $var(Y) = np(1-p)$ .

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

## 28.4 Contraste de hipótesis

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

$$H_0 : \theta \in \Theta_0 \quad (28.19)$$

$$H_1 : \theta \in \Theta_1 \quad (28.20)$$

Basándonos en una muestra aleatoria  $X_1, \dots, X_n$  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  $(X_1, \dots, X_n)$ . Entonces el contraste de hipótesis se basa en tomar un estadístico o función de la muestra que denotamos  $\delta(X_1, \dots, X_n)$  de modo que si  $\delta(X_1, \dots, X_n) \in C$  entonces rechazamos la hipótesis nula mientras que si  $\delta(X_1, \dots, X_n) \notin C$  entonces no rechazamos la hipótesis nula. Notemos que simplemente estamos partiendo 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  $\delta$  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

$$\pi(\theta) = P(\delta \in C | \theta). \quad (28.21)$$

### 28.4.1 Contraste de la media en la poblaciones normales

Si tenemos una muestra  $X_1, \dots, X_n$  de una población normal con media  $\mu$  y varianza  $\sigma^2$  donde ambos parámetros se asumen desconocidos un test habitualmente considerado es si la media toma un valor dado. El test formalmente planteado sería:

$$H_0 : \mu = \mu_0, \quad (28.22)$$

$$H_1 : \mu \neq \mu_0. \quad (28.23)$$

Siendo  $S^2 = \frac{\sum_{i=1}^n (X_i - \bar{X})^2}{n-1}$ , el estadístico habitualmente utilizado es el siguiente

$$T = \frac{\bar{X} - \mu_0}{S/\sqrt{n}}.$$

Bajo la hipótesis nula este estadístico sigue una distribución t de Student con  $n - 1$  grados de libertad,

$$T \sim t(n - 1).$$

Si suponemos que trabajamos con un nivel de significación  $\alpha$  la región crítica en la cual rechazamos la hipótesis nula sería

$$|T| > t_{n-1, 1-\frac{\alpha}{2}}.$$

### 28.4.2 Test del cociente de verosimilitudes

El cociente de verosimilitudes para contrastar estas hipótesis se define como

$$\Lambda = \frac{\max_{\theta \in \Theta_0} L(\theta)}{\max_{\theta \in \Theta} L(\theta)} \quad (28.24)$$

Es razonable pensar que en la medida en que  $\Lambda$  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  $\Theta_0$ . 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

$$-2 \log \lambda = -2 \log \frac{L_0}{L_1} = -2(l_0 - l_1) \quad (28.25)$$

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  $\Theta_0$  y sobre  $\Theta$ .

### 28.4.3 Test de Wald

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

$$H_0 : \theta = \theta_0, \quad (28.26)$$

$$H_1 : \theta \neq \theta_0. \quad (28.27)$$

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

$$z = \frac{\hat{\theta} - \theta_0}{SE(\hat{\theta})} \quad (28.28)$$

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  $\theta$  es un vector de parámetros y queremos contrastar las hipótesis dadas en 28.26. La versión multivariante del estadístico dado en 28.28 viene dada por

$$W = (\hat{\theta} - \theta_0)' [cov(\hat{\theta})]^{-1} (\hat{\theta} - \theta_0), \quad (28.29)$$

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  $\theta$ .

### 28.4.4 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 28.7 (Intervalo de confianza para la media de una normal)** Veámoslo con un ejemplo y luego planteamos la situación más general. Tenemos una muestra aleatoria  $X_1, \dots, X_n$  i.i.d. tales que  $X_i \sim N(\mu, \sigma^2)$ . Entonces es conocido que

$$\frac{\bar{X}_n - \mu}{S/\sqrt{n}} \sim t_{n-1}. \quad (28.30)$$

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

$$P(-t_{n-1, 1-\alpha/2} \leq \frac{\bar{X}_n - \mu}{S/\sqrt{n}} \leq t_{n-1, 1-\alpha/2}) = 1 - \alpha. \quad (28.31)$$

La ecuación anterior la podemos reescribir como

$$P(\bar{X}_n - t_{n-1, 1-\alpha/2} \frac{S}{\sqrt{n}} \leq \mu \leq \bar{X}_n + t_{n-1, 1-\alpha/2} \frac{S}{\sqrt{n}}) = 1 - \alpha. \quad (28.32)$$

Tenemos una muestra aleatoria  $X_1, \dots, X_n$  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  $\mu$  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  $\mu$  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  $\alpha$  para contrastar las hipótesis anteriores y consideramos el siguiente conjunto formado por todos los  $\theta_0$  tales que no rechazamos la hipótesis nula al nivel  $\alpha$ . 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  $\chi_k^2(1 - \alpha)$  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

$$\{\theta_0 : -2[l(\theta_0) - l(\hat{\theta})] < \chi_k^2(1 - \alpha)\} \quad (28.33)$$

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

$$\{\theta_0 : \frac{|\hat{\theta} - \theta_0|}{SE(\hat{\theta})} < Z_{1-\alpha/2}\} \quad (28.34)$$

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

## Capítulo 29

# Modelos lineales generalizados

En este tema se habla de modelos lineales generalizados. Este tipo de modelos se utilizan, en el contexto de los datos ómicos, cuando trabajamos con conteos de secuencias cortas alineadas sobre regiones genómicas. La referencia básica en este capítulo es el texto [4].

Los datos que pretendemos modelizar son pares  $(x_i, y_i)$  siendo  $x_i$  un vector de covariables o variables predictoras que queremos utilizar para conocer o predecir el valor de  $y_i$ . Los valores  $y_i$  que pretendemos predecir serán categóricos con dos categorías (binarios)  $y_i \in \{0, 1\}$  o un más de dos o bien conteos. Nuestro interés se centra en la modelización de conteos pero es conveniente adoptar una visión más amplia para la comprensión y manejo de este tipo de modelos probabilísticos. En donde se modeliza la distribución conjunta de las respuestas  $Y_i$  condicionadas a los predictores  $x_i$ .

### 29.1 Un modelo lineal generalizado

Un modelo lineal generalizado<sup>266</sup> se compone de los tres elementos que siguen:

**La componente aleatoria** En donde se indica la variable respuesta  $Y$  así como su distribución de probabilidad (marginal).

**La componente sistemática** Especificamos las variables explicativas (o independientes o predictoras) utilizadas como combinarlas linealmente.

**Función de enlace** <sup>267</sup> Especifica la función que nos relaciona la media de la variable  $Y$ ,  $EY$ , con la combinación lineal de las variables predictoras.

Veamos con detalle cada elemento.

**La componente aleatoria de un GLM** consiste de una variable aleatoria  $Y$  con observaciones independientes ( $Y_1, \dots, Y_n$ ). Los valores observados son  $(y_1, \dots, y_n)$  Suponemos que la distribución de  $Y$  está en la *familia exponencial natural*. Para la  $i$ -ésima variable  $Y_i$  suponemos una función de densidad o de probabilidad en la familia de dispersión exponencial.<sup>268</sup>

<sup>266</sup> Abreviadamente utilizaremos la expresión **GLM** por **generalized linear model**.

<sup>267</sup> Link function.

<sup>268</sup> Exponential dispersion family

$$f(y_i|\theta_i, \phi) = \exp \left[ \frac{y_i\theta_i - b(\theta_i)}{a(\phi)} + c(y_i, \phi) \right] \quad (29.1)$$

donde  $\theta_i$  es el parámetro natural y  $\phi$  es el parámetro de dispersión. Si  $a(\phi) = 1$  entonces tenemos la familia exponencial natural con densidad

$$f(y_i|\theta_i) = h(y_i) \exp [y_i\theta_i - b(\theta_i)]$$

**La componente sistemática de un GLM** es el vector  $(\eta_1, \dots, \eta_n)$  siendo

$$\eta_i = \sum_j \beta_j x_{ij}, \text{ con } i = 1, \dots, n,$$

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. Cuando esto es así consideraremos que el vector de covariables para la  $i$ -ésima observación viene dado por  $x_i = (1, x_{i1}, \dots, x_{ip-1})'$ . De este modo podemos representar de un modo simplificado

$$\eta_i = x'_i \boldsymbol{\beta}$$

donde  $x_i$  es el vector de covariables o predictores y  $\boldsymbol{\beta}_i = (\beta_0, \beta_1, \dots, \beta_{p-1})'$ .

Mediante la función de enlace<sup>269</sup> relacionamos las componentes aleatoria y sistemática. En concreto nos especifica cómo obtener la componente sistemática a partir de la media de la variable  $Y_i$ ,  $\mu_i = E(Y_i)$ , con  $i = 1, \dots, n$ . En concreto tenemos que

$$\eta_i = g(\mu_i)$$

O lo que es lo mismo

$$g(\mu_i) = \sum_j \beta_j x_{ij}$$

para  $i = 1, \dots, n$ .

En el siguiente ejemplo consideramos el caso de respuesta binaria.

**Ejemplo 29.1 (Modelos logit binomiales para datos binarios)** Tenemos respuesta binaria (éxito como 1 y fracaso como 0), esto es,  $Y \sim Bi(1, \pi)$ .

$$\begin{aligned} f(y|\pi) &= \pi^y (1-\pi)^{1-y} = \\ &(1-\pi)[\pi/(1-\pi)]^y = (1-\pi) \exp \left( y \log \frac{\pi}{1-\pi} \right) \end{aligned}$$

con  $y = 0, 1$ . El parámetro natural sería

$$Q(p) = \log \frac{p}{1-p} = \text{logit}(p)$$

**Ejemplo 29.2 (Modelos loglineales Poisson para conteos)** La variable respuesta  $Y$  es un conteo (número de defectos, conteo en una tabla de contingencia o, en nuestro caso, lecturas alienadas sobre un gen). Asumimos que la variable  $Y$  sigue una distribución de Poisson con media  $\mu$ . La función de probabilidad viene dada por

$$Y \sim Po(\mu)$$

<sup>269</sup> Link function

donde  $EY = \mu$ . Veamos que esta función de probabilidad pertenece a la familia exponencial natural.

$$f(y; \mu) = \frac{e^{-\mu} \mu^y}{y!} = e^{-\mu} \frac{1}{y!} e^{y \log \mu},$$

con  $y = 0, 1, \dots$ . El parámetro natural sería

$$Q(\mu) = \log(\mu),$$

y la función de enlace canónica es  $\eta = \log \mu$ .

## 29.2 Desviación

Los valores observados son  $y = (y_1, \dots, y_n)$  y el vector de medias  $\boldsymbol{\mu} = (\mu_1, \dots, \mu_n)$ . Sea  $L(\boldsymbol{\mu}; y)$  la logverosimilitud expresada como función de las medias. Sea  $L(\hat{\boldsymbol{\mu}}; y)$  el máximo de la logverosimilitud *asumiendo el modelo*. Si utilizamos un parámetro distinto para cada observación. Esto nos proporciona un ajuste perfecto con

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

La logverosimilitud máxima es  $L(y; y)$ . El modelo con un parámetro por observación se llama un *modelo saturado*. No supone reducción alguna de los datos. No es modelo parsimonioso que suponga una descripción simplificada de los datos. La **desviación** se define como la diferencia entre las dos logverosimilitudes, esto es,

$$-2 \left[ L(\hat{\boldsymbol{\mu}}; y) - L(y; y) \right].$$

Es el test del cociente de verosimilitud para contrastar el modelo que asumimos frente a la alternativa del modelo saturado. Cuando los conteos de Poisson o bien el número de pruebas las distintas binomiales con  $n$  fijo tenemos que aproximadamente

$$-2 \left[ L(\hat{\boldsymbol{\mu}}; y) - L(y; y) \right] \sim \chi^2(n - p)$$

## 29.3 Modelos lineales generalizados para datos binarios

Consideramos que la variable aleatoria  $Y$  es binaria. Entonces  $EY = P(Y = 1)$  y la denotamos por  $\pi(x)$  indicando la dependencia de  $x = (x_1, \dots, x_p)$ . Además su varianza viene dada por  $\text{var}(Y) = \pi(x)(1 - \pi(x))$ .

**Ejemplo 29.3 (GLM binomial y tablas  $2 \times 2$ )** La variable predictora  $X$  es binaria.

$$\text{link}[\pi(x)] = \alpha + \beta x.$$

El efecto de  $X$  viene descrito por

$$\beta = \text{link}[\pi(1)] - \text{link}[\pi(0)]$$

Con la función de enlace identidad:

$$\beta = \pi(1) - \pi(0)$$

es la diferencia de proporciones. Si la función de enlace es el logaritmo natural:

$$\beta = \log \pi(1) - \log \pi(0) = \log \frac{\pi(1)}{\pi(0)}$$

es el logaritmo del riesgo relativo. Con el la función de enlace logit:

$$\beta = \text{logit}(\pi(1)) - \text{logit}(\pi(0)) = \log \frac{\pi(1)/(1-\pi(1))}{\pi(0)/(1-\pi(0))}$$

es el logaritmo de los odds ratio.

**Ejemplo 29.4** Probit y otras funciones de enlace Con un solo predictor parece natural el modelo

$$\pi(x) = F(x)$$

siendo  $F$  una función de distribución de probabilidad. Si consideramos  $F(x) = \Phi(x)$  siendo  $\Phi(x)$  la función de distribución de la normal estándar entonces

$$\pi(x) = \Phi(\alpha + \beta x)$$

o equivalentemente

$$\Phi^{-1}(\pi(x)) = \alpha + \beta x$$

y tenemos un modelo probit.

## 29.4 Modelo Poisson loglineal

Asumimos  $Y$  con distribución de Poisson. El modelo loglineal con variable explicativa  $X$  es

$$\log \mu = \alpha + \beta x.$$

En este modelo

$$\mu = \exp(\alpha + \beta x) = e^\alpha \left( e^\beta \right)^x$$

**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. Interpretación como mixturas de Poisson. No es un problema cuando  $Y$  tiene una distribución normal pues la normal tiene un parámetro que la modeliza.

Consideremos un modelo lineal generalizado binomial negativo. 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  $y = 0, 1, 2, \dots$  donde  $k$  y  $\mu$  son los parámetros.

Se tiene que

$$E(Y) = \mu, \quad \text{var}(Y) = \mu + \mu^2/k.$$

El parámetro  $1/k$  es un parámetro de dispersión. Si  $1/k \rightarrow 0$ , entonces  $\text{var}(Y) \rightarrow \mu$  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.

### 29.4.1 GLM Poisson e independencia en tablas de contingencia $I \times J$

- Vamos a utilizar un modelo loglineal Poisson para modelizar conteos en tablas de contingencia.

- Suponemos que el conteo  $Y_{ij} \sim Po(\mu_{ij})$ .

- Suponemos que

$$\mu_{ij} = \mu \alpha_i \beta_j$$

siendo  $\alpha_i, \beta_j \geq 0$  tales que  $\sum_i \alpha_i = 1$  y  $\sum_j \beta_j = 1$ .

- Si consideramos un log link tenemos

$$\log \mu_{ij} = \log \mu + \log \alpha_i + \log \beta_j$$

- Se comprueba que si hay independencia tenemos que  $\alpha_i = \pi_{i+}$  y  $\beta_j = \pi_{+j}$ .

## 29.5 Verosimilitud de modelos lineales generalizados

### 29.5.1 Familia de dispersión exponencial

- Tenemos  $(y_1, \dots, y_n)$  independientes.

- La variable  $y_i$  tiene densidad en la familia de dispersión exponencial

$$f(y_i; \theta_i, \phi) = \exp\left(\left[y_i \theta_i - b(\theta_i)\right]/a(\phi) + c(y_i, \phi)\right)$$

- $\theta_i$  es el parámetro natural.

- Si  $\phi$  es conocido entonces la densidad anterior es de la familia exponencial natural.

	Normal	Poisson	Binomial
$f(y)$	$\frac{1}{\sqrt{2\pi}\sigma} \exp(-\frac{(y-\mu)^2}{2\sigma^2})$	$\mu^y e^{-\mu}/y!$	$\binom{n}{y} (\frac{\mu}{n})(1-\frac{\mu}{n})^{n-y}$
$\theta$	$\mu$	$\log \mu$	$\log(\mu/(n-\mu))$
$\phi$	$\sigma^2$	1	1
$a(\phi)$	$\phi$	$\phi$	$\phi$
$b(\theta)$	$\frac{\theta^2}{2}$	$\exp(\theta)$	$n \log(1 + e^\theta)$
$c(y, \phi)$	$-\frac{1}{2}[\frac{y^2}{\phi} + \log(2\pi\phi)]$	$-\log y!$	$\log \binom{n}{y}$

Se tiene que la media y varianza de la componente aleatoria vienen dadas por Se verifica:

$$\mu_i = E(Y_i) = b'(\theta_i).$$

$y$

$$var(Y_i) = b''(\theta_i)a(\phi).$$

La componente sistemática viene dada por

$$\eta_i = \sum_j \beta_j x_{ij}, \quad i = 1, \dots, n.$$

y en forma matricial

$$\boldsymbol{\eta} = X\beta$$

con  $\boldsymbol{\eta} = (\eta_1, \dots, \eta_p)'$  y  $\beta = (\beta_1, \dots, \beta_p)'$  siendo  $X$  la matriz de modelo.

- Un GLM relaciona  $\eta_i$  con  $\mu_i$  con la función link.

$$\eta_i = g(\mu_i) = \sum_j \beta_j x_{ij}, \quad i = 1, \dots, n.$$

- El parámetro natural  $\theta_i$  es

$$\theta_i = g(\mu_i) = \sum_j \beta_j x_{ij}$$

La logverosimilitud sería

$$L(\beta) = \sum_i L_i = \sum_i \log f(y_i; \theta_i, \phi) = \sum_i \frac{y_i \theta_i - b(\theta_i)}{a(\phi)} + \sum_i c(y_i, \phi).$$

Las ecuaciones de verosimilitud son

$$\sum_{i=1}^n \frac{(y_i - \mu_i)x_{ij}}{\text{var}(Y_i)} \frac{\partial \mu_i}{\partial \eta_i} = 0, \quad j = 1, \dots, p.$$

La matriz de covarianza asintótica se puede obtener del siguiente modo.

- Puesto que  $-E \frac{\partial^2 L(\beta)}{\partial \beta_h \partial \beta_j} = E \frac{\partial L(\beta)}{\partial \beta_h} \frac{\partial L(\beta)}{\partial \beta_j}$ ,

$$\begin{aligned} \mathbb{I}_{hj} &= -E \frac{\partial^2 L(\beta)}{\partial \beta_h \partial \beta_j} = -\sum_{i=1}^n E \frac{\partial^2 L_i(\beta)}{\partial \beta_h \partial \beta_j} = \\ &\sum_{i=1}^n E \frac{\partial L_i(\beta)}{\partial \beta_h} \frac{\partial L_i(\beta)}{\partial \beta_j} = \sum_{i=1}^n \frac{x_{ih}x_{ij}}{\text{var}(Y_i)} \left( \frac{\partial \mu_i}{\partial \eta_i} \right)^2 \end{aligned}$$

- Y la podemos expresar como

$$\mathbb{I} = X' W X$$

siendo

$$W = \text{diag}(w_1, \dots, w_p)$$

con

$$w_i = \frac{1}{\text{var}(Y_i)} \left( \frac{\partial \mu_i}{\partial \eta_i} \right)^2$$

La matriz de covarianza asintótica estimada.

- Estimamos  $W$  en  $\hat{\beta}$  y tendremos

$$\hat{\mathbb{I}} = X' \hat{W} X$$

siendo

$$\widehat{\text{cov}}(\hat{\beta}) = (X' \hat{W} X)^{-1}$$

**Ejemplo 29.5 (Modelo loglineal de Poisson)** • Tenemos

$$\log \mu = X\beta$$

- La componente sistemática se relaciona con la media como

$$\eta_i = \log \mu_i$$

de donde

$$\frac{\partial \mu_i}{\partial \eta_i} = \exp \eta_i = \mu_i$$

- Además

$$\text{var } Y_i = \mu_i$$

- En este caso las ecuaciones de verosimilitud son

$$\sum_i (y_i - \mu_i)x_{ij} = 0 \text{ para } j = 1, \dots, p.$$

- Estamos igualando los estadísticos suficientes  $\sum_i y_i x_{ij}$  a sus valores esperados  $\sum_i \mu_i x_{ij}$

- Además

$$w_i = \frac{1}{\text{var}(Y_i)} \left( \frac{\partial \mu_i}{\partial \eta_i} \right)^2 = \mu_i$$

## 29.6 Inferencia para modelos lineales generalizados

### 29.6.1 Desviación y bondad de ajuste

- Un GLM saturado es el modelo que tiene un parámetro distinto para cada observación.
- Nos proporciona un (completamente inútil) ajuste perfecto.
- Sea  $\tilde{\theta}_i$  la estimación de  $\theta_i$  en el modelo saturado que corresponde con  $\tilde{\mu}_i = y_i$  para cualquier  $i$ .
- Consideraremos un modelo (insaturado) y denotemos por  $\hat{\theta}_i$  y  $\hat{\mu}_i$  los estimadores máximo verosímiles.
- La **desviación** viene dada por

$$\begin{aligned} & -2[L(\hat{\mu}; y) - L(y; y)] = \\ & 2 \sum_i \left[ y_i \tilde{\theta}_i - b(\tilde{\theta}_i) \right] / a(\phi) - 2 \sum_i \left[ y_i \hat{\theta}_i - b(\hat{\theta}_i) \right] / a(\phi) \end{aligned}$$

Si asumimos que

$$a(\phi) = \frac{\phi}{\omega_i}$$

entonces

$$\begin{aligned} & -2[L(\hat{\mu}; y) - L(y; y)] = \\ & 2 \sum_i \omega_i \left[ y_i (\tilde{\theta}_i - \hat{\theta}_i) - b(\tilde{\theta}_i) + b(\hat{\theta}_i) \right] / \phi = D(y; \hat{\mu}) / \phi. \end{aligned}$$

- $D(y; \hat{\mu}) / \phi$  es la *desviación escalada* y

- $D(y; \hat{\mu})$  es la desviación.

**Ejemplo 29.6 (Desviación para modelos loglineales Poisson)** • Tenemos  
 $\hat{\theta}_i = \log \hat{\mu}_i$ ,  $b(\hat{\theta}_i) = \exp \hat{\theta}_i = \hat{\mu}_i$ ,  $\tilde{\theta}_i = \log y_i$ ,  $b(\tilde{\theta}) = y_i$ ,  $a(\phi) = 1$ .

- La desviación es igual a

$$D(y, \hat{\mu}) = 2 \sum_i [y_i \log(y_i/\hat{\mu}_i) - y_i + \hat{\mu}_i].$$

- Si el modelo incorpora un término constante entonces una de las ecuaciones de verosimilitud es  $\sum_i y_i = \sum_i \hat{\mu}_i$  y

$$D(y, \hat{\mu}) = 2 \sum_i y_i \log(y_i/\hat{\mu}_i).$$

- Por tanto, si hay término constante en el modelo

$$D(y, \hat{\mu}) = 2 \sum_i \text{Observado} \log \frac{\text{Observado}}{\text{Ajustado}}.$$

**Ejemplo 29.7 (Desviación para modelos binomiales)** • Denotamos  $y_i$  la proporción muestral basada en una muestra  $n_i$ .

- Tenemos

$$\hat{\theta}_i = \log[\hat{\pi}_i/(1 - \hat{\pi}_i)]$$

$$\tilde{\theta}_i = \log[y_i/(1 - y_i)]$$

$$a(\phi) = \frac{1}{n_i}$$

$$\phi = 1, \omega_i = n_i.$$

- La desviación viene dada por

$$2 \sum_i n_i y_i \log \frac{n_i y_i}{n_i \hat{\pi}_i} + 2 \sum_i (n_i - n_i y_i) \log \frac{n_i - n_i y_i}{n_i - n_i \hat{\pi}_i}$$

- Si consideramos la tabla de contingencia en donde la fila  $i$ -ésima corresponde con el  $i$ -ésimo setting (todas las covariables o variables predictoras son comunes) y tenemos dos columnas correspondiendo con éxito y fracaso entonces la desviación tiene la (bonita) interpretación

$$2 \sum_i \text{Observado} \log \frac{\text{Observado}}{\text{Ajustado}}$$

### 29.6.2 Comparación de modelos utilizando la desviación

- En un modelo binomial o Poisson,  $\phi = 1$ , y la desviación es

$$D(y; \hat{\mu}) = -2[L(\hat{\mu}; y) - L(y; y)].$$

- Queremos comparar dos modelos  $M_0$  frente a  $M_1$ .

- $M_0$  anidado en  $M_1$  (posiblemente eliminando algunas variables).
- Se tiene

$$-2[L(\hat{\mu}_0; y) - L(\hat{\mu}_1; y)] = D(y; \hat{\mu}_0) - D(y; \hat{\mu}_1)$$

por lo que el estadístico del test del cociente de verosimilitudes es la diferencia de las desviaciones.

Comparación de modelos utilizando la desviación:

- En concreto en modelos loglineales Poisson con término constante y en modelos logit binomiales se tiene

$$D(y; \hat{\mu}_0) - D(y; \hat{\mu}_1) = 2 \sum_i \text{Observado} \log \frac{\text{Ajustado}_1}{\text{Ajustado}_0}$$

### 29.6.3 Residuos en un GLM

- Si definimos

$$d_i = 2\omega_i \left[ y_i(\tilde{\theta}_i - \hat{\theta}_i) - b(\tilde{\theta}_i) + b(\hat{\theta}_i) \right]$$

entonces el **residuo de la desviación** es

$$\sqrt{d_i} \times \text{signo}(y_i - \hat{\mu}_i)$$

- El **residuo de Pearson** es

$$e_i = \frac{y_i - \hat{\mu}_i}{\widehat{\text{var}}(Y_i)^{1/2}}$$

- Pero

$$\text{cov}(Y - \hat{\mu}) = \text{cov}(Y)(I - H)$$

con

$$H = W^{\frac{1}{2}} X (X' W X)^{-1} X' W^{\frac{1}{2}}$$

- Sustituyendo  $W$  por  $\hat{W}$  obtenemos  $\hat{H}$ .
- Si  $\hat{h}_i$  es el elemento que ocupa la posición  $i$ -ésima en la diagonal principal entonces el **residuo de Pearson estandarizado** es

$$r_i = \frac{y_i - \hat{\mu}_i}{\left( \widehat{\text{var}}(Y_i)(1 - \hat{h}_i) \right)^{\frac{1}{2}}}$$

- [notaR/notaR025.pdf](#).

### 29.6.4 Ajuste de un GLM mediante Newton-Raphson

- Sea el vector gradiente

$$u = \frac{\partial L(\beta)}{\partial \beta}$$

y la matriz hessiana

$$H = \left[ \frac{\partial^2 L(\beta)}{\partial \beta_a \partial \beta_b} \right]_{a,b=1,\dots,p}$$

- $u^{(t)}$  y  $H^{(t)}$  denotan  $u$  y  $H$  evaluados en  $\beta^{(t)}$ .
- Actualizamos nuestra estimación  $\beta^{(t)}$

$$\beta^{(t+1)} = \beta^{(t)} - \left( H^{(t)} \right)^{-1} u^{(t)}$$

hasta que no haya una variación apreciable en la estimación obtenida.

### 29.6.5 IRLS

En esta sección vamos a ver el procedimiento que se utiliza para el

<sup>270</sup> Iteratively Reweighted Least Square, esto es, mínimos cuadrados iterativamente reponderados. ajuste de los modelos lineales generalizados conocido como IRLS<sup>270</sup>

Vamos a suponer que la componente aleatoria del modelo o distribución de la variable respuesta sigue una distribución en la familia de dispersión exponencial.

- Es como el método Newton-Raphson en donde sustituimos la matriz hessiana que no es más que la información observada por la matriz de información de Fisher o valor esperado de dicha matriz, es decir, por

$$\mathbb{I} = - \left[ E \frac{\partial^2 L(\beta)}{\partial \beta_a \partial \beta_b} \right]_{a,b=1,\dots,p}$$

- Actualizamos la estimación de los parámetros mediante

$$\beta^{(t+1)} = \beta^{(t)} + \left( \mathbb{I}^{(t)} \right)^{-1} u^{(t)}$$

o bien

$$\mathbb{I}^{(t)} \beta^{(t+1)} = \mathbb{I}^{(t)} \beta^{(t)} + u^{(t)}$$

- Notemos que  $\mathbb{I} = X' W X$  y  $\mathbb{I}^{(t)} = X' W^{(t)} X$  donde  $W^{(t)}$  es  $W$  evaluada en  $\beta^{(t)}$ . Cuando finalizamos el proceso iterativo tendremos la matriz  $\mathbb{I}^{(t)}$  cuya inversa es la matriz de covarianzas de los estimadores. Obtenemos dicha matriz como un subproducto.

### 29.7 ML como mínimos cuadrados reponderados iterativamente

- Si consideramos un modelo lineal general

$$z = X\beta + \epsilon$$

donde  $\epsilon$  tiene matriz de covarianzas  $V$  entonces la estimación mínimo cuadrática es

$$(X' V^{-1} X)^{-1} X' V^{-1} z$$

- Se ve fácilmente que

$$\mathbb{I}^{(t)} \beta^{(t)} + u^{(t)} = X' W^{(t)} z^{(t)}$$

con

$$\begin{aligned} z_i^{(t)} &= \sum_j x_{ij} \beta_j^{(t)} + (y_i - \mu_i^{(t)}) \frac{\partial \eta_i^{(t)}}{\partial \mu_i^{(t)}} = \\ &\quad \eta_i^{(t)} + (y_i - \mu_i^{(t)}) \frac{\partial \eta_i^{(t)}}{\partial \mu_i^{(t)}} \end{aligned}$$

- El método de actualización del Fisher scoring method queda como

$$(X' W^{(t)} X) \beta^{(t+1)} = X' W^{(t)} z^{(t)}$$

que no son más que las ecuaciones normales en un modelo lineal general donde las respuestas son  $z^{(t)}$  y la matriz de covarianzas del error es  $W^{(t)}$ .

- Notemos que  $z_i$  se relaciona con  $y_i$  mediante una aproximación lineal de la función link  $g$ .

$$g(y_i) \approx g(\mu_i) + (y_i - \mu_i) g'(\mu_i) = \eta_i + (y_i - \mu_i) \frac{\partial \eta_i}{\partial \mu_i} = z_i$$

- En cada iteración  $z$  es estimado mediante  $z^{(t)}$  y son utilizados como respuestas mientras que la matriz de covarianzas es estimada con  $\hat{W}^{(t)}$ .
- Obtenemos  $\beta^{(t+1)}$  que nos da una nueva componente sistemática  $\eta^{(t+1)}$  (utilizada para calcular  $\hat{W}^{(t+1)}$ ) y un nuevo vector respuesta  $z^{(t+1)}$ .
- Vemos que estamos aplicando iterativamente unos mínimos cuadrados ponderados donde en cada iteración cambia la matriz de pesos (la matriz de covarianza estimada).
- Por ello se habla de **mínimos cuadrado iterativos reponderados** o **iterative reweighted least squares**.

## 29.8 Cuasiverosimilitud Wedderburn(1974)

- Las ecuaciones de verosimilitud son

$$\sum_{i=1}^n \frac{(y_i - \mu_i)x_{ij}}{var(Y_i)} \frac{\partial \mu_i}{\partial \eta_i} = 0, \quad j = 1, \dots, p.$$

siendo  $v(\mu_i) = var(Y_i)$ .

- Las ecuaciones de verosimilitud depende de la distribución de  $Y_i$  solamente a través de su media y su varianza,  $\mu_i$  y  $v(\mu_i)$ .
- Dada una distribución tenemos determinada la relación entre media y varianza, esto es,  $v(\mu_i)$ .
- Wedderburn(1974) propuso asumir solamente una relación entre media y varianza en lugar de una distribución en la variable respuesta.
- Asumimos una función link y un predictor lineal.

- No asumimos una distribución para  $Y_i$ . En lugar de ello, asumimos

$$\text{var}(Y_i) = v(\mu_i)$$

para alguna función varianza  $v$ .

- Las ecuaciones que permiten obtener los estimadores cuasiverosímiles son las mismas.
- Pero no son ecuaciones de verosimilitud sin asumir una distribución.
- Wedderburn propuso utilizar estos estimadores incluso aunque no asumamos que  $Y_i$  están en la familia exponencial natural.
- La matriz de covarianzas asintótica es tiene la forma

$$(X' \hat{W} X)^{-1}$$

con

$$w_i = \left( \frac{\partial \mu_i}{\partial \eta_i} \right)^2 \frac{1}{\text{var}(Y_i)}$$

### 29.8.1 Sobredispersión en GLM Poisson y cuasiverosimilitud

- Una posibilidad es considerar

$$v(\mu_i) = \phi \mu_i,$$

para alguna constante  $\phi$ .

- En las ecuaciones que utilizamos para estimar (que no son de verosimilitud) se cancela  $\phi$ .
- Los estimadores son los mismos que los máximo verosímiles.
- [notaR/notaR036.pdf](#)

# Capítulo 30

## Meta-análisis

Trata del problema de combinar los resultados de distintos experimentos o análisis con el fin de obtener una conclusión única.

### 30.1 Método de Fisher

Supongamos que contrastamos una misma hipótesis nula,  $H_0$ ,  $k$  veces de un modo independiente (distintos conjuntos de datos obtenidos independientemente uno de otro). Si suponemos que el estadístico del contraste que estamos utilizando tiene una distribución continua entonces el p-valor sigue una distribución uniforme en el intervalo unitario  $[0, 1]$ ,  $P \sim U(0, 1)$  para cada uno de los tests. Una forma de combinar estos p-valores es la propuesta por [Ronald A. Fisher](#) y conocida como el método de Fisher.<sup>271</sup> Tenemos que los p-valores,  $P_1, \dots, P_k$  (antes de tomar las muestras) son independientes y con una distribución común  $P_i \sim U(0, 1)$ . Fisher propuso como forma de combinar estos valores el siguiente estadístico

$$X^2 = -2 \sum_{i=1}^k \ln P_i. \quad (30.1)$$

Asumiendo que la hipótesis nula  $H_0$  es cierta (bajo la hipótesis nula) y que los tests son independientes entonces el estadístico anterior se distribuye como una ji-cuadrado con  $2k$  grados de libertad.<sup>272</sup>

$$X^2 = -2 \sum_{i=1}^k \ln P_i \sim \chi_{2k}. \quad (30.2)$$

Notemos que un p-valor significativo significa un valor p pequeño y por lo tanto su logaritmo neperiano será muy pequeño (y negativo). Finalmente el estadístico  $X^2$  será grande si los  $p_i$  observados son pequeños, esto es, si nos indican que la hipótesis nula no es cierta. El p-valor asociado al test anterior está calculado *bajo* la hipótesis de tests independientes. ¿Y si son dependientes? La dependencia que cabe esperar es de tipo positivo, esto es, si son pequeños tenderán a ser todos pequeños. De este modo observaremos un p-valor en el test anterior menor al que observaríamos si realmente tuviéramos p-valores independientes.

### 30.2 Método de Stouffer

<sup>273</sup> Es un procedimiento muy relacionado con el método de Fisher. Si

<sup>271</sup> Fisher's method o Fisher's combined probability test.

<sup>272</sup> Si el estadístico tiene una distribución continua entonces  $P \sim U(0, 1)$ . Es fácil ver que  $-2 \ln P \sim Exp(1) = Ga(1, 1)$ , es decir,  $P$  se distribuye exponencial con media 1 que es una distribución gamma con parámetros  $(1, 1)$ . Entonces  $-\sum_{i=1}^k \ln P_i \sim Ga(k, 1)$  y finalmente  $-2 \sum_{i=1}^k \ln P_i \sim Ga(k, 1/2)$ . Pero una gamma con parámetros  $k$  y  $1/2$  no es más que una ji-cuadrado con  $2k$  grados de libertad.

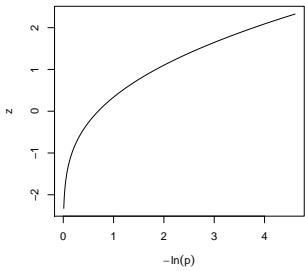
<sup>273</sup> Stouffer's Z-score method.

<sup>274</sup>  $\Phi(z) = \int_{-\infty}^z \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2}t^2}$   $P_i$  son los p-valores aleatorios originales y denotamos por  $\Phi$  la función de distribución de la normal estándar,<sup>274</sup> entonces definimos el Z-score con  $-\infty < z < +\infty$ .

$$Z = \frac{\sum_{i=1}^k Z_i}{\sqrt{k}}, \quad (30.3)$$

siendo  $Z_i = \Phi^{-1}(1 - P_i)$ .

Notemos que el método de Fisher suma los valores  $-2 \ln(p_i)$ . Por otro lado el Z-valor de Stouffer lo que propone es (por una constante) una suma de  $Z_i$  valores. En ambos casos sumamos la contribución de cada test que en un caso lo cuantificamos con  $-2 \ln(p_i)$  y en otro con  $\Phi^{-1}(1 - p_i)/\sqrt{k}$ . En la figura ?? comparamos los sumandos con el siguiente código.



```
p = seq(.01,.99,.001)
plot(-log(p),qnorm(1-p),type="l",xlab = expression(-ln(p)),
      ylab = "z")
```

Figura 30.1: Comparamos la aportación de cada test según método de Fisher y Stouffer. En abscisas  $\ln(p)$  y en ordenadas  $\Phi^{-1}(1 - p)$ .

# Capítulo 31

## Miscelánea

En el texto hemos utilizado una serie de técnicas y conceptos estadísticos. Es preferible, antes que insertar su explicación en el texto incluirlos en un capítulo aparte. Cada sección responde a un concepto distinto. No hay ningún tipo de continuidad entre las secciones. Es un capítulo auxiliar para el resto de los capítulos. Cuando explicamos los conceptos también indicamos cómo hacerlo con R.

### 31.1 Algoritmo bipesado de Tukey de un solo paso

Su denominación en la literatura estadística sería *One-Step Tukey's Biweight Algorithm*. Veamos cómo funciona. Tenemos unos datos  $x_1, \dots, x_n$  de los cuales pretendemos tener algún tipo de valor medio o de valor central que no esté muy afectado por valores anómalos como lo está, por ejemplo, la media muestral.

1. Calculamos la mediana  $m_x$  de los datos  $x_1, \dots, x_n$ .
2. Determinamos las distancias  $d_i = |x_i - m_x|$ .
3. Calculamos la mediana  $m_d$  de  $d_i$  con  $i = 1, \dots, n$ . Esta mediana  $m_d$  o mediana de las desviaciones absolutas se suele denotar como MAD (median absolute deviation). Es una medida de dispersión.

4. Consideramos<sup>1</sup>

$$u_i = \frac{x_i - m_x}{cm_d + \epsilon},$$

con  $i = 1, \dots, n$ . Los valores por defecto son  $c = 5$  y  $\epsilon = 0.0001$  (con el valor de  $\epsilon$  evitamos la división por cero).

5. Consideraremos una función biquadrada (bisquare) definida como

$$w(u) = \begin{cases} (1 - u^2)^2 & \text{si } |u| \leq 1 \\ 0 & \text{si } |u| > 1 \end{cases}$$

6. Se define el estimador en un solo paso como

$$T_{bi}(x_1, \dots, x_n) = \frac{\sum_{i=1}^n w(u_i)x_i}{\sum_{i=1}^n w(u_i)}.$$

---

<sup>1</sup> En la implementación original del procedimiento  $c = 1$  y  $\epsilon = 0$ .

Notemos que si sustituimos el valor  $m_x$  por  $T_b$  podríamos aplicar iterativamente el procedimiento hasta que se estabilice. No lo hacemos. Simplemente se hace una sola iteración.

**Ejemplo 31.1** *Supongamos que tenemos los datos siguientes*

```
x = c(12, 3, 45, 21, 34, 35, 21, 456)
```

*Y ahora podemos usar la función tukey.biweight del paquete [51, affy].*

```
library(affy)
tukey.biweight(x, c = 5, epsilon = 1e-04)

## [1] 25.1902
```

## 31.2 Median polish

Tenemos una matriz de datos tal que en la fila i y columna j tenemos  $y_{ij}$  con  $i = 1, \dots, I$  y  $j = 1, \dots, J$ . El procedimiento del median polish consiste en expresar estos valores de la siguiente forma

$$y_{ij} = m + a_i + b_j + e_{ij}$$

de forma que se verifique que:

- La mediana de  $\{a_1, \dots, a_I\}$  sea cero. Abreviadamente denotamos  $\text{Mediana}\{a_1, \dots, a_I\} = 0$ .
- $\text{Mediana}\{b_1, \dots, b_J\} = 0$ .
- $\text{Mediana}\{e_{i1}, \dots, e_{iJ}\} = 0$  para cada i.
- $\text{Mediana}\{e_{1j}, \dots, e_{IJ}\} = 0$  para cada j.

Pueden haber muchas soluciones para las ecuaciones y restricciones que acabamos de considerar. Consideraremos la siguiente matriz.

$$\begin{array}{ccc} y_{11} & \dots & y_{1J} \\ \vdots & & \vdots \\ y_{I1} & \dots & y_{IJ} \end{array}$$

El algoritmo median polish (propuesto por Tukey y Mosteller) consiste en aumentar la matriz con una columna y una fila adicionales. Consideremos la matriz aumentada siguiente

$$\begin{array}{ccc|c} e_{11} & \dots & e_{1J} & a_1 \\ \vdots & & \vdots & \vdots \\ e_{I1} & \dots & e_{IJ} & a_I \\ \hline b_1 & \dots & b_J & m \end{array}$$

El procedimiento actúa iterativamente sobre esta matriz aumentada del siguiente modo:

1. Inicializamos el procedimiento con todos los  $a_i = 0$  y  $b_j = 0$ ,  $m = 0$  y  $e_{ij} = y_{ij}$ .

2. Calculamos la mediana de las columnas de la 1 a la I.
3. Restamos dicha mediana a cada elemento de las filas de la 1 a la I y se la sumamos a la columna I+1, esto es, se la sumamos a los valores  $a_i$  y m.
4. Calculamos la mediana de cada columna para
5. Restamos dicha mediana a cada elemento de la columna.
6. Sumamos, para cada columna j, la mediana de la columna al valor de la última columna.
- 7.

**Ejemplo 31.2** Consideremos los datos.

```
y = matrix(c(18,11,8,21,4,13,7,5,16,7,15,6,7,16,6,19,15,12,18,5),
            ncol=5)
rownames(y) = paste("chip",1:4)
colnames(y) = paste("sonda",1:5)
```

Veamos los datos.

```
y
##          sonda 1 sonda 2 sonda 3 sonda 4 sonda 5
## chip 1     18      4     16      7     15
## chip 2     11     13      7     16     12
## chip 3      8      7     15      6     18
## chip 4     21      5      6     19      5
```

Y aplicamos una median polish.

```
medpolish(y)

## 1: 82
## Final: 82
##
## Median Polish Results (Dataset: "y")
##
## Overall: 10.5
##
## Row Effects:
## chip 1 chip 2 chip 3 chip 4
##      5      2     -2     -4
##
## Column Effects:
## sonda 1 sonda 2 sonda 3 sonda 4 sonda 5
##     1.0    -1.5     0.0     0.5    -0.5
##
## Residuals:
##          sonda 1 sonda 2 sonda 3 sonda 4 sonda 5
## chip 1     1.5    -10     0.5     -9      0
## chip 2    -2.5      2    -5.5      3      0
## chip 3    -1.5      0     6.5     -3     10
## chip 4    13.5      0    -0.5     12     -1
```

### 31.3 Datos faltantes

Tenemos nuestra matriz de datos de expresión  $X$  y tenemos datos faltantes en algunas características para algunas muestras. Una posibilidad es no perder datos utilizando alguna técnica que impute un valor a los datos que no tenemos. En § 31.3.1 explicamos el procedimiento utilizado por el método SAM (§ 9.1) y por el método GSA (§ 17.11).

#### 31.3.1 Algoritmo de normalización del k-vecino más próximo

Fijamos el número de vecinos  $k$  a utilizar en la imputación. Por defecto se toman  $k = 10$ . El procedimiento es el siguiente:

1. Para cada gen  $i$  con al menos un valor faltante:
  - (a) Denotamos por  $S_i$  las muestras para las cuales *conocemos* la expresión del gen.
  - (b) Determinamos los  $k$ -vecinos más próximos del gen  $i$  utilizando solamente las muestras  $S_i$  para calcular la distancia euclídea. Si los otros genes tienen datos faltantes en las muestras  $S_i$  se utilizan las muestras de las cuales se conoce su expresión.
  - (c) Para las muestras que no están en  $S_i$  asignamos el promedio de los  $k$ -vecinos más próximos. Si alguno de estos  $k$ -vecinos no tiene alguna expresión entonces hacemos la media de los que disponemos.
2. Si un gen tiene todavía un dato faltante entonces asignamos la expresión media de la muestra considerada.

El algoritmo anterior puede ser lento cuando tenemos muchos genes. Con objeto de acelerar el proceso de asignación de datos faltantes se combina con el siguiente procedimiento de agrupamiento recursivo basado en el k-medias (§ 14.7.1).

1. Si el número de genes  $p$  es mayor que  $p_{max}$  (por defecto, vale 1500):
  - (a) Ejecutamos una clasificación del procedimiento k-medias con  $k = 2$  al conjunto de los genes (clasificamos genes y no muestras). Las distancias entre los genes están basadas en las muestras para las cuales se conocen ambas expresiones.
  - (b) Formamos dos matrices de expresión una con cada conjunto de genes. Aplicamos recursivamente el paso anterior.
2. Si  $p < p_{max}$  entonces aplicamos el procedimiento de imputación utilizando el promedio de los  $k$ -vecinos más próximos en cada uno de los grupos encontrados previamente.

El procedimiento fue propuesto en [139] y está implementado en la función `impute::impute.knn`.

## 31.4 Datos multimodales

### 31.4.1 TCGA

<sup>275</sup> The Cancer Genome Atlas. Podemos bajar datos con los conteos ya calculados en <https://tcga-data.nci.nih.gov/tcga/tcgahome2.jsp> y en <https://cghub.ucsc.edu/> podemos conseguir datos originales.

Los datos con los que vamos a trabajar proceden de TCGA.<sup>275</sup> Podemos utilizar el paquete [84, RTCGA] permite el manejo de datos de esta base de datos de un modo simple.

Lo que sigue está tomado (en gran parte) de la viñeta del paquete [162, RnaSeqSampleSizeData].

Los datos se pueden bajar de <https://tcga-data.nci.nih.gov/tcga/tcgadownload.jsp>. Por ejemplo, en [https://tcga-data.nci.nih.gov/tcgafiles/ftp\\_auth/distro\\_ftputers/anonymous/tumor/brcacgcunc.edu/illuminahiseq\\_rnaseqv2/rnaseqv2/](https://tcga-data.nci.nih.gov/tcgafiles/ftp_auth/distro_ftputers/anonymous/tumor/brcacgcunc.edu/illuminahiseq_rnaseqv2/rnaseqv2/) se tiene las distintas versiones de las muestras de BRCA. Los conteos están en los ficheros `.rsem.genes.results`.



## **Parte VIII**

# **Investigación reproducible**



## Capítulo 32

# Por favor, no publiquéis más

### 32.1 Sobre la gran cantidad de publicaciones ¿innecesarias?

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.<sup>276</sup> Al final de esta década 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 tienen la culpa. 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.<sup>277</sup>

Evidentemente el planeta no tiene un problema de superpoblación de personas. Es de publicaciones científicas.

En campos como las Matemáticas o la Estadística 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. Todo está a la vista. 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. Es muy recomendable leer [12]. Reproduce 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.

Y debiera de ser posiblemente candidato a algún premio importante.

<sup>276</sup> Asumimos que no es un superhéroe con algún superpoder.

<sup>277</sup> Un optimista diría que doblamos el conocimiento que la humanidad ha logrado atesorar cada nueve años.

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. Y desde el principio.

En muchas ocasiones biólogos o médicos o biotecnólogos o bioquímicos o ... me han indicado que querían aprender más de 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 Estadística es muy fácil. Se coge un libro sencillo. El 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.

Es una opinión.

## 32.2 Investigación reproducible

<sup>278</sup> Reproducible research

<sup>279</sup> Literate programming

<sup>280</sup> En ciencias experimentales.

Tratamos sobre **investigación reproducible**.<sup>278</sup> relacionado con la programación comentada<sup>279</sup> aunque no son sinónimos.

Estamos acostumbrados a que los investigadores<sup>280</sup> en sus publicaciones comenten los resultados obtenidos en sus investigaciones. Han obtenido unos datos, los han analizado y, finalmente, los han interpretado y discutido. Hemos 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? En gran parte, es una cuestión de fe. ¿Dónde están los datos? Exactamente, no aproximadamente: ¿qué análisis de los datos han realizado? En otras palabras, déme 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 noticia de la existencia de la investigación.

Por investigación reproducible entendemos procedimientos que permitan *reproducir* la investigación completa.

Personalmente añadiría que los distintos elementos han de ser totalmente libres. Los datos han de estar a disposición de la comunidad<sup>281</sup> 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.<sup>282</sup>

<sup>281</sup> Una vez se han publicado los trabajos.

<sup>282</sup> Todo lo que he usado en este manual es libre. No se ha utilizado software propietario ni datos que no estén a disposición de todos en algún repositorio público.

Este manual es un ejemplo de lo que [158] llama un **documento dinámico**. Es un documento en donde se combinan las explicaciones metodológicas junto con el código que las implementa.<sup>283</sup> La herramienta fundamental utilizada es el paquete [159, knitr].<sup>284</sup> En [158] tenemos una detallada y amena exposición que recomiendo vivamente. Dos son las opciones para escribir el texto, L<sup>A</sup>T<sub>E</sub>Xy Markdown. Este manual está hecho con L<sup>A</sup>T<sub>E</sub>X. Es un documento largo con muchas expresiones matemáticas que hacían aconsejable (imprescindible para un estadístico como yo) utilizar este lenguaje de marcas, el primero de todos y posiblemente el mejor. Sin embargo, los documentos con ejemplos asociados al manual están hechos con otro lenguaje de marcas (ligero) conocido como Markdown. Utilizamos RMarkdown ([6, rmarkdown]).

De un modo genérico son de interés las referencias [58, 57] así como las siguientes direcciones:

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 L<sup>A</sup>T<sub>E</sub>X entonces una (muy) buena referencia [106].

En este documento nos centramos en R y por ello la opción para investigación reproducible es combinar R bien con Markdown bien con L<sup>A</sup>T<sub>E</sub>X. Sin embargo, en otros lenguajes hay otras opciones similares y tan buenas como esta. Es recomendable leer <https://blog.ouseful.info/2017/11/15/programming-meh-lets-teach-how-to-write-computational-essays-instead/>.

## 32.3 Markdown

Es un lenguaje de marcas ligero (minimal sería más correcto). Pretende ser una forma rápida de escribir HTML. Segun su autor, [John Gruber](#), “HTML is a publishing format; Markdown is a writing format.”

La dirección indicada contiene una muy buena exposición de este lenguaje de marcas.

De las distintas versiones del lenguaje Markdown la más interesante para nosotros es la asociada a [pandoc](#), **Pandoc’s Markdown**.

## 32.4 RMarkdown

El paquete [6, rmarkdown] incorpora una implementación del lenguaje de marcas Markdown y utilizado conjuntamente con [159, knitr] permite generar unos muy buenos informes. La página <http://rmarkdown.rstudio.com/> es el mejor lugar para aprender a manejarlo.

## 32.5 RStudio y RMarkdown

Es la mejor opción. RStudio permite elegir esta opción y lleva incorporados en los menús opciones para generar informes en formatos html, pdf e incluso<sup>285</sup> en formato Word.

<sup>283</sup> Sobre la obtención de los datos hay que consultar las referencias bibliográficas.

<sup>284</sup> Hay miles de paquetes de R pero pocos como este maravilloso paquete.

<sup>285</sup> En el colmo de las locuras.

## 32.6 emacs y RMarkdown

El editor *emacs* puede ser configurado para trabajar con R y [Mark-down](#). Consiste en utilizar el paquete [polymode](#). Siguiendo lo indicado en esta página los pasos a seguir (y funciona) son los siguientes:

1. En `/home/gag/emacs.d/`<sup>1</sup> ejecutamos

```
git clone https://github.com/vitoshka/polymode.git
```

2. 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)
```

---

<sup>1</sup>Sustituimos por el directorio personal correspondiente.

# Capítulo 33

## Lo que no se debe hacer pero se hace

En un texto de Estadística aplicada a datos ómicos parece que lo natural es explicar solamente cómo diseñar un experimento y cómo analizar de un modo correcto los datos obtenidos. Pero la experiencia le dice a uno que raramente preguntan los experimentadores previamente a realizar su estudio. Luego no suelen encontrar a algún alma que les intente analizar los datos obtenidos sin pensar un poco en el diseño previo. Esto es así.

En este tema hablamos de barbaridades. Pequeñas y grandes barbaridades. Posiblemente las más peligrosas son las pequeñas por lo frecuentes. Ejemplos que me he ido encontrando de lo que se hace y (en mi opinión) no se debe hacer. Responde a una experiencia personal. Vamos a explicar en negativo (lo incorrecto) y no en positivo (lo correcto) como se pretende en el resto del texto. No es una mala opción. Si todo el texto se hiciera así sería muy largo ya que el número de incorrecciones que se pueden cometer es infinito.

### 33.1 Mejor no usarlo

Un breve listado de cosas que hay que evitar.

Esperemos que, al menos, sea un infinito numerable. Pero no lo sé. Podría ser de un cardinal estrictamente mayor. Yo diría que con probabilidad uno lo es.

**Diagramas de sectores** Los periódicos los aman. Los experimentadores también. Y no lo entiendo. Lo más habitual entre los estadísticos es odiarlos. En mi caso con toda mi alma. No usarlos. Es mucho más claro un diagrama de barras como opción alternativa si queremos un dibujo de frecuencias. La comparación entre las frecuencias de las distintas categorías es mucho más simple. Otra vez, evitarlos.

**Histogramas** Los histogramas como una estimación de la función de densidad de la variable (aleatoriedad) que estamos estudiando tienen una gran tradición en Estadística. Su principal problema es que la elección del número de clases es crítica. El dibujo cambia mucho cuando modificamos el número de clases. Depende demasiado de una elección previa del usuario. Los programas llevan procedimientos que fijan este número. Es mejor utilizar un estimador no paramétrico de la densidad o estimadores basados en funciones kernel. En el texto lo usamos. Con grandes cantidades de datos

dan una muy buena (y robusta) estimación de la función de densidad de la variable aleatoria.

### 33.2 Combinando información de distintos experimentos

Las (buenas) revistas científicas han introducido el buen hábito de obligar a que se depositen los datos en repositorios público. Este texto utiliza datos de estos repositorios. Esto es muy bueno. Además debiera de obligarse a poner el código que se ha desarrollado en el lenguaje o lenguajes que sean para poder reproducir el estudio. Este último punto todavía no es habitual pero creo que lo será en no mucho tiempo. Al menos en buenas revistas. Tener muchos bancos de datos disponibles es bueno. Pero puede ser malo, de hecho, puede ser malísimo si no se usan con sensatez. Ha aparecido otra bibliografía que utiliza estos datos *prestados* para realizar análisis conjuntos o **meta-análisis**. El meta-análisis es una parte de la Estadística que se explica poco. No suele estar en los programas de las asignaturas básicas. Y podría estar. Y debiera de estar porque se están cometiendo auténticos crímenes (contra la humanidad).

En datos ómicos hay una cierta confusión entre qué son las variables y qué son las observaciones. Y a esta confusión no ayuda el costumbre de representar las matrices de expresión traspuestas. La matriz de datos tiene habitualmente las filas correspondiendo con las observaciones y las columnas correspondiendo a las variables que observamos en dichas observaciones. Sin embargo, con datos ómicos es muy habitual (y en este texto seguimos este hábito) representar la matriz de expresión de modo que las columnas son observaciones o muestras y las filas son las variables (genes, sondas, grupos de sondas, etc). Las columnas o muestras u observaciones son independientes mientras que las filas no lo son. Es conveniente seguir esta costumbre porque los paquetes de Bioconductor lo hacen y son nuestra fuente principal de software estadístico.

Cuando combinamos información de distintos experimentos es claro que las variables pueden parecer ser las mismas pero no serlo. Una variable puede estar cuantificando información sobre un mismo gen pero tiene porqué tener una misma distribución de probabilidad. No tenemos muestras aleatorias en donde se asume una distribución común. Por ejemplo, nos colocamos en la mejor de las situaciones. Cada experimento utiliza el mismo chip. ¿Podemos simplemente combinar las columnas y metadatos? Los distintos experimentos se han realizado en distintos laboratorios. ¿Sus protocolos experimentales son los mismos? ¿El material biológico se ha seleccionado de la misma forma? ¿Hay influencia del laboratorio? Son preguntas que si tienen respuesta negativa significa que no observamos lo mismo aunque sea el mismo chip y el mismo diseño.

Por supuesto si el chip no es el mismo no tiene sentido combinar la información considerando que tenemos una muestra mayor y quedándonos con aquellos genes que están en todos los chips. Aunque hablamos de unos mismos genes no tenemos la misma variable aleatoria. Si la co-variable que utilizamos para evaluar los perfiles de expresión es la **misma** entonces podemos pensar en realizar un meta-análisis utilizando los p-valores pero no combinando efectos. No tenemos la misma variable.

¿Y con RNA-seq? La cosa no es tan simple de evaluar. Es claro que

el tema de los protocolos han de ser los mismos. Y luego hay que utilizar los mismos alineadores y con los mismos parámetros. Hay que considerar los posibles errores sistemáticos producidos por los procedimientos de alineamiento. En este sentido si se utilizan datos procesados disponibles en la red no parece razonable combinar esta información. Se debe de utilizar el mismo material base, con lecturas de la misma longitud, conseguidas con el mismo producto.

Una práctica (que espero no se transforme en habitual) consiste en construir bancos de datos reunión datos de distintos experimentos.

Esta confusión hace que a veces se toman distintos experimentos. De cada uno de estos experimentos se observa una variable distinta asociada a un mismo gen. Y se construye una matriz de datos donde en filas tenemos genes y en columnas tenemos variables medidas en distintos experimentos. Las variables que observamos en distintas experimentaciones aunque se refieran a un mismo gen son observaciones independientes que además no están observadas en las mismas condiciones experimentales. Por lo tanto no podemos estudiar la dependencia que pueda existir entre ellas. Insisto, son valores independientes de distribuciones distintas y la posible dependencia que observemos será puro ruido. Y se hace.



## Apéndice A

# Soluciones ejercicios

**Solución (Ej. 1) —** `1.mean(x)`

```
## [1] 5
var(x)
## [1] 4.484848
sd(x)
## [1] 2.117746
```

`2.sum(x == 3)`

```
## [1] 16
```

`3.y = sort(x)`

```
cumsum(y)
## [1] 0 1 3 5 7 9 11 13 15 17 20
## [12] 23 26 29 32 35 38 41 44 47 50 53
## [23] 56 59 62 65 69 73 77 81 85 89 93
## [34] 97 101 105 109 113 117 121 125 129 133 137
## [45] 141 146 151 156 161 166 171 176 181 186 191
## [56] 196 201 206 211 216 221 226 232 238 244 250
## [67] 256 262 268 274 280 286 292 298 304 310 316
## [78] 323 330 337 344 351 358 365 372 379 387 395
## [89] 403 411 419 427 435 443 452 461 470 479 489
## [100] 500
```

**Solución (Ej. 2) —** `1.sum(x >= 39.4)`

```
## [1] 23328
```

O bien podemos ver los TRUE de

```



```

4.Significa que son mayores o iguales a 40 y estrictamente menores que 41. Por tanto podemos

```

sum(x >= 40 & x < 41)
## [1] 638

```

Otra posibilidad es usar la función base::floor.

```

sum(floor(x) == 40)
## [1] 638

```

**Solución (Ej. 3)** — 1.ncol(x)

```

## [1] 122

```

```

2.which.max(apply(x,2,mean))
## [1] 58

```

```

3.which.min(apply(x,1,mean))
## [1] 143

```

```

4.which.min(apply(x,1,IQR))
## [1] 282

```

```

5.sum(x[,1] >= x[,2])
## [1] 1183

```

```

6.rowmean = apply(x,1,mean)
sum(x[,45] >= rowmean)
## [1] 1220

```

**Solución (Ej. 4)** — Empezamos leyendo los datos y transformando en factor la variable golub.cl.

```
data(golub, package="multtest")
golub.fac = factor(golub.cl, levels=0:1, labels=c("ALL", "AML"))
```

1.La expresión media en cada condición la tenemos con

```
m1 = apply(golub, 1, mean)
```

```
2.sd1 = apply(golub, 1, sd)
```

```
3.df = data.frame(x0= m1, y0 =sd1)
pdf(paste0(dirTamiFigures, "tt171113a.pdf"))
ggplot(df, aes(x=x0, y=y0)) + geom_point()
dev.off()
```

**Solución (Ej. 7) — 1.ncol(x)**

```
## [1] 5
```

```
2.apply(x, 2, max)
```

```
3.minrow = apply(x, 1, min)
which.min(minrow)
## [1] 78
```

```
4.which.max(minrow)
```

```
## [1] 50
```

```
5.quantile(x[, 1], probs=0.34)
##          34%
## 0.3489822
```

```
6.meanrow = apply(x, 1, mean)
quantile(x[, 1], probs=0.34)
##          34%
## 0.3489822
```

```
7.min(x)
```

```
## [1] 0.0004653491
```

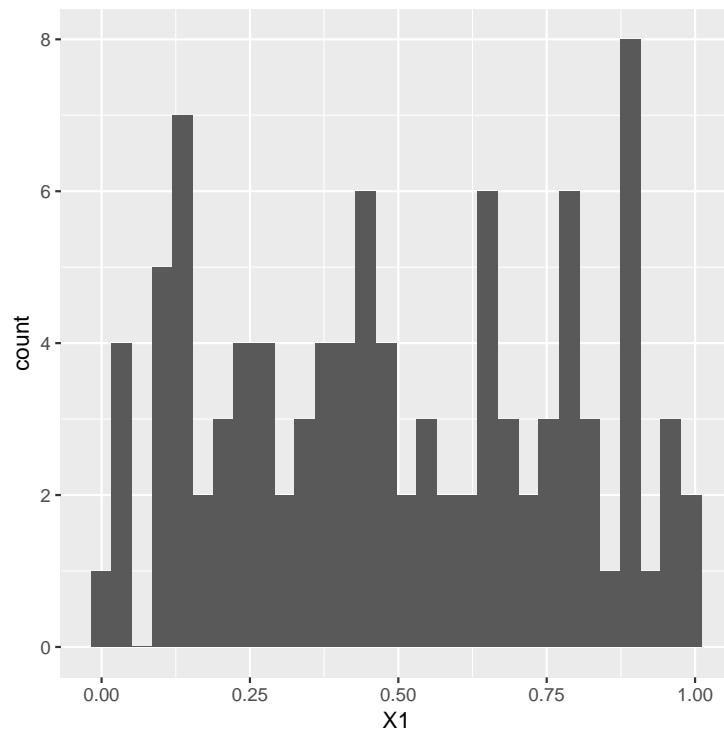
```
max(x)
```

```
## [1] 0.9994045
```

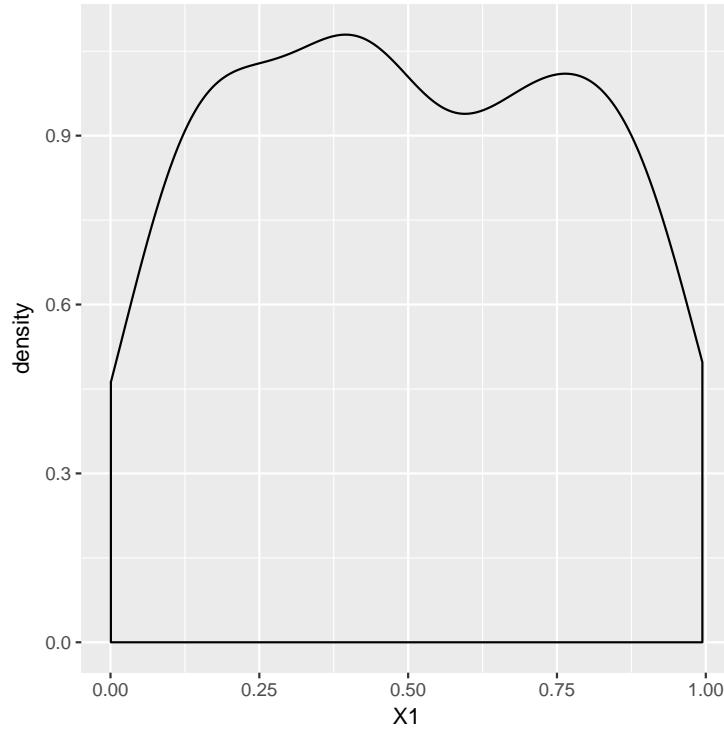
```
mean(x)
```

```
## [1] 0.4952837
```

```
8.pacman::p_load(ggplot2)
df= data.frame(x)
ggplot(df,aes(x=x1)) + geom_histogram()
```



```
9.pacman::p_load(ggplot2)
df= data.frame(x)
ggplot(df,aes(x=x1)) + geom_density()
```



```

10.sum(x>.12)
## [1] 448

11.x1 = apply(x< .12,1,sum)

12.IQR(x1)
## [1] 1

13.sum(x[,1] > x[,2])
## [1] 51

```

**Solución (Ej. 8) —** 1.data(golub, package="multtest")

```

df = t(golub)
df = data.frame(df)

```

```
2.names(df) = paste0("probe",1:ncol(df))
```

```
3.type = factor(golub.cl,levels=0:1,labels=c("ALL","AML"))
df= data.frame(df,type)
```

**Solución (Ej. 9) —** a100218a = function(x)

```
list(media = mean(x),varianza = var(x))
```

```
Solución (Ej. 10) — a100218a = function(x){
  q0 = quantile(x,probs = c(alpha,1-alpha))
  sel0 = which(x >= q0[1] & x <= q0[2])
  list(media_ajustada = mean(x[sel0]),varianza_ajustada = var(x[sel0]))
}
```

**Solución (Ej. 11)** — Los dos primeros paquetes [72, ggrepify] y [48, faraway] están en CRAN y por lo tanto los instalamos con utils::install.packages. Nos pedirá que elijamos repositorio. Elegimos el que nos guste.

```
pkgs = c("ggfortify", "faraway")
install.packages(pkgs)
```

Los paquetes [111, multtest] y [34, edgeR] son de Bioconductor. Se instalan con

```
source("https://bioconductor.org/biocLite.R")
pkgs = c("multtest", "edgeR")
biocLite(pkgs)
```

**Solución (Ej. 12)** — Cargamos la base de datos de tipo OrgDb correspondiente a homo sapiens.

```
pacman::p_load(org.Hs.eg.db)
```

Buscamos cómo acceder al SYMBOL o nombre abreviado del gen.

```
keytypes(org.Hs.eg.db)
## [1] "ACNUM"          "ALIAS"           "ENSEMBL"
## [4] "ENSEMBLPROT"    "ENSEMBLTRANS"   "ENTREZID"
## [7] "ENZYME"         "EVIDENCE"       "EVIDENCEALL"
## [10] "GENENAME"       "GO"              "GOALL"
## [13] "IPI"            "MAP"             "OMIM"
## [16] "ONTOLOGY"       "ONTOLOGYALL"   "PATH"
## [19] "PFAM"           "PMID"           "PROSITE"
## [22] "REFSEQ"         "SYMBOL"         "UCSCKG"
## [25] "UNIGENE"        "UNIPROT"
```

Por tanto los nombres abreviados de los genes los tendremos con

```
keys(org.Hs.eg.db, keytype="SYMBOL")
```

Una opción muy simple puede ser

```
(posBRCA1 = which(keys(org.Hs.eg.db, keytype="SYMBOL") == "BRCA1"))
## [1] 554
(posBRCA2 = which(keys(org.Hs.eg.db, keytype="SYMBOL") == "BRCA2"))
## [1] 556
posBRCA = c(posBRCA1,posBRCA2)
```

Supongamos que no tenemos claro su nombre exacto y sabemos que es BRCA seguido de algo más. Podemos mirar aquellos genes que empiezan por BRCA.

```
(beginBRCA = grep("^BRCA", keys(org.Hs.eg.db, keytype="SYMBOL")))
## [1] 554 556 6410 14110 23877
```

¿Cuáles son sus nombres abreviados?

```
keys(org.Hs.eg.db, keytype="SYMBOL") [beginBRCA]
## [1] "BRCA1"    "BRCA2"    "BRCATA"   "BRCA3"
## [5] "BRCA1P1"
```

Construimos un data.frame con la información pedida.

```
symbol = keys(org.Hs.eg.db, keytype="SYMBOL") [posBRCA]
entrezid = keys(org.Hs.eg.db, keytype="ENTREZID") [posBRCA]
ensembl = keys(org.Hs.eg.db, keytype="ENSEMBL") [posBRCA]
go = keys(org.Hs.eg.db, keytype="GO") [posBRCA]
data.frame(symbol, entrezid, ensembl, go)
## symbol entrezid ensembl go
## 1 BRCA1      672 ENSG00000125730 GO:0006101
## 2 BRCA2      675 ENSG00000244731 GO:0008610
```

**Solución (Ej. 13)** — pacman::p\_load(org.Mm.eg.db)  
`(posBRCA1 = which(keys(org.Mm.eg.db, keytype="SYMBOL") == "BRCA1"))`  
`## integer(0)`

**Solución (Ej. 21)** — 1. Seleccionamos:

```
data(gse20986, package="tamidata")
```

¿Qué columnas ocupan?

```
pacman::p_load("Biobase")
pData(gse20986)[, "tissue"] == "iris" | pData(gse20986)[, "tissue"] == "huvec"
```

De otro modo:

```
(sel = which(is.element(pData(gse20986)[, "tissue"], c("iris", "huvec"))))
## [1] 1 4 6 10 11 12
```

Seleccionamos las muestras.

```
eset = gse20986[, sel]
```

Con el siguiente código eliminamos las categorías que no utilizamos.

```
pData(eset)[, "tissue"] = factor(pData(eset)[, "tissue"])
```

2. Con genefilter::rowttests.

```
tt = genefilter::rowttests(eset, pData(eset)[, "tissue"])
```

```
3.tt1 = multtest::mt.rawp2adjp(tt[, "p.value"], "BH")
```

Los p-valores ajustados serían

```
tt1$adjp[, 2]
```

4. Tenemos como significativos

```
sum(tt1$adjp[,2] < 0.05,na.rm = TRUE)
## [1] 1828
```

que ocupan las siguientes posiciones en la matriz original.

```
sigIH = tt1$index[1:1828]
```

```
5.sel = which(is.element(pData(gse20986),c("tissue","retina","huvec")))
eset = gse20986[,sel]
pData(eset)[,"tissue"] = factor(pData(eset)[,"tissue"])
tt = genefilter::rowttests(eset,pData(eset)[,"tissue"])
tt1 = multtest::mt.rawp2adjp(tt[, "p.value"], "BH")
sigRH = tt1$index[1:sum(tt1$adjp[,2] < 0.05,na.rm = TRUE)]
```

```
6.sel = which(is.element(pData(gse20986),c("tissue","coroides","huvec")))
eset = gse20986[,sel]
pData(eset)[,"tissue"] = factor(pData(eset)[,"tissue"])
tt = genefilter::rowttests(eset,pData(eset)[,"tissue"])
tt1 = multtest::mt.rawp2adjp(tt[, "p.value"], "BH")
sigCH = tt1$index[1:sum(tt1$adjp[,2] < 0.05,na.rm = TRUE)]
```

```
7.sigIR = intersect(sigIH,sigRH)
sigIRC = intersect(sigIR,sigCH)
```

Y el resultado (no mostrado) es

```
sigIRC
```

Que corresponde con los genes.

```
featureNames(gse20986[sigIRC,])
```

**Solución (Ej. 22)** — Leemos los datos.

```
data(gse1397,package="tamidata")
eset = gse1397
```

```
1.tt = genefilter::rowttests(eset,pData(eset)[,"type"])
```

**Solución (Ej. 25)** —

```
1.pacman::p_load("SummarizedExperiment","GenomicRanges","edgeR","multiTest")
data(PRJNA297664,package="tamidata")
se = PRJNA297664
rm(PRJNA297664)
pvalor13 = binomTest(assay(se)[,1], assay(se)[,3])
```

```
2.tt = multtest::mt.rawp2adjp(pvalor13, "BH")
pvalor13.BH = tt$adjp[, "BH"]
```

```
3.numsig = sum(pvalor13.BH < 0.001)
sig13 = rownames(se)[tt$index[1:numsig]]  
  
4.pvalor25 = binomTest(assay(se)[,2], assay(se)[,5])
tt = multtest::mt.rawp2adjp(pvalor25, "BH")
pvalor25.BH = tt$adjp[, "BH"]
numsig = sum(pvalor25.BH < 0.001)
sig25 = rownames(se)[tt$index[1:numsig]]  
  
5.intersect(sig13,sig25)
```



# Bibliografía

- [1] H. Abdi y L. J. Williams. «Principal component analysis». En: *Wiley Interdisciplinary Reviews: Computational Statistics* 2.4 (2010), págs. 433-459. url: </home/gag/BIBLIOGRAFIA/REPRINTS/AbdiWilliams10.pdf>.
- [2] Sudipta Acharya, Sriparna Saha y N. Nikhil. «Unsupervised gene selection using biological knowledge : application in sample clustering». En: *BMC Bioinformatics* 18.1 (nov. de 2017), pág. 513. issn: 1471-2105. doi: [10.1186/s12859-017-1933-0](https://doi.org/10.1186/s12859-017-1933-0). url: <https://doi.org/10.1186/s12859-017-1933-0>.
- [3] A. Agresti. *Categorical Data Analysis*. Second. Wiley, 2002. url: [/home/gag/BIBLIOGRAFIA/MISLIBROS/Agresti\\_Alan\\_Categorical\\_Data\\_Analysis\\_2nd\\_ed\\_2002.pdf](/home/gag/BIBLIOGRAFIA/MISLIBROS/Agresti_Alan_Categorical_Data_Analysis_2nd_ed_2002.pdf).
- [4] A. Agresti. *Categorical Data Analysis*. Third. Wiley-Interscience, 2013. url: <http://www.stat.ufl.edu/~aa/cda/cda.html>.
- [5] Adrian Alexa y Jorg Rahnenfuhrer. *topGO: Enrichment Analysis for Gene Ontology*. R package version 2.30.0. 2016.
- [6] JJ Allaire y col. *rmarkdown: Dynamic Documents for R*. R package version 1.8. 2017. url: <https://CRAN.R-project.org/package=rmarkdown>.
- [7] Simon Anders y Wolfgang Huber. «Differential expression analysis for sequence count data». En: *Genome Biology* 11.10 (2010), R106. issn: 1465-6906. doi: [10.1186/gb-2010-11-10-r106](https://doi.org/10.1186/gb-2010-11-10-r106). url: <http://genomebiology.com/2010/11/10/R106>.
- [8] S. Anders y col. «Count-based differential expression analysis of RNA sequencing data using R and Bioconductor». En: *Nat. Protocols* 8.9 (2013), págs. 1765-1786. doi: [10.1038/nprot.2013.099](https://doi.org/10.1038/nprot.2013.099).
- [9] John D. Storey with contributions from Andrew J. Bass, Alan Dabney y David Robinson. *qvalue: Q-value estimation for false discovery rate control*. R package version 2.10.0. 2015. url: <https://github.com/jdstorey/qvalue>.
- [10] Guillermo Ayala. *tami: Statistical Bioinformatics*. R package version 0.9. 2018. url: <http://www.uv.es/ayala/docencia/tami>.
- [11] Guillermo Ayala. *tamidata: Data sets for Statistical Bioinformatics*. R package version 0.5. 2017. url: <http://www.uv.es/ayala/docencia/tami>.

- [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](https://doi.org/10.1038/533452A). url: <http://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970>.
- [13] Andrew J. Bass y col. *biobroom: Turn Bioconductor objects into tidy data frames*. R package version 1.10.0. 2017. url: <https://github.com/StoreyLab/biobroom>.
- [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. url: <http://www.jstor.org/stable/2346101>.
- [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] J. M. Bland y D. G. Altman. «Statistical methods for assessing agreement between two methods of clinical measurement». En: *Lancet* (1986), págs. 307-310. url: <http://www-users.york.ac.uk/~mb55/meas/ba.htm>.
- [17] Ben Bolstad. *affyPLM: Methods for fitting probe-level models*. R package version 1.54.0. 2017. url: <https://github.com/bmbolstad/affyPLM>.
- [18] Ben Bolstad. «Methods in Microarray Normalization». En: Drug Discovery Series 10. CRC Press, 2008. Cap. 3, págs. 41-60.
- [19] B.M. Bolstad y col. «A comparison of normalization methods for high density oligonucleotide array data based on variance and bias». En: *Bioinformatics* 19.2 (2003), págs. 185-193. doi: [10.1093/bioinformatics/19.2.185](https://doi.org/10.1093/bioinformatics/19.2.185). eprint: <http://bioinformatics.oxfordjournals.org/content/19/2/185.full.pdf+html>. url: <http://bioinformatics.oxfordjournals.org/content/19/2/185.abstract>.
- [20] Anne-Laure Boulesteix y Martin Slawski. «Stability and aggregation of ranked gene lists». En: *Briefings in Bioinformatics* 10.5 (2009), págs. 556-568. doi: [10.1093/bib/bbp034](https://doi.org/10.1093/bib/bbp034). eprint: <http://bib.oxfordjournals.org/content/10/5/556.full.pdf+html>. url: <http://bib.oxfordjournals.org/content/10/5/556.abstract>.
- [21] Richard Bourgon, Robert Gentleman y Wolfgang Huber. «Independent filtering increases detection power for high-throughput experiments». En: *Proceedings of the National Academy of Sciences* 107.21 (2010), págs. 9546-9551. doi: [10.1073/pnas.0914005107](https://doi.org/10.1073/pnas.0914005107). eprint: <http://www.pnas.org/content/107/21/9546.full.pdf+html>. url: <http://www.pnas.org/content/107/21/9546.abstract>.
- [22] Rosemary Braun, Leslie Cope y Giovanni Parmigiani. «Identifying differential correlation in gene/pathway combinations». En: *BMC Bioinformatics* 9.1 (2008), pág. 488. issn: 1471-2105. doi: [10.1186/1471-2105-9-488](https://doi.org/10.1186/1471-2105-9-488). url: <http://www.biomedcentral.com/1471-2105/9/488>.

- [23] Andrew C Browning y col. «Comparative gene expression profiling of human umbilical vein endothelial cells and ocular vascular endothelial cells». En: *British Journal of Ophthalmology* 96.1 (2012), págs. 128-132. doi: [10.1136/bjophthalmol-2011-300572](https://doi.org/10.1136/bjophthalmol-2011-300572). eprint: <http://bjophthalmol.bmjjournals.com/content/96/1/128.full.pdf+html>. url: <http://bjophthalmol.bmjjournals.com/content/96/1/128.abstract>.
- [24] Marc Carlson. *GO.db: A set of annotation maps describing the entire Gene Ontology*. R package version 3.5.0. 2017.
- [25] Marc Carlson. *hgu133a.db: Affymetrix Human Genome U133 Set annotation data (chip hgu133a)*. R package version 3.2.3. 2016.
- [26] Marc Carlson. *hgu133plus2.db: Affymetrix Human Genome U133 Plus 2.0 Array annotation data (chip hgu133plus2)*. R package version 3.2.3. 2016.
- [27] Marc Carlson. *hgu95av2.db: Affymetrix Human Genome U95 Set annotation data (chip hgu95av2)*. R package version 3.2.3. 2016.
- [28] Marc Carlson. *org.Hs.eg.db: Genome wide annotation for Human*. R package version 3.5.0. 2017.
- [29] Marc Carlson. *org.Mm.eg.db: Genome wide annotation for Mouse*. R package version 3.5.0. 2017.
- [30] M. Carlson y col. *GenomicFeatures: Tools for making and manipulating transcript centric annotations*. R package version 1.30.0. 2017.
- [31] Benilton Carvalho y Rafael Irizarry. *oligo: Preprocessing tools for oligonucleotide arrays*. R package version 1.42.0. 2017.
- [32] Min Chen y col. «A powerful Bayesian meta-analysis method to integrate multiple gene set enrichment studies». En: *Bioinformatics* 29.7 (2013), págs. 862-869. doi: [10.1093/bioinformatics/btt068](https://doi.org/10.1093/bioinformatics/btt068). eprint: <http://bioinformatics.oxfordjournals.org/content/29/7/862.full.pdf+html>. url: <http://bioinformatics.oxfordjournals.org/content/29/7/862.abstract>.
- [33] Quan Chen, Xianghong J. Zhou y Fengzhu Sun. «Finding Genetic Overlaps Among Diseases Based on Ranked Gene Lists». En: *Journal of Computational Biology* 22.2 (feb. de 2015), págs. 111-123. issn: 1066-5277. doi: [10.1089/cmb.2014.0149](https://doi.org/10.1089/cmb.2014.0149). url: <http://dx.doi.org/10.1089/cmb.2014.0149>.
- [34] Yunshun Chen y col. *edgeR: Empirical Analysis of Digital Gene Expression Data in R*. R package version 3.20.2. 2017. url: <http://bioinf.wehi.edu.au/edgeR>.
- [35] Peter J A Cock y col. «The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants». En: *Nucleic Acids Research* 38.6 (nov. de 2009), págs. 1767-1771. issn: 1362-4962. url: <http://www.ncbi.nlm.nih.gov/pmc/articles/PMC2847217/>.
- [36] Ana Conesa y col. «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](https://doi.org/10.1186/s13059-016-0881-8). url: <http://dx.doi.org/10.1186/s13059-016-0881-8>.

- [37] Aedín C. Culhane y col. «GeneSigDB: a manually curated database and resource for analysis of gene expression signatures». En: *Nucleic Acids Research* 40.D1 (2012), págs. D1060-D1066. doi: [10.1093/nar/gkr901](https://doi.org/10.1093/nar/gkr901). eprint: <http://nar.oxfordjournals.org/content/40/D1/D1060.full.pdf+html>. url: <http://nar.oxfordjournals.org/content/40/D1/D1060.abstract>.
- [38] Sean Davis. *GEOquery: Get data from NCBI Gene Expression Omnibus (GEO)*. R package version 2.46.13. 2017. url: <https://github.com/seandavi/GEOquery>.
- [39] Nicolas Delhomme e Ismael Padoleau. *RnaSeqTutorial: RNA-Seq Tutorial (EBI Cambridge UK, October 2011)*. R package version 0.16.0. 2015.
- [40] S. Dudoit, J.P. Shaffer y J.C. Boldrick. «Multiple hypothesis testing in microarray experiments». En: *Statistical Science* 18 (2003). Microarrays, págs. 71-103.
- [41] Sandrine Dudoit y Robert Gentleman. *Cluster Analysis in DNA Microarray Experiments*. Bioconductor Short Course Winter 2002. Bioconductor.org. 2002. url: <http://www.bioconductor.org/help/course-materials/2002/Seattle02/Cluster/cluster.pdf>.
- [42] Steffen Durinck y Wolfgang Huber. *biomaRt: Interface to BioMart databases (e.g. Ensembl, COSMIC, Wormbase and Gramene)*. R package version 2.34.1. 2017.
- [43] Brad Efron y R. Tibshirani. *GSA: Gene set analysis*. R package version 1.03. 2010. url: <https://CRAN.R-project.org/package=GSA>.
- [44] 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](https://doi.org/10.1214/07-AOAS101).
- [45] Brent Ewing y Phil Green. «Base-Calling of Automated Sequencer Traces Using Phred. II. Error Probabilities». En: *Genome Research* 8.3 (1998), págs. 186-194. doi: [10.1101/gr.8.3.186](https://doi.org/10.1101/gr.8.3.186). eprint: <http://genome.cshlp.org/content/8/3/186.full.pdf+html>. url: <http://genome.cshlp.org/content/8/3/186.abstract>.
- [46] Brent Ewing y col. «Base-Calling of Automated Sequencer Traces UsingPhred. I. Accuracy Assessment». En: *Genome Research* 8.3 (1998), págs. 175-185. doi: [10.1101/gr.8.3.175](https://doi.org/10.1101/gr.8.3.175). eprint: <http://genome.cshlp.org/content/8/3/175.full.pdf+html>. url: <http://genome.cshlp.org/content/8/3/175.abstract>.
- [47] 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](https://doi.org/10.1093/bioinformatics/btl567). eprint: <http://bioinformatics.oxfordjournals.org/content/23/2/257.full.pdf+html>. url: <http://bioinformatics.oxfordjournals.org/content/23/2/257.abstract>.
- [48] Julian Faraway. *faraway: Functions and Datasets for Books by Julian Faraway*. R package version 1.0.7. 2016. url: <https://CRAN.R-project.org/package=faraway>.

- [49] Richard Fox y Matthew Dimmic. «A two-sample Bayesian t-test for microarray data». En: *BMC Bioinformatics* 7.1 (2006), pág. 126. issn: 1471-2105. doi: [10.1186/1471-2105-7-126](https://doi.org/10.1186/1471-2105-7-126). url: <http://www.biomedcentral.com/1471-2105/7/126>.
- [50] Kyle Furge y Karl Dykema. *PGSEA: Parametric Gene Set Enrichment Analysis*. R package version 1.52.0. 2012.
- [51] Laurent Gautier y col. «affy—analysis of Affymetrix Gene-Chip data at the probe level». En: *Bioinformatics* 20.3 (2004), págs. 307-315. issn: 1367-4803. doi: [http://dx.doi.org/10.1093/bioinformatics/btg405](https://doi.org/10.1093/bioinformatics/btg405).
- [52] Ludwig Geistlinger, Gergely Csaba y Ralf Zimmer. *EnrichmentBrowser: Seamless navigation through combined results of set-based and network-based enrichment analysis*. R package version 2.4.6. 2017.
- [53] Ludwig Geistlinger y col. «From sets to graphs: towards a realistic enrichment analysis of transcriptomic systems». En: *Bioinformatics* 27.13 (2011), págs. i366-i373. doi: [10.1093/bioinformatics/btr228](https://doi.org/10.1093/bioinformatics/btr228). eprint: <http://bioinformatics.oxfordjournals.org/content/27/13/i366.full.pdf+html>. url: <http://bioinformatics.oxfordjournals.org/content/27/13/i366.abstract>.
- [54] R. Gentleman. *annotate: Annotation for microarrays*. R package version 1.56.1. 2017.
- [55] Robert Gentleman. *Category: Category Analysis*. R package version 2.44.0. 2017.
- [56] Robert Gentleman. *GOstats: Tools for manipulating GO and microarrays*. R package version 2.44.0. 2017.
- [57] 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](https://doi.org/10.2202/1544-6115.1034).
- [58] Robert Gentleman y Duncan Temple Lang. *Statistical Analyses and Reproducible Research*. Inf. téc. Bioconductor Project. Bioconductor Project Working Papers, 2004.
- [59] R. Gentleman y col. *Biobase: Base functions for Bioconductor*. R package version 2.38.0. 2017.
- [60] R. Gentleman y col., eds. *Bioinformatics and Computational Biology Solutions Using R and Bioconductor*. Springer, 2005.
- [61] R. Gentleman y col. *genefilter: methods for filtering genes from high-throughput experiments*. R package version 1.60.0. 2017.
- [62] T. R. Golub y col. «Molecular classification of cancer: class discovery and class prediction by gene expression monitoring.» eng. En: *Science* 286.5439 (1999), págs. 531-537. url: [10.1126/science.286.5439.53](https://doi.org/10.1126/science.286.5439.53).
- [63] Todd Golub. *golubEsets: exprSets for golub leukemia data*. R package version 1.20.0. 2017.
- [64] Cedric Gondro. *Primer to Analysis of Genomic Data Using R*. Springer International Publishing, 2015. doi: [10.1007/978-3-319-14475-7](https://doi.org/10.1007/978-3-319-14475-7).

- [65] Juergen Gross y Uwe Ligges. *nortest: Tests for Normality*. R package version 1.0-4. 2015. url: <https://CRAN.R-project.org/package=nortest>.
- [66] Manuel Ugidos Guerrero. «Estudio del perfil de expresión de micro-RNA en pacientes de cáncer de mama». Tesis de mtria. Universidad de Valencia, 2017.
- [67] Justin Guinney y Robert Castelo. *GSVA: Gene Set Variation Analysis for microarray and RNA-seq data*. R package version 1.26.0. 2017. url: <https://github.com/rcastelo/GSVA>.
- [68] Jason A. Hackney y col. *ReportingTools: Tools for making reports in various formats*. R package version 2.17.3. 2017.
- [69] Florian Hahne y col. *Bioconductor Case Studies*. Use R! Springer, 2008.
- [70] Sonja Hanzelmann, Robert Castelo y Justin Guinney. «GSVA: gene set variation analysis for microarray and RNA-Seq data». En: *BMC Bioinformatics* 14.1 (2013), pág. 7. issn: 1471-2105. doi: <10.1186/1471-2105-14-7>. url: <http://www.biomedcentral.com/1471-2105/14/7>.
- [71] Frank E Harrell Jr. *Hmisc: Harrell Miscellaneous*. R package version 4.1-0. 2017. url: <https://CRAN.R-project.org/package=Hmisc>.
- [72] Masaaki Horikoshi y Yuan Tang. *ggfortify: Data Visualization Tools for Statistical Analysis Results*. R package version 0.4.1. 2017. url: <https://CRAN.R-project.org/package=ggfortify>.
- [73] Wolfgang Huber y col. «Orchestrating high-throughput genomic analysis with Bioconductor». En: *Nat Meth* 12.2 (feb. de 2015), págs. 115-121. issn: 1548-7091. url: <http://dx.doi.org/10.1038/nmeth.3252>.
- [74] Affymetrix Inc., Crispin J Miller y PICR. *plier: Implements the Affymetrix PLIER algorithm*. R package version 1.48.0. 2017.
- [75] Rafael A. Irizarry y col. *affy: Methods for Affymetrix Oligonucleotide Arrays*. R package version 1.56.0. 2017.
- [76] R.A. Irizarry y col. «Exploration, normalization, and summaries of high density oligonucleotide array probe level data». En: *Biostatistics* 4.2 (2003), págs. 249-264. doi: <10.1093/biostatistics/4.2.249>. eprint: <http://biostatistics.oxfordjournals.org/content/4/2/249.full.pdf+html>. url: <http://biostatistics.oxfordjournals.org/content/4/2/249.abstract>.
- [77] R.A. Irizarry y col. «Gene set enrichment analysis made simple». En: *Statistical Methods in Medical Research* 18.6 (dic. de 2009). Gene set analysis, págs. 565-575.
- [78] Laurent Jacob, Pierre Neuvial y Sandrine Dudoit. *DEGraph: Two-sample tests on a graph*. R package version 1.30.0. 2012.
- [79] Laurent Jacob, Pierre Neuvial y Sandrine Dudoit. «More power via graph-structured tests for differential expression of gene networks». En: *Ann. Appl. Stat.* 6.2 (jun. de 2012), págs. 561-600. doi: <10.1214/11-AOAS528>. url: <http://dx.doi.org/10.1214/11-AOAS528>.

- [80] Arnoud J. Kal y col. «Dynamics of Gene Expression Revealed by Comparison of Serial Analysis of Gene Expression Transcript Profiles from Yeast Grown on Two Different Carbon Sources». En: *Molecular Biology of the Cell* 10.6 (1999), págs. 1859-1872. doi: [10.1091/mbc.10.6.1859](https://doi.org/10.1091/mbc.10.6.1859). eprint: <http://www.molbiolcell.org/content/10/6/1859.full.pdf+html>. url: <http://www.molbiolcell.org/content/10/6/1859.abstract>.
- [81] Audrey Kauffmann, Ibrahim Emam y Michael Schubert. *ArrayExpress: Access the ArrayExpress Microarray Database at EBI and build Bioconductor data structures: ExpressionSet, AffyBatch, NChannelSet*. R package version 1.38.0. 2017.
- [82] L. Kaufman y P.J. Rousseeuw. *Finding Groups in Data. An Introduction to Cluster Analysis*. Wiley, 1990.
- [83] Eija Korpelainen y col. *RNA-seq Data Analysis A Practical Approach*. CRC Press, 2015.
- [84] Marcin Kosinski y Przemyslaw Biecek. *RTCGA: The Cancer Genome Atlas Data Integration*. R package version 1.8.0. 2016. url: <https://github.com/RTCGA/RTCGA>.
- [85] Eric Lecoutre. *R2HTML: HTML Exportation for R Objects*. R package version 2.3.2. 2016. url: <https://CRAN.R-project.org/package=R2HTML>.
- [86] Jun Li y Robert Tibshirani. «Finding consistent patterns: A nonparametric approach for identifying differential expression in RNA-Seq data». En: *Statistical Methods in Medical Research* 22.5 (2013), págs. 519-536. doi: [10.1177/0962280211428386](https://doi.org/10.1177/0962280211428386). eprint: <http://smm.sagepub.com/content/22/5/519.full.pdf+html>. url: <http://smm.sagepub.com/content/22/5/519.abstract>.
- [87] Xiaochun Li. *ALL: A data package*. R package version 1.20.0. 2009.
- [88] Q. Liu y col. «Comparative evaluation of gene-set analysis methods». En: *BMC Bioinformatics* 8.1 (2007), págs. 1-15. issn: 1471-2105. doi: [10.1186/1471-2105-8-431](https://doi.org/10.1186/1471-2105-8-431). url: <http://dx.doi.org/10.1186/1471-2105-8-431>.
- [89] Michael Love. *airway: RangedSummarizedExperiment for RNA-Seq in airway smooth muscle cells, by Himes et al PLoS One 2014*. R package version 0.112.0. 2017.
- [90] Michael Love. *parathyroidSE: RangedSummarizedExperiment for RNA-Seq of primary cultures of parathyroid tumors by Haglund et al., J Clin Endocrinol Metab 2012*. R package version 1.16.0. 2017.
- [91] Michael Love, Wolfgang Huber y Simon Anders. «Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2». En: *Genome Biology* 15.12 (2014), pág. 550. issn: 1465-6906. doi: [10.1186/s13059-014-0550-8](https://doi.org/10.1186/s13059-014-0550-8). url: <http://genomebiology.com/2014/15/12/550>.
- [92] P. Lund Steven y col. *Detecting Differential Expression in RNA-sequence Data Using Quasi-likelihood with Shrunken Dispersion Estimates*. 2012. url: <http://www.degruyter.com/view/j/sagmb.2012.11.issue-5/1544-6115.1826/1544-6115.1826.xml>.

- [93] Henryk Maciejewski. «Gene set analysis methods: statistical models and methodological differences». En: *Briefings in Bioinformatics* (2013). doi: [10.1093/bib/bbt002](https://doi.org/10.1093/bib/bbt002). eprint: <http://bib.oxfordjournals.org/content/early/2013/02/09/bib.bbt002.full.pdf+html>. url: <http://bib.oxfordjournals.org/content/early/2013/02/09/bib.bbt002.abstract>.
- [94] S. C. Madeira y A. L. Oliveira. «Bioclustering algorithms for biological data analysis: a survey». En: *IEEE/ACM Transactions on Computational Biology and Bioinformatics* 1.1 (2004), págs. 24-45. doi: [10.1109/TCBB.2004.2](https://doi.org/10.1109/TCBB.2004.2).
- [95] Martin Maechler y col. *cluster: "Finding Groups in Data": Cluster Analysis Extended Rousseeuw et al.* R package version 2.0.6. 2017. url: <https://CRAN.R-project.org/package=cluster>.
- [96] Paolo Martini y col. «Along signal paths: an empirical gene set approach exploiting pathway topology». En: *Nucleic Acids Research* 41.1 (2013), e19. doi: [10.1093/nar/gks866](https://doi.org/10.1093/nar/gks866). eprint: <http://nar.oxfordjournals.org/content/41/1/e19.full.pdf+html>. url: <http://nar.oxfordjournals.org/content/41/1/e19.abstract>.
- [97] V.K . Mootha y col. «PGC-1alpha-responsive genes involved in oxidative phosphorylation are coordinately downregulated in human diabetes». En: *Nature Genetics* 34 (2003), págs. 267-73.
- [98] Martin Morgan, Seth Falcon y Robert Gentleman. *GSEABase: Gene set enrichment data structures and methods*. R package version 1.40.1. 2017.
- [99] Martin Morgan, Michael Lawrence y Simon Anders. *ShortRead: FASTQ input and manipulation*. R package version 1.36.0. 2017.
- [100] Martin Morgan y col. *Rsamtools: Binary alignment (BAM), FASTA, variant call (BCF), and tabix file import*. R package version 1.30.0. 2017. url: <http://bioconductor.org/packages/release/bioc/html/Rsamtools.html>.
- [101] Martin Morgan y col. *SummarizedExperiment: SummarizedExperiment container*. R package version 1.8.1. 2017.
- [102] Ali Mortazavi y col. «Mapping and quantifying mammalian transcriptomes by RNA-Seq». En: *Nat Meth* 5.7 (jul. de 2008), págs. 621-628. issn: 1548-7091. url: [http://dx.doi.org/10.1038/nmeth.1226](https://doi.org/10.1038/nmeth.1226).
- [103] Haroon Naeem y col. «Rigorous assessment of gene set enrichment tests». En: *Bioinformatics* 28.11 (2012), págs. 1480-1486. doi: [10.1093/bioinformatics/bts164](https://doi.org/10.1093/bioinformatics/bts164). eprint: <http://bioinformatics.oxfordjournals.org/content/28/11/1480.full.pdf+html>. url: <http://bioinformatics.oxfordjournals.org/content/28/11/1480.abstract>.
- [104] Richard M. Neve y col. «A collection of breast cancer cell lines for the study of functionally distinct cancer subtypes». En: *Cancer cell* 10.6 (2006), págs. 515-527. doi: [10.1016/j.ccr.2006.10.008](https://doi.org/10.1016/j.ccr.2006.10.008).

- [105] Michael A. Newton y col. «Random-set methods identify distinct aspects of the enrichment signal in gene-set analysis». En: *Annals of Applied Statistics* 1.1 (2007), págs. 85-106. doi: [10.1214/07-AOAS104](https://doi.org/10.1214/07-AOAS104). url: <http://projecteuclid.org/euclid.aoas/1183143730>.
- [106] Tobias Oetiker y col. *La introducción no-tan-corta a LATEX* 2e. Feb. de 2010.
- [107] 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](https://doi.org/10.1186/gb-2010-11-12-220). url: <http://genomebiology.com/2010/11/12/220>.
- [108] Hervé Pagès y col. *AnnotationDbi: Annotation Database Interface*. R package version 1.40.0. 2017.
- [109] Yudi Pawitan y Alexander Ploner. *OCplus: Operating characteristics plus sample size and local fdr for microarray experiments*. R package version 1.52.0. 2017.
- [110] J. K. Pickrell y col. «Understanding mechanisms underlying human gene expression variation with RNA sequencing». En: *Nature* 464 (2010). doi: [10.1038/nature08872](https://doi.org/10.1038/nature08872). url: <http://dx.doi.org/10.1038/nature08872>.
- [111] Katherine S. Pollard y col. *multtest: Resampling-based multiple hypothesis testing*. R package version 2.34.0. 2017.
- [112] Weiliang Qiu, Mei-Ling Ting Lee y George Alex Whitmore. *sizepower: Sample Size and Power Calculation in Microarray Studies*. R package version 1.48.0. 2017.
- [113] R Core Team. *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria, 2017. url: <https://www.R-project.org/>.
- [114] Yasir Rahmatallah, Frank Emmert-Streib y Galina Glazko. «Comparative evaluation of gene set analysis approaches for RNA-Seq data». En: *BMC Bioinformatics* 15.1 (2014), págs. 1-15. issn: 1471-2105. doi: [10.1186/s12859-014-0397-8](https://doi.org/10.1186/s12859-014-0397-8). url: <http://dx.doi.org/10.1186/s12859-014-0397-8>.
- [115] C.R. Rao. *Linear Statistical Inference and Its Applications*. Wiley, 1967.
- [116] 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](https://doi.org/10.1093/bioinformatics/btf877). eprint: <http://bioinformatics.oxfordjournals.org/content/19/3/368.full.pdf+html>. url: <http://bioinformatics.oxfordjournals.org/content/19/3/368.abstract>.
- [117] Tyler Rinker y Dason Kurkiewicz. *pacman: Package Management Tool*. R package version 0.4.6. 2017. url: <https://CRAN.R-project.org/package=pacman>.
- [118] David Robinson. *broom: Convert Statistical Analysis Objects into Tidy Data Frames*. R package version 0.4.3. 2017. url: <https://CRAN.R-project.org/package=broom>.

- [119] M. D. Robinson y A. Oshlack. «A scaling normalization method for differential expression analysis of RNA-seq data». En: *Genome Biol* 11 (2010). doi: [10.1186/gb-2010-11-3-r25](https://doi.org/10.1186/gb-2010-11-3-r25). url: <http://dx.doi.org/10.1186/gb-2010-11-3-r25>.
- [120] 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](https://doi.org/10.1093/bioinformatics/btm453). eprint: <http://bioinformatics.oxfordjournals.org/content/23/21/2881.full.pdf+html>. url: <http://bioinformatics.oxfordjournals.org/content/23/21/2881.abstract>.
- [121] 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](https://doi.org/10.1093/biostatistics/kxm030). eprint: <http://biostatistics.oxfordjournals.org/content/9/2/321.full.pdf+html>. url: <http://biostatistics.oxfordjournals.org/content/9/2/321.abstract>.
- [122] Gabriele Sales y col. «Graphite Web: web tool for gene set analysis exploiting pathway topology». En: *Nucleic Acids Research* (2013). doi: [10.1093/nar/gkt386](https://doi.org/10.1093/nar/gkt386). eprint: <http://nar.oxfordjournals.org/content/early/2013/05/10/nar.gkt386.full.pdf+html>. url: <http://nar.oxfordjournals.org/content/early/2013/05/10/nar.gkt386.abstract>.
- [123] Fatemeh Seyednasrollah, Asta Laiho y Laura L. Elo. «Comparison of software packages for detecting differential expression in RNA-seq studies». En: *Briefings in Bioinformatics* (2013). doi: [10.1093/bib/bbt086](https://doi.org/10.1093/bib/bbt086). eprint: <http://bib.oxfordjournals.org/content/early/2013/12/02/bib.bbt086.full.pdf+html>. url: <http://bib.oxfordjournals.org/content/early/2013/12/02/bib.bbt086.abstract>.
- [124] Mirko Signorelli, Veronica Vinciotti y Ernst C. Wit. «NEAT: an efficient network enrichment analysis test». En: *BMC Bioinformatics* 17.1 (2016), págs. 1-17. issn: 1471-2105. doi: [10.1186/s12859-016-1203-6](https://doi.org/10.1186/s12859-016-1203-6). url: <http://dx.doi.org/10.1186/s12859-016-1203-6>.
- [125] Mirko Signorelli, Veronica Vinciotti y Ernst C. Wit. *neat: Efficient Network Enrichment Analysis Test*. R package version 1.1.1. 2017. url: <https://CRAN.R-project.org/package=neat>.
- [126] P.P. Sinha. *Bioinformatics with R Cookbook*. Packt Publishing, 2014.
- [127] Martin Slawski y Anne-Laure Boulesteix. *GeneSelector: Stability and Aggregation of ranked gene lists*. R package version 2.28.0. 2017.
- [128] 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.
- [129] Gordon Smyth y col. *limma: Linear Models for Microarray Data*. R package version 3.34.4. 2017. url: <http://bioinf.wehi.edu.au/limma>.

- [130] Peter H.A. Sneath y Rober R. Sokal. *Numerical Taxonomy. The principles and practice of numerical Classification.* San Francisco, USA: W.H. Freeman y Company, 1973.
- [131] Phillip Stafford, ed. *Methods in Microarray Normalization.* CRC Press, 2008.
- [132] John D. Storey y Robert Tibshirani. «Statistical significance for genomewide studies». En: *Proceedings of the National Academy of Sciences* 100.16 (2003), págs. 9440-9445. doi: [10.1073/pnas.1530509100](https://doi.org/10.1073/pnas.1530509100). eprint: <http://www.pnas.org/content/100/16/9440.full.pdf+html>. url: <http://www.pnas.org/content/100/16/9440.abstract>.
- [133] Aravind Subramanian y col. «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](https://doi.org/10.1073/pnas.0506580102). eprint: <http://www.pnas.org/content/102/43/15545.full.pdf+html>. url: <http://www.pnas.org/content/102/43/15545.abstract>.
- [134] Dan Tenenbaum. *KEGGREST: Client-side REST access to KEGG.* R package version 1.14.1. 2017.
- [135] Terry M Therneau. *RNASeqPower: Sample size for RNAseq studies.* R package version 1.18.0. 2017.
- [136] Lu Tian y col. «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](https://doi.org/10.1073/pnas.0506577102). eprint: <http://www.pnas.org/content/102/38/13544.full.pdf+html>. url: <http://www.pnas.org/content/102/38/13544.abstract>.
- [137] R. Tibshirani y col. *samr: SAM: Significance Analysis of Microarrays.* R package version 2.0. 2011. url: <https://CRAN.R-project.org/package=samr>.
- [138] Shailesh Tripathi, Galina V. Glazko y Frank Emmert-Streib. «Ensuring the statistical soundness of competitive gene set approaches: gene filtering and genome-scale coverage are essential». En: *Nucleic Acids Research* 41.7 (2013), e82. doi: [10.1093/nar/gkt054](https://doi.org/10.1093/nar/gkt054). eprint: <http://nar.oxfordjournals.org/content/41/7/e82.full.pdf+html>. url: <http://nar.oxfordjournals.org/content/41/7/e82.abstract>.
- [139] Olga Troyanskaya y col. «Missing value estimation methods for DNA microarrays». En: *Bioinformatics* 17.6 (2001), págs. 520-525. doi: [10.1093/bioinformatics/17.6.520](https://doi.org/10.1093/bioinformatics/17.6.520). eprint: <http://bioinformatics.oxfordjournals.org/content/17/6/520.full.pdf+html>. url: <http://bioinformatics.oxfordjournals.org/content/17/6/520.abstract>.
- [140] 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](https://doi.org/10.1073/pnas.091062498). eprint: <http://www.pnas.org/content/98/9/5116.full.pdf+html>. url: <http://www.pnas.org/content/98/9/5116.abstract>.

- [org/content/98/9/5116.full.pdf+html](http://www.pnas.org/content/98/9/5116.full.pdf+html). url: <http://www.pnas.org/content/98/9/5116.abstract>.
- [141] Maarten van Iterson. *SSPA: General Sample Size and Power Analysis for Microarray and Next-Generation Sequencing Data*. R package version 2.18.0. 2017. url: <http://www.humgen.nl/MicroarrayAnalysisGroup.html>.
- [142] Leif Varemo e Intawat Nookaew. *piano: Platform for integrative analysis of omics data*. R package version 1.18.0. 2017. url: <http://www.sysbio.se/piano>.
- [143] Marie Verbanck, Sébastien Le y Jerome Pages. «A new unsupervised gene clustering algorithm based on the integration of biological knowledge into expression data». En: *BMC Bioinformatics* 14.1 (2013), pág. 42. issn: 1471-2105. doi: [10.1186/1471-2105-14-42](https://doi.org/10.1186/1471-2105-14-42). url: <http://www.biomedcentral.com/1471-2105/14/42>.
- [144] John Verzani. *UsingR: Data Sets, Etc. for the Text "Using R for Introductory Statistics", Second Edition*. R package version 2.0-5. 2015. url: <https://CRAN.R-project.org/package=UsingR>.
- [145] Zhong Wang, Mark Gerstein y Michael Snyder. «RNA-Seq: a revolutionary tool for transcriptomics». En: *Nat Rev Genet* 10.1 (ene. de 2009), págs. 57-63. issn: 1471-0056. url: <http://dx.doi.org/10.1038/nrg2484>.
- [146] Gregory R. Warnes, Peng Liu y Fasheng Li. *ssize: Estimate Microarray Sample Size*. R package version 1.52.0. 2012.
- [147] Hadley Wickham. *reshape: Flexibly Reshape Data*. R package version 0.8.7. 2017. url: <https://CRAN.R-project.org/package=reshape>.
- [148] 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](https://doi.org/10.18637/jss.v021.i12). url: <https://www.jstatsoft.org/index.php/jss/article/view/v021i12>.
- [149] Hadley Wickham. «Tidy Data». En: *Journal of Statistical Software* 59.1 (2014), págs. 1-23. issn: 1548-7660. doi: [10.18637/jss.v059.i10](https://doi.org/10.18637/jss.v059.i10). url: <https://www.jstatsoft.org/index.php/jss/article/view/v059i10>.
- [150] Hadley Wickham y Winston Chang. *ggplot2: Create Elegant Data Visualisations Using the Grammar of Graphics*. R package version 2.2.1. 2016. url: <https://CRAN.R-project.org/package=ggplot2>.
- [151] Rand R. Wilcox. *Basic Statistics*. Oxford University Press, 2009. url: [/home/gag/BIBLIOGRAFIA/MISLIBROS/Wilcox\\_Basic\\_Statistics\\_Oxford\\_University\\_Press\\_2009.pdf](/home/gag/BIBLIOGRAFIA/MISLIBROS/Wilcox_Basic_Statistics_Oxford_University_Press_2009.pdf).
- [152] E. Wit y J.D. McClure. *Statistics for microarrays: design, analysis, and inference*. Wiley, 2004.

- [153] Daniela M Witten y Robert Tibshirani. «Scientific research in the age of omics: the good, the bad, and the sloppy». En: *Journal of the American Medical Informatics Association* 20.1 (2013), págs. 125-127. doi: [10.1136/ajmijnl-2012-000972](https://doi.org/10.1136/ajmijnl-2012-000972). eprint: <http://jamia.bmj.com/content/20/1/125.full.pdf+html>. url: <http://jamia.bmj.com/content/20/1/125.abstract>.
- [154] 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](https://doi.org/10.1093/nar/gks461). eprint: <http://nar.oxfordjournals.org/content/40/17/e133.full.pdf+html>. url: <http://nar.oxfordjournals.org/content/40/17/e133.abstract>.
- [155] Hao Wu. *PROPER: PROspective Power Evaluation for RNA-seq*. R package version 1.10.0. 2016.
- [156] Jean Wu y Rafael Irizarry with contributions from James MacDonald Jeff Gentry. *germa: Background Adjustment Using Sequence Information*. R package version 2.50.0. 2017.
- [157] Jianguo Xia, Erin E Gill y Robert E W Hancock. «NetworkAnalyst for statistical, visual and network-based meta-analysis of gene expression data». En: *Nat. Protocols* 10.6 (jun. de 2015), págs. 823-844. issn: 1754-2189. url: [http://dx.doi.org/10.1038/nprot.2015.052](https://doi.org/10.1038/nprot.2015.052).
- [158] Yihui Xie. *Dynamic Documents with R and knitr*. 2nd. Chapman & Hall/CRC The R Series. Chapman y Hall/CRC, 2015. isbn: 1498716962, 9781498716963, 9781498716970.
- [159] Yihui Xie. *knitr: A General-Purpose Package for Dynamic Report Generation in R*. R package version 1.17. 2017. url: <https://CRAN.R-project.org/package=knitr>.
- [160] Qing Xiong y col. «Integrating genetic and gene expression evidence into genome-wide association analysis of gene sets». En: *Genome Research* 22.2 (2012), págs. 386-397. doi: [10.1101/gr.124370.111](https://doi.org/10.1101/gr.124370.111). eprint: <http://genome.cshlp.org/content/22/2/386.full.pdf+html>. url: <http://genome.cshlp.org/content/22/2/386.abstract>.
- [161] Matthew Young. *geneLenDataBase: Lengths of mRNA transcripts for a number of genomes*. R package version 1.14.0. 2017.
- [162] Shilin Zhao y col. *RnaSeqSampleSizeData: RnaSeqSampleSizeData*. R package version 1.10.0. 2014.
- [163] Yi-Hui Zhou, William T. Barry y Fred A. Wright. «Empirical pathway analysis, without permutation». En: *Biostatistics* (2013). doi: [10.1093/biostatistics/kxt004](https://doi.org/10.1093/biostatistics/kxt004). eprint: <http://biostatistics.oxfordjournals.org/content/early/2013/02/20/biostatistics.kxt004.full.pdf+html>. url: <http://biostatistics.oxfordjournals.org/content/early/2013/02/20/biostatistics.kxt004.abstract>.
- [164] Chandler Zuo y Sunduz Keles. *CSSP: ChIP-Seq Statistical Power*. R package version 1.16.0. 2015.

# Índice alfabético

- affy, 91
- Affymetrix
  - Fichero CDF, 56
  - Fichero CEL, 56
  - Fichero DAT, 56
  - hgu133a.db, 255
- Affymetrix GeneChip, 55
- ALL, 79, 259
- annotate, 196
  - annotation, 196
  - getSYMBOL, 197
  - lookUp, 197
- Análisis cluster, 5
- Análisis de la varianza, 139
- ArrayExpress, 95
- base
  - apply, 129
- Biobase
  - ExpressionSet, 80
  - sample.ExpressionSet, 75
- Bioconductor, 26
  - ALL, 26
  - biocLite, 26
  - multtest
    - golub, 14
- Bootstrap, 294
- Category, 255
- coldata, 112
- Comparaciones múltiples, 143
- Control débil del error, 146
- Control fuerte del error, 146
- Datos
  - GEO
    - GSE21779, 90
    - gse1397, 290
    - GSE37211, 114
    - GSE64099, 113
- Distribución binomial
  - Aproximación Poisson, 339
- Distribución binomial negativa, 340
- Distribución empírica, 330
- Distribución geométrica, 340
- Distribución Poisson, 338
- Distribución de los sucesos raros, 339
- edgeR
  - binomTest, 174
- Ensembl, 197
- Error cuadrático medio, 451
- Error estándar, 451
- Espacio paramétrico, 450
- Estimador, 451
- Estimador máximo verosímil, 453
- FDR
  - False discovery rate, 145
  - Tasa de falsamente rechazados, 145
- Filtrado independiente, 128
- Fold-change, 133
- Fox-Dimmic, 290
- Función de distribución empírica, 330
- FWER
  - Familywise error rate, 145
  - tasa de error global, 145
- Gene Ontology: GO, 197
- Gene set analysis, 5
- Gene set enrichment analysis, 5
- genefilter, 131
  - kOverA, 131
  - nsFilter, 133
  - rowFtests, 141
  - rowttests, 137
- GeneSelector, 289, 290
  - AggregateSimple, 297
  - GenerateBootMatrix, 295
  - RankingFC, 290
  - RankingLimma, 290
  - RankingTstat, 138, 290
  - RepeatRanking, 295
  - toplist, 292
- GenomicRanges
  - assay, 112
  - coldata, 112

- rowRanges, 113
- GEOquery, 94
  - getGEOSuppFiles, 94
- ggplot2, 87
  - qplot
    - facets, 17
- GO.db, 256
- golubEsets
  - golub, 79
- GOSTats, 255
  - hyperGTest, 256
- GSE1397, 91
- GSE20986, 94, 195, 196, 244, 257
  - gse20986, 258
- GSE34764, 95
- GSE37211, 111
- GSE64099, 113
- GSEABase
  - GeneSetCollection, 247
- GSEAbase, 244
  - details, 245
  - geneIds, 244
  - GeneSet, 244
  - GeneSetCollection, 245
- hgu133plus2.db, 196, 258
- IDE
  - RStudio, 8
- Intervalo de confianza, 457
- IRanges::DataFrame, 112
- Jackknife, 294
- leave-one-out, 294
- Limma, 290
- limma
  - normalizeBetweenArrays, 74
- Lista ordenada, 289
- Listas ordenadas
  - estabilidad, 290
- Logverosimilitu, 449
- MA plot, 61
- MAS5, 65
- Matriz de correlaciones muestral, 453
- Matriz de covarianzas muestral, 453
- max T por pasos de bajada, 150
- Median polish, 70
- Meta-analisis
  - Método de Fisher, 471
  - Método de Stouffer, 471
- Microarray, 55
- multtest
- golub, 77
- p.adjust, 151
- Método de Benjamini y Hochberg, 150
- Método de Bonferroni, 148
- Método de Hochberg, 150
- Método de Holm, 149
- Método de Sídák, 149
- NCBI Gene Expression Omnibus (GEO), 90
- Nivel de confianza, 457
- Normalización de cuantiles, 68
- nsFilter, 258
- org.Sc.sgd.db, 246
- p-valor ajustado, 147
- Paquete R
  - parathyroidSE, 111
- PGSEA, 281
- Prueba Bernoulli, 334
- q-valor, 152
- qvalue, 154
- R
  - as.matrix, 81
  - base
    - apply, 19
    - class, 16
    - colnames, 19
    - dim, 16
    - factor, 15
    - help, 8
    - help.start, 8
    - mean, 19
    - ncol, 16
    - nrow, 16
    - rownames, 19
    - sort, 11
  - class, 81
  - data.frame, 81
  - function, 23
  - Instalación, 7
  - Listas, 22
  - Logical Operators, 11
  - matplot, 58
  - Paquetes
    - ggplot2, 15
    - UsingR, 8
  - paste0, 23
  - read.csv, 81
  - read.table, 81
  - return, 24

sapply, 84  
 save, 91  
 setwd, 90  
 Short-refcard, 8  
 which, 11  
 R2HTML, 197  
     HTML, 198  
 Región crítica, 455  
 ReportingTools, 198  
     finish, 199  
     HTMLReport, 199  
     publish, 199  
 reshape  
     melt, 58, 87  
 Robust multichip average (RMA),  
     67  
 rowRanges, 113  
 RStudio, 8  
  
 SAM, 157  
     samr, 160  
 Selección no específica, 128  
 Sesgo, 451  
 SummarizedExperiment  
     RangedSummarizedExperiment,  
         112  
  
 tami13  
     AllTami.R, 26  
 TCGA: The Cancer Genome Atlas,  
     477  
 Test de Fisher unilateral, 251  
     fisher.test, 253  
 Test de Wald, 456  
 Test del cociente de verosimilitu-  
     des, 456  
 Top-k list, 290  
 topGO, 281  
  
 Variable aleatoria  
     Bernoulli, 334  
     discreta  
         Poisson, 338  
 Variable Bernoulli, 334  
 Verosimilitud, 449

# Glosario

**Affymetrix** <http://www.affymetrix.com/>. 31

**BAM** Es la versión binaria del formato **SAM**. <http://genome.sph.umich.edu/wiki/SAM> y la especificación del formato la tenemos en <http://samtools.sourceforge.net/SAM1.pdf>. 107

**Bioconductor** Red de espejos con paquetes de R para análisis de datos ómicos: <https://www.bioconductor.org/>. 26, 494

**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). . 48

**Bowtie2** Alineador de secuencias. <http://bowtie-bio.sourceforge.net/bowtie2/index.shtml> . 109, 110, 121

**CDS** Es la región codificante de un gen: Coding DNA Sequence. Es la parte del gen (DNA o RNA) compuesta por los exones que codifican proteína. El CDS es la porción de un transcríto que es trasladado por un ribosoma. . 39, 42

**CRAN** Red de espejos con paquetes de R: <https://cran.r-project.org/> . 26, 494

**Ensembl** <https://en.wikipedia.org/wiki/Ensembl>. 33, 34, 49, 51, 52

**Entrez** <https://en.wikipedia.org/wiki/Entrez>. 31, 33, 49, 51, 52

**ExpressionSet** Clase S4 para almacenar datos de expresión de micro-arrays. Definida en [59].. xvi, 22, 215

**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 línea 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. . 102, 107

**Gene Ontology** Abreviadamente GO. [https://en.wikipedia.org/wiki/Gene\\_ontology](https://en.wikipedia.org/wiki/Gene_ontology) y en <http://geneontology.org/> .. 33, 34, 52, 240, 243, 250, 256, 257, 263, 264, 275, 284

**GEO** NCBI GEO (Gene Expression Omnibus) es una base de datos de datos de expresión obtenidos con microarrays. También tiene algo de datos de secuenciación. Su dirección es <http://www.ncbi.nlm.nih.gov/geo/>.. 56, 90, 94, 101, 113

**GitHub** Es un lugar web para desarrollo colaborativo de software. <http://bowtie-bio.sourceforge.net/bowtie2/index.shtml>. 516

**GTF/GFF** Formato de fichero cuya descripción detallada se puede encontrar en <http://www.ensembl.org/info/website/upload/gff.html>.. 108

**KEGG** <http://www.genome.jp/kegg/> <https://en.wikipedia.org/wiki/KEGG> Es una colección de bases de datos sobre genomas, rutas biológicas, enfermedades, medicamentos y sustancias químicas.. 243, 250, 284

**OrgDb** Tipo de paquetes de anotación centrados en el organismos a que se refieren. Por ejemplo [28, org.Hs.eg.db].. 494

**R** R es un entorno de programación estadístico. <https://www.r-project.org/> . xv, 77, 101

**RNA-Seq** <https://en.wikipedia.org/wiki/RNA-Seq>. . 99

**RPKM** Reads per kilobase per million reads.  $C$  es el total de lecturas alineadas sobre la característica. El total de lecturas que podemos alinear es  $N$  (o tamaño de la librería),  $L$  la longitud de la característica de interés en bp.  $RPKM = \frac{10^9 C}{NL}$ .. 117

**SAM** Sequence Alignment/Map <https://samtools.github.io/hts-specs/SAMv1.pdf> . 108, 110

**SAM** <http://genome.sph.umich.edu/wiki/SAM> y la especificación del formato la tenemos en <http://samtools.sourceforge.net/SAM1.pdf>.. 107, 515

**Samtools** Es un conjunto de programas para trabajar con datos de secuenciación. Ver <http://www.htslib.org/>. En Debian/Ubuntu se instala con apt-get install samtools. En R/Bioconductor tenemos el paquete [100, Rsamtools].. 107

**SRA** NCBI SRA (Sequence Read Archive) 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*.. 101, 107

**STAR** Es un ejecutable que se baja desde GitHub <https://github.com/alexdobin/STAR/archive/master.zip>. Con *STAR aligner* en Google lo encontramos... 107