UNIVERSIDADE DA CORUÑA Universidade Vigo USC UNIVERSIDADE DE SANTIAGO DE COMPOSTELA

Trabajo Fin de Máster

Evaluación y mejora de algoritmos bioinformáticos con base estadística para la detección de macro-indels

Jiménez Otero, Norman

Máster en Técnicas Estadísticas

Curso 2015-2016

Propuesta de Trabajo Fin de Máster

Título en galego: Avaliación e mellora dos algoritmos bioinformáticos con base estatística para a detección de macro-indels

Título en español: Evaluación y mejora de algoritmos bioinformáticos con base estadística para la detección de macro-indels

English title: Evaluation and improvement of statistically-based bioinformatics algorithms for the detection of macro-indels

Modalidad: Modalidad B

Autor: Jiménez Otero, Norman, Universidad de Vigo

Director: de Uña Álvarez, Jacobo, Universidad de Vigo

Tutora: González Castro, Lorena, Gradiant

Breve resumen del trabajo:

La secuenciación del ADN ha sufrido un desarrollo sin precedentes con la introducción en los últimos años de las tecnologías de secuenciación masiva (Next-Generation sequencing, NGS), que gracias a su alto rendimiento están potenciando el desarrollo de nuevas aplicaciones y pruebas biológicas. Desde la aparición del NGS, se han utilizado diferentes algoritmos de machine learning para modelar los perfiles genéticos, aplicándose en diversos estudios que analizan la estructura de las variaciones del genoma humano y tratan de encontrar relaciones estadísticas con las enfermedades. El proyecto GRIDD (desarrollado en Gradiant con la colaboración del CHUS) tiene como objetivo el desarrollo de un algoritmo que permita la detección automática de macroinserciones y macrodeleciones (un tipo específico de variantes estructurales del genoma) que permitirá asistir a los profesionales sanitarios y acelerar el diagnóstico de enfermedades neurometabólicas congénitas. Para realizar el análisis estadístico de los datos se están utilizando actualmente técnicas clásicas de reconocimiento de patrones, como el análisis de componentes principales (PCA) y las máquinas de vectores soporte (SVMs). Entre los objetivos a alcanzar dentro de esta práctica se encuentran el de evaluar diferentes algoritmos y herramientas existentes (como por ejemplo CONTRA[1]), así como la aplicación de técnicas estadísticas que contribuyan a mejorar los resultados de detección del algoritmo que está siendo desarrollado dentro del marco de este proyecto. Por ello, es deseable que el alumno sea capaz de comprender algoritmos implementados en diferentes lenguajes de programación (como Python o C/C++) y de manejar herramientas estadísticas como R o Matlab.

Recomendaciones:

Otras observaciones:

Gradiant se reserva el derecho a participar en el proceso selectivo del estudiante.

Don de Uña Álvarez, Jacobo, Catedrático de Universidad de la Universidad de Vigo, y doña González Castro, Lorena, Investigadora, Ingeniera de Telecomunicaciones de Gradiant, informan que el Trabajo Fin de Máster titulado

Evaluación y mejora de algoritmos bioinformáticos con base estadística para la detección de macro-indels

fue realizado bajo su dirección por don Jiménez Otero, Norman para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal.

En Vigo, a 29 de Junio de 2016.

El director:

Don de Uña Álvarez, Jacobo

La tutora:

Doña González Castro, Lorena

El autor:

Don Jiménez Otero, Norman

Agradecimientos

Quiero agradecer este trabajo, especialmente, a mi tutor Jacobo, que, además, de despertar en mí el interés por las matemáticas en mi último año de carrera en biología y motivarme para estudiar este master, me ha apoyado y ayudado para poder realizar este trabajo, ya que sin su consejo y conocimiento no hubiera sido posible su realización.

También quiero agradecer a Lorena y a Helena su afecto, apoyo y recibimiento, durante mi estancia en Gradiant, y quiero agradecerles haberme dado la oportunidad de poder trabajar con ellas. Sin su ayuda no habría sido posible este trabajo.

También quiero agradecer a mis padres y en especial a Paula el apoyo y ánimos que me ha dado para durante estos últimos años.

Índice general

Resumen

Resumen en español

En este trabajo se introducen los conceptos involucrados en los experimentos de secuenciación de nueva generación (NGS), se explica el origen de los datos que producen estos experimentos, incluidos los sesgos asociados, y el efecto que producen las alteraciones genéticas en los mismos, y se utiliza esa información para proponer un mecanismo generador de los datos. Basándose en este conocimiento, se desarrolla paso a paso un método estadístico para poder detectar alteraciones que incluye la construcción de contrastes de especificación clásicos e intervalos de confianza basados en remuestreo, además se desarrolla un simulador de datos que incluye la posibilidad de generar cuatro tipos de alteraciones que son deleciones y duplicaciones en homocigosis y en heterocigosis.

También se analiza el uso del algoritmo CONTRA (Li et al.,2012), para la detección de variaciones en el número de copias (CNVs) en el mismo contexto, y se discute su uso como parte del proyecto GRIDD (desarrollado en Gradiant con la colaboración del CHUS).

English abstract

This work introduces the concepts involved in next generation sequencing experiments (NGS), explains the origin of the data produced by those experiments, including the associated biases, and the effect genetic disorders have on them. That information is used to propose a data generating mechanism. Based on that knowledge, a step-by-step statistical model is developed to detect disorders, including the proposal of classic specification tests and confidence intervals based on resampling. A data simulator has also been developed to generate four types of disorders, which are deletions and duplications in homozygosis and heterozygosis.

The work also includes an analysis of the CONTRA algorithm (Li et al.,2012) to detect copy number variations (CNVs) in the same context, and it discusses its use as part of the GRIDD project (developed at Gradiant with the collaboration of USC University Hospital Complex, CHUS).

Prefacio

El proyecto GRIDD (desarrollado en Gradiant con la colaboración del CHUS) tiene como objetivo el desarrollo de un algoritmo que permita la detección automática de macroinserciones y macrodeleciones (un tipo específico de variantes estructurales del genoma) que permitirá asistir a los profesionales sanitarios y acelerar el diagnóstico de enfermedades neurometabólicas congénitas.

Entre los objetivos a alcanzar dentro de este trabajo, se encuentran el de evaluar diferentes algoritmos y herramientas existentes (como por ejemplo CONTRA), así como la aplicación de técnicas estadísticas que contribuyan a mejorar los resultados de detección del algoritmo que está siendo desarrollado dentro del marco de este proyecto.

Este trabajo se articula en los siguientes apartados. Primeramente, en los Capítulos 1 y 2 damos una introducción al problema de la detección de alteraciones en bases de ADN en experimentos de secuenciación de nueva generación (NGS). En el Capítulo 3 se considera un modelo estadístico para longitud de reads constante y se proponen distintos métodos de detección basados en dicho modelo. En particular, se aplica un contraste clásico de Neyman-Pearson para variables de Poisson, y se considera un método de corrección para la multiplicidad de contrastes. Se diseñan mecanismos de simulación del citado modelo para el caso de no alteración, y se aplica a la construcción de límites de confianza; y también mecanismos de simulación de las cuatro tipologías elementales de alteración. Ambos mecanismos de simulación se utilizan para estudiar la validez del contraste. Finalmente, se propone un estadístico resumen que integra los valores de cobertura de todas las bases de la región bajo estudio, y se investiga su comportamiento en simulaciones. El Capítulo 4 repite los apartados anteriores pero incorporando una nueva dificultad: la posible aleatoriedad en la longitud de los reads. El trabajo continúa con un enfoque basado en el análisis de datos funcionales (Capítulo 5), y con una descripción del algoritmo CONTRA (Capítulo 6), que es una herramienta que se ha propuesto en la literatura para detectar alteraciones a nivel global, y cuyo comportamiento hemos estudiado durante la práctica en Gradiant.

Capítulo 1

Introducción

Debido a la elevada demanda de secuenciación y a la necesidad de reducir los costes, se ha producido un desarrollo, sin precedentes, de las tecnologías de secuenciación masiva (Next-Generation sequencing, NGS). A su vez, la expansión de estas técnicas hace necesario el desarrollo de herramientas informáticas capaces de analizar el volumen de datos y de buscar relaciones estadísticas entre ellos. Los fines de los experimentemos de secuenciación son variados y que van desde: el conocimiento de las secuencias de ADN, como en el caso del Proyecto Genoma Humano, por citar un ejemplo, hasta la aplicación clínica para conocer y diagnosticar enfermedades.

Al tratarse de un campo actual, existe gran variedad de software destinado a trabajar con los datos obtenidos de los experimentos de secuenciación, con enfoques muy distintos. Además, en su gran mayoría se tratan de herramientas en desarrollo, y con escasa validación en experimentos reales.
El propósito del proyecto es analizar algoritmos, para el caso concreto, de detección de macroinserciones y macrodeleciones y, en particular, el uso del algoritmo CONTRA (Li et al., 2012). Pero por el feedback obtenido y por el análisis llevado a cabo en este trabajo, nos damos cuenta de que CONTRA, no es un algoritmo adecuado para la tarea, ya que no presenta mucha fiabilidad en los resultados.

La variación en el número de copias (CNVs) son una variación estructural del ADN que se manifiesta en forma de macroinserciones y macrodeleciones; son segmentos de ADN igual o mayor de 1 kb cuyo número de copias es variable si se compara con un genoma de referencia.
Estas variaciones del ADN se encuentran en alrededor del 13 % del genoma humano. Lo que diferencia las CNVs de los polimorfismos de un solo nucleótido (SNPs), es que estos últimos solo afectan a una base.

Para analizar estos algoritmos es necesario un volumen grande de muestras, para poder planificar un diseño experimental que permita evaluar los distintos algoritmos.

Un problema que nos hemos encontrado es la escasa presencia de muestras validadas, lo que nos lleva a recurrir a simuladores de muestras. Existen gran variedad de simuladores disponibles, y adaptados diversos contextos, pero nos encontramos muchos problemas a la hora de conseguir un simulador que se adapte a nuestros propósitos, y que arrojara simulaciones de confianza.

El simulador más cercano a nuestro contexto, es TargetedSim, que es de hecho el simulador utilizado por los autores de CONTRA. Pero se trata de una herramienta no validada y que ya no goza de soporte por parte de su autor (https://sourceforge.net/projects/targetedsim/). Además, viendo los perfiles de cobertura de las muestras simuladas extraídas del reposito de CONTRA, no parece muy razonable que en la realidad podamos encontrar situaciones similares, ya que las alteraciones aparecen todas simuladas en las colas y dando lugar a perfiles que no se parecen a los observados en casos reales

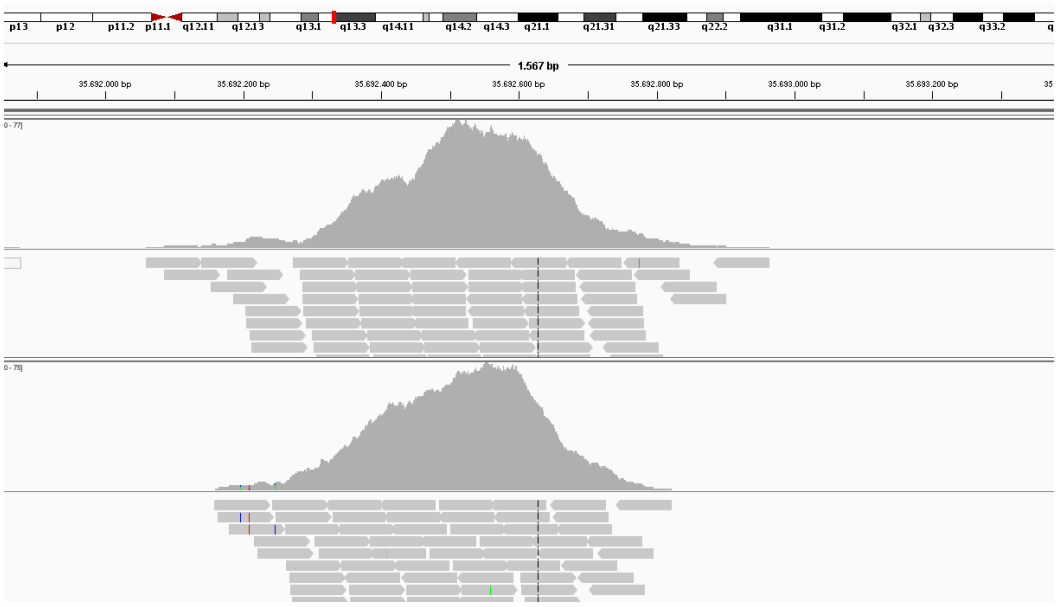

Figura 1.1: Imagen de perfiles de cobertura simulados con TargetedSim; arriba, una duplicación en las colas, y, abajo, el perfil sin alteración

de CNVs. Esto se ve en la Figura 1.1, donde la región simulada con alteración (arriba) es más larga que la región normal, y el perfil alterado no presenta una cobertura lo suficientemente alta como para tratarse de una duplicación, como veremos más adelante.

Llegado a un punto del proyecto, por la ineficacia del algoritmo base de trabajo (CONTRA), y por la falta de muestras, hemos optado desarrollar nuestro propio mecanismo de detección; que es el que está propuesto en este trabajo. Además por el enfoque que se le ha dado, permite extender la herramienta a la detección de anomalías distintas a las CNVs, pero que tengan repercusión detectable en los perfiles de cobertura, alterando la generación de estos perfiles estándar, como es el caso de las SNPs.

El enfoque que se le ha dado al estudio pasa por identificar el comportamiento de los datos, para poder construir contrastes e intervalos de confianza y, además, poder desarrollar un simulador que permita poder generar un número suficiente, y controlado, de muestras, con el fin de desarrollar un plan experimental que permita validar nuestra propuesta y así solventar el problema de la escasez de muestras.

Como parte del desarrollo del proyecto, se ha trabajado en adaptar el código Python original de CONTRA al software estadístico R, para poder analizar el algoritmo paso a paso en un lenguaje interpretado. Todos los pasos del algoritmo se encuentran en el script Contra.R. CONTRA trabaja con

archivos con extensión BAM y BED fundamentalmente, aunque admite archivos en formato SAM, lo que requiere el uso de rutinas externas y que se han adaptado en su mayoría a R. Pero para la ejecución completa del script se requiere la ejecución en un entorno de Linux, ya que desde el mismo script se llama a diversas rutinas externas de BEDTools (Quinlan & Hall, 2010) y SAMtools (Li H. et al. 2009), que solo trabajan en entorno de Linux.

Además se ha añadido código a este script que permite extraer la información de la cobertura de los archivos que se carguen en el script, y que se han utilizado para obtener los datos con los que hemos trabajo en los demás scripts, tanto en los simuladores, contrastes e intervalos de confianza como en los scripts de datos funcionales.

Capítulo 2

Secuenciación de nueva generación (NGS)

En la actualidad, las técnicas de targeted resequencing (TR) permiten reducir costes y esfuerzos al centrar los esfuerzos de la secuenciación en determinas regiones del ADN. Usando secuenciación de nueva generación (NGS) reducimos también costes y obtenemos procesos muy automatizados para la aplicación clínica. La targeted resequencing consiste en seleccionar una o varias secuencias de interés determinadas, lo que incluye poder trabajar con los exones, que son la parte que se traduce a proteínas del ADN. La secuenciación de nueva generación de estas secuencias o regiones seleccionadas del ADN se realiza con distintas técnicas, como Illumina, Solid, etc. Estos procesos de secuenciación varían en la química utilizada, pero el proceso es en esencia el mismo. Las diferencias fundamentales en las distintas técnicas de secuenciación de nueva generación residen en el tamaño de las lecturas que se obtienen, en la eficiencia, en el tamaño máximo que podemos secuenciar y en su coste.

En un proceso de secuenciado Illumina (www.illumina.com), de forma muy general, partimos de la secuencia de ADN de doble cadena obtenida de un individuo. Esta doble cadena se fragmenta en trozos y esta fragmentación se produce al azar, de modo que se producen fragmentos de distinta longitud; generalmente, estos fragmentos tienen un tamaño en torno a $100 - 150$ bases. A cada fragmento se le añaden unos fragmentos de ADN artificial en los extremos, insertos, los cuales contienen unas secuencias $P5$ y $P7$ que son "pegajosas" y sirven para unirse al soporte de secuenciación. Estas secuencias de ADN con los insertos se desnaturalizan y así obtenemos ADN de cadena sencilla. Los extremos $P5$ y $P7$ se adhieren al soporte de secuenciación formando un arco o puente (bridge). Entonces empieza el proceso de secuenciación, en el que se amplifica cada fragmento. En dicho proceso se producen ciclos y por cada ciclo se añade un aminoácido marcado con fluorescencia a cada fragmento; gracias a esta luz emitida, identificamos qué base se ha unido y de esta forma conocemos la secuencia aminoacídica de cada fragmento.

Estas secuencias obtenidas en el proceso son lo que conocemos como reads o lecturas, y que darán lugar a los datos con los que trabajan algunos de los algoritmos para detectar variaciones en el ADN.

El proceso esquematizado puede verse en la Figura 2.1

2.1. Forma de los datos

Para obtener los datos, estas secuencias o reads se alinean frente a un genoma de referencia y, de este modo, podemos localizar el sito de origen en el genoma de los reads generados en el experimento, y así obtener lo que denominaremos cobertura. Un ejemplo de un proceso de alineado se muestra en

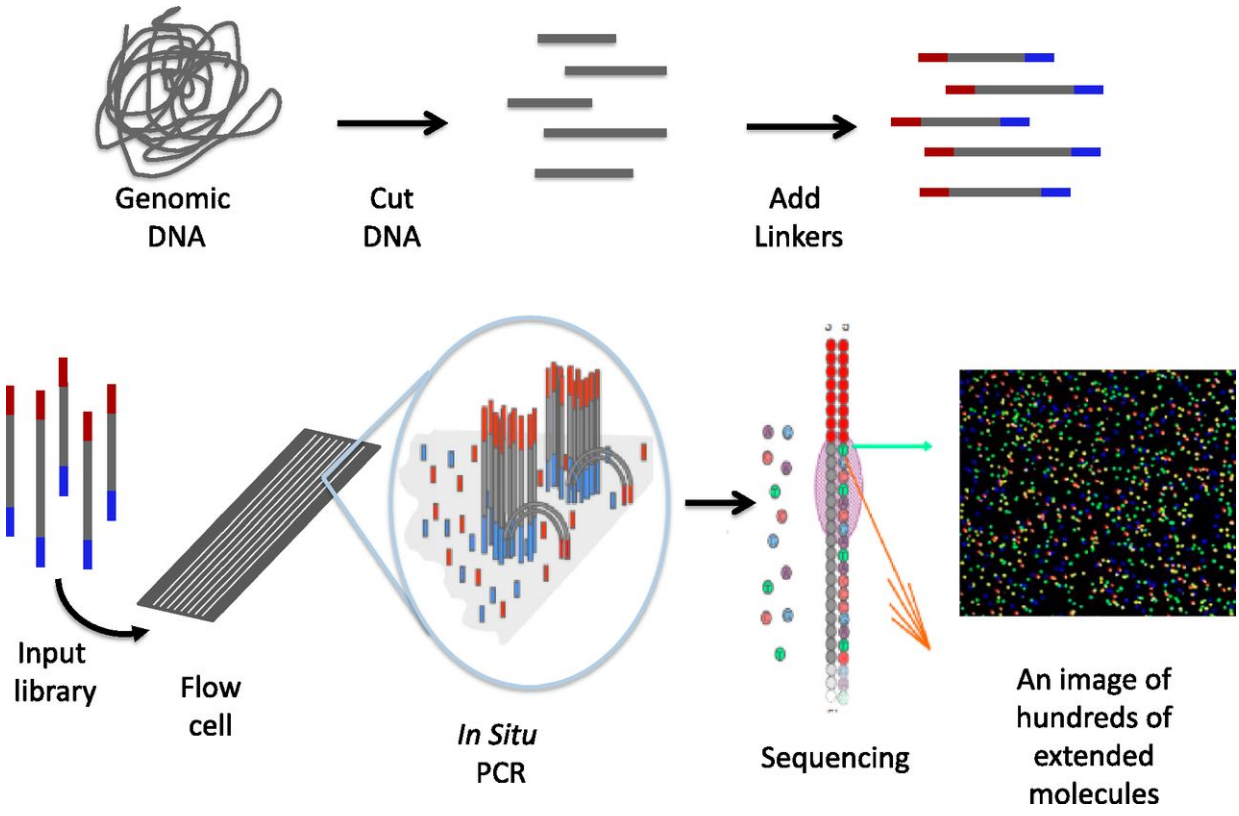

Figura 2.1: Proceso de secuenciación. El proceso se inicia mediante la fragmentación al azar de ADN , se añaden adaptadores a los fragmentos para generar una biblioteca. Las bibliotecas se introducen en un soporte que contiene oligonucleótidos complementarios a los adaptadores en la superficie. Se añade una sola base marcada por cada ciclo del secuenciador, y se toma una imagen de la base añadida. (Figura extraida de: Jill M, 2013)

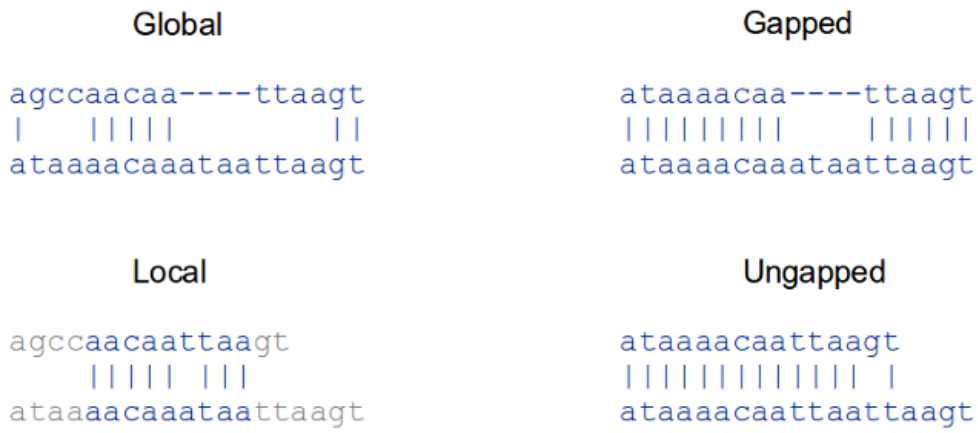

Figura 2.2: Diferentes formas de alinear los reads (secuencia superior), frente a una secuencia de referencias previamente conocida (secuencia inferior).(Figura extraida de: Joaquin Cañizares, Jose Miguel Blanca. Grado de Biotecnologia. 2015)

la Figura 2.3.

El alineamiento no es una cuestión sencilla. En experimentos de resecuenciación este paso se denomina mapeo. En el mapeo el proceso de comparación del read frente a la secuencia de referencia se realiza para cada read individualmente, y este alineamiento puede ser de diversos tipos, como se ve en la Figura 2.2.

El proceso de mapeo es lento debido al gran número de reads que se procesan, y el método que se suele utilizar es la transformación Burrows Wheeler, (M.Burrows & D.Wheeler, 1994), que es el método más rápido, (Li y Durbin, 2009), El método funciona compactando la secuencia de referencia en estructuras que facilitan la búsqueda de aparcamientos perfectos. Esta implementado en las herramientas BWA y Bowtie.

Como se intuye en la Figura 2.2, al cambiar el tipo de alineamiento estamos afectando a la información, y esto tiene repercusión en los datos con los que trabajamos. Pero el objetivo de este trabajo no es entrar en los métodos de alineamiento, así que nos centramos en la problemática general y el uso de alineamientos globales y gapped, y no en el efecto sobre los datos de usar uno u otro tipo de mapeado, ya que no solo afectaran a la cobertura, sino a la forma en la que se manifiestan en la cobertura las alteraciones que pretendemos detectar.

De forma general, la cobertura (coverage) se define como el número promedio de reads que representan a un determinado nucleótido en la secuencia total reconstruida. Puede ser calculada a partir del largo original del genoma secuenciado (G), el número de secuencias producidas (N) y el largo promedio de dichas secuencias (R), como;

$$Cobertura = N \times \frac{R}{G}$$

Por ejemplo, un genoma hipotético de 2000 bases, secuenciado con 24 reads de 350 bases de largo en promedio, tiene una cobertura de: $24 \times (350/2000) = 4.2$ X. Este valor significa que el genoma fue secuenciado 4.2 veces. Cuanto mayor es la cobertura, hay menor posibilidad de error en en la secuencia

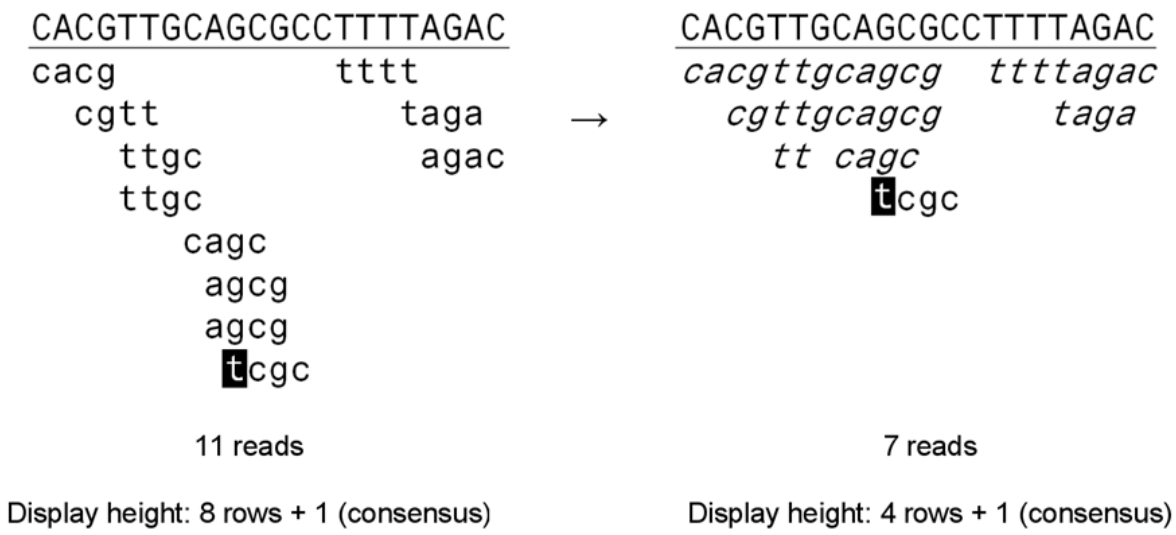

Figura 2.3: Los reads se alinean respecto a un genoma de referencia, lo que nos permite conocer su posición en el genoma. (Figura extraída de: Sequence assembly with MIRA 4, 2014)

final. Este valor obtenido representa la cobertura general de un experimento pero, si nos fijamos a nivel de una base, la cobertura se representa en unos gráficos como el que vemos en la Figura 2.4. Estos gráficos se llaman perfiles de cobertura y nos muestran en cuántos reads aparece una base determinada.

El objetivo de este trabajo es tratar de modelizar el comportamiento de la cobertura a nivel de base, con el fin de encortar un modelo que permita aplicar técnicas estadísticas a este tipo de datos.

La fragmentación del ADN es aleatoria y genera un determinado rango de tamaños de los reads, pero la longitud de los reads depende del número de ciclos que se lleven a cabo en el experimento de secuenciación, y la posición de estos reads se distribuye de forma aleatoria por las regiones seleccionadas para la secuenciación.

Con las secuencias alineadas frente al genoma de referencia, podemos ver, en la Figura 2.4, que en cada zona existen muchos reads que se solapan y, como vemos, cada base aparece en varios reads. De este modo, obtenemos la cobertura del experimento a nivel de la base, que es el número de veces que cada base es secuenciada, y en general esta cobertura se espera que sea aproximadamente la misma para todas las bases.

En la Figura 2.4 se ve, por el perfil de cobertura, que la cobertura no es igual en toda la región, sino que aparecen colas a los extremos en las que cae la cobertura. Este fenómeno se produce porque sólo se tienen en cuenta los reads que encajan completamente en la región y, los que no lo hagan, no son tenidos en cuenta. Esto hace que, conforme nos acercamos a los extremos de la región, menos reads son tenidos en cuenta.

Lander ES. & Waterman MS. (1988) estimaron que el número esperado de veces que cada base es secuenciada, asumiendo que los reads se distribuyen de forma aleatoria, sigue una distribución de Poisson. Pero el número de veces que es secuenciada cada base no es independiente de las demás, sino que está relacionada con la longitud del read.

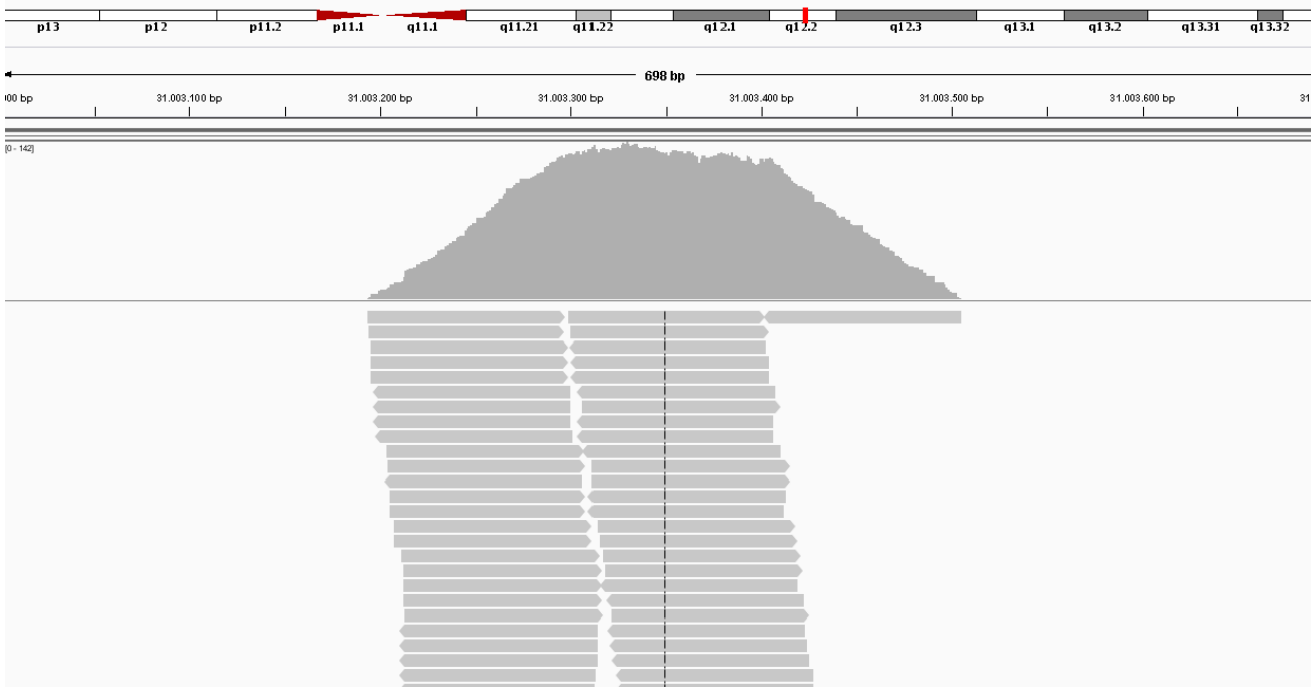

Figura 2.4: Lecturas de un experimento de secuenciación (reads) y perfil de cobertura que generan después de ser alineadas frente a un genoma de referencia. La forma de pirámide truncada de la cobertura es producida porque los reads que no se encuentren completamente dentro de la región son eliminados

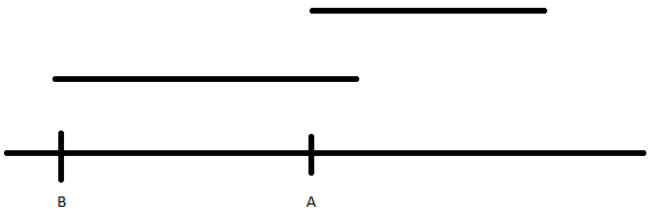

Figura 2.5: Representación de 2 bases A y B y 2 reads, uno que empieza en la base A y otro que empieza en la base B y se solapa con la base A

El fenómeno de la dependencia se aprecia sobre todo cuando tenemos reads muy largos, como vemos en las Figuras 2.8 y 2.9. Este fenómeno sucede porque el read no es un único evento de una base, sino que es un evento para un número de bases equivalente a la longitud del read de modo que, como vemos en la Figura 2.5, la cobertura de cada base (A) se puede expresar como las lecturas que comienzan en esa base (A), más las lecturas que comenzaron en una base anterior (B) y solapan con esa base (A).

Esto es, la probabilidad de una Poisson de parámetro lambda (λ), más la suma acumulada del número de eventos Poisson observados un número de bases anteriores equivalentes al tamaño de los reads.

Lo vemos con un ejemplo. Supongamos que tenemos unos reads de tamaño de 3 (tiene una longitud de 3 bases, y esta longitud es constante), una región de 6 bases y una cobertura media, λ, de 2 reads por base (cobertura de la base). Al simular una Poisson de parámetro λ igual a 2 obtenemos los reads que empiezan en cada una de las 6 bases, de forma independiente e idénticamente distribuidas: en nuestra simulación, A=3, B=5, C=2, D=1, E=1 y F=3. Cada read tiene una longitud de 3 bases; eso significa que da cobertura a 3 bases a partir de la base en la que se coloca de modo que, sumando el resultado de la Poisson para cada base a los eventos observados para las 2 bases anteriores, obtenemos el perfil de cobertura, esto es: A=3, B=3+5, C=3+5+2, D=5+2+1, E=2+1+1 y F=1+1+3. En este ejemplo vemos cómo la distribución de la cobertura para cada base no es independiente de las anteriores; sólo es independiente en el caso de que los reads son de tamaño de 1 base (Figura 2.7). Y los reads que no solapen completamente en la región son eliminados, lo que produce la caída de la cobertura en las colas. El esquema de este ejemplo puede verse en la Figura 2.6.

En el caso extremo contrario de una región de 100 bases y reads de tamaño de 100 bases, como solo contaremos los reads que solapen al 100 % con la región de interés, el resultado será una línea horizontal y todas las bases tendrán la misma cobertura; sólo se colocarán reads en la primera base de la región. Esto se ve en la Figura 2.9.

2.2. Errores de secuenciación y longitud de los reads

En el ejemplo sencillo anterior hemos supuesto que los reads son de longitud 3 y que esta longitud es constante. En un experimento de secuenciación sin errores, la longitud de los reads es constante e igual al número de ciclos de secuenciador realizados.

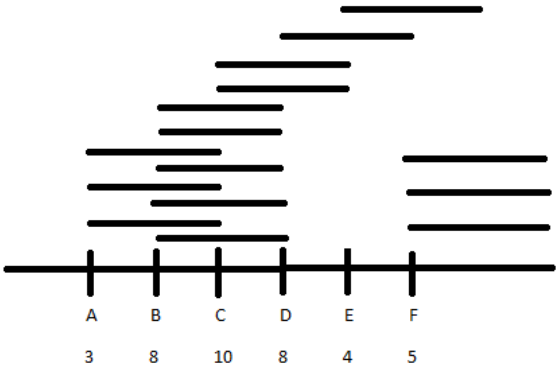

Figura 2.6: Representación de 6 bases y reads que se reparten en la región y la cobertura de cada base

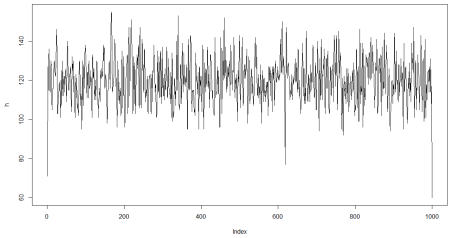

Figura 2.7: Simulación con longitud de read de 1 base y región de 1000 bases

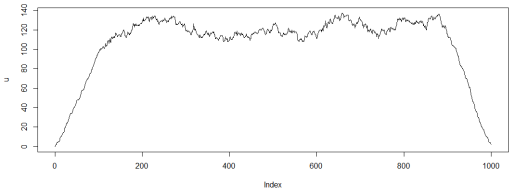

Figura 2.8: Simulación con longitud de read de 500 base y región de 1000 bases

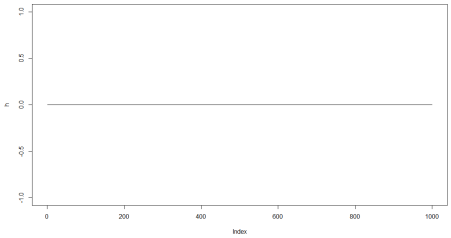

Figura 2.9: Simulación con longitud de read de 1000 bases y región de 1000 bases

De modo que en un proceso 100 % eficaz, todos los reads serán del mismo tamaño. Sin embargo, aparecen reads de longitud menor debido a un proceso de adiciones de bases que pueden adelantarse o retrasarse a su ciclo de incorporación: estos fenómenos se llaman *prephasing* y *phasing*.

El efecto del *prephasing* y *phasing* se puede mitigar mediante la aplicación de correcciones. Las muestras suelen presentar solamente un "pequeño" número de secuencias diferentes.

De modo que, si queremos tener en cuenta los fenómenos de *prephasing* y *phasing*, debido a que no se ha realizado un proceso de corrección, o debido a que los fenómeno de *prephasing* y *phasing* se producen en exceso, podríamos asumir que la longitud de estos reads es una variable aleatoria; serían reads de tamaño aleatorio en un intervalo del tamaño mínimo de los fragmentos, y el número de ciclos en el secuenciador que se realicen, siempre que el número de ciclos no excedan el tamaño máximo de los fragmentos. Este contexto se planteara más adelante, pero de forma general trabajaremos suponiendo reads de longitud constante (Capítulo 3).

Hay que mencionar que muchas veces se producen errores de secuenciación (Benjamín Rodríguez-Santiago y Lluís Armengol, 2012), pero estos errores se suelen producir al azar. No obstante, determinadas plataformas parecen producir específicamente cierto tipo de errores. En la plataforma Illumina, por ejemplo, los errores se correlacionan con la posición en el read, acumulándose éstos con mayor frecuencia hacia el final de él. Por el contrario, en la plataforma Roche/454, los errores no dependen de la posición en el read, sino que tienden a acumularse alrededor de secuencias de homopolímeros (regiones de ADN con 6-7 o más nucleótidos idénticos consecutivos).

Por lo tanto, en algunas plataformas, los errores no se distribuyen aleatoriamente a lo largo de la secuencia, sino que se repiten en las mismas regiones incluso en experimentos de secuenciación independientes. Para cuantificar los errores y evitar que afecten al proceso, los reads obtenidos con el secuenciador pasan un análisis previo en el que se eliminan reads mal alineados y se cuantifica la precisión de la secuenciación.

2.3. Sesgo GC

Hasta ahora hemos repasado los factores que influyen en el comportamiento de la cobertura, pero queda un último fenómeno, que es el sesgo GC; este sesgo es debido a la excesiva o escasa aparición de bases de guanina y citosina en las secuencias. El sesgo GC, o sesgo por el contenido en guanina-citosina, se produce en experimentos de secuenciación de nueva generación, en especial en Illumina (Aird et al., 2011). Cuando el contenido GC es muy alto o muy bajo en una región, esto afecta a la cobertura, reduciéndola.

Varios autores han propuesto diferentes métodos para corregir este sesgo y se han desarrollado varias aplicaciones, por lo que eliminar el sesgo GC es un paso previo importante al análisis de las secuencias. Concretamente se ha propuesto un método de regresión polinomial basada en el agrupamiento de reads en función de su contenido GC (Boeva et al, 2011), y también se ha desarrollado un método para estimar y corregir sesgo debido al contenido GC, que trabaja a nivel de base (Benjamini & Speed, 2011).

Descritos los procesos involucrados en la secuenciación, estamos en disposición de construir un modelo que explique este comportamiento y que nos permita llevar a cabo análisis sobre la cobertura.

Capítulo 3

Modelo

3.1. Modelo: R constante

La información de cobertura disponible se refiere a la región de interés de f bases. En particular, para cada base b ($b = 1, \ldots, f$), tenemos N_b, que es el número de reads que empiezan en la base b, y $R_{b,i}$ ($i = 1, \ldots, N_b$), que es la longitud de cada uno de los reads de cada base b y que asumimos constante; por lo tanto, $R_{b,i} = R$, $b = 1, \ldots, f$, $i = 1, \ldots, N_b$.

Para calcular la cobertura de una base b particular, definimos la contribución de una base j a tal cobertura; es lo que llamaremos $X_{b,j}$:

$$X_{b,j} = \begin{cases} N_j \sim Poisson(\lambda) & si\ j + R - 1 \leq f\ y\ j + R - 1 \geq b \geq j \\ 0 & en\ otro\ caso \end{cases}$$

donde N_j es el número de reads que empiezan en la base j, que asumimos sigue una $Poisson(\lambda)$ (Lander y Waterman, 1988). La condición $j + R - 1 \leq f$ nos garantiza contar solo los reads que entren en la región, y la condición $j + R - 1 \geq b \geq j$ son las posiciones donde empiezan los reads que cubren la base b.

Obtenemos entonces la cobertura de la base b, Cov_b, que se define como:

$Cov_b = \sum_{j=1}^{f} X_{b,j}$

Cov_b es una suma de elementos de valor cero o N_j; para quedarnos sólo con los elementos que son distintos de 0, los cuales son los $X_{b,j}$ que aportan información a la base b, escogemos los j tales que: $j = max(b - R + 1, 1), \ldots, min(b, f - R + 1)$, donde $max(b - R + 1, 1)$ es el índice de la primera base que aporta información a b y es ≥ 1 (comienzo de la región). Y $min(b, f - R + 1)$ es el indice de la última base que aporta información a b y es $\leq b$.

Sabemos que la contribución de cada una de las j ($j = max(b - R + 1, 1), \ldots, min(b, f - R + 1)$) bases a la cobertura de la base b vale N_j, y que $N_j \sim Poisson(\lambda)$. Además, las contribuciones de las distintas bases j son independientes e idénticamente distribuidas, por lo que podemos aplicar la propiedad de reproductividad de la Poisson.

Entonces, la cobertura de la base b, Cov_b, se puede expresar de la siguiente manera:

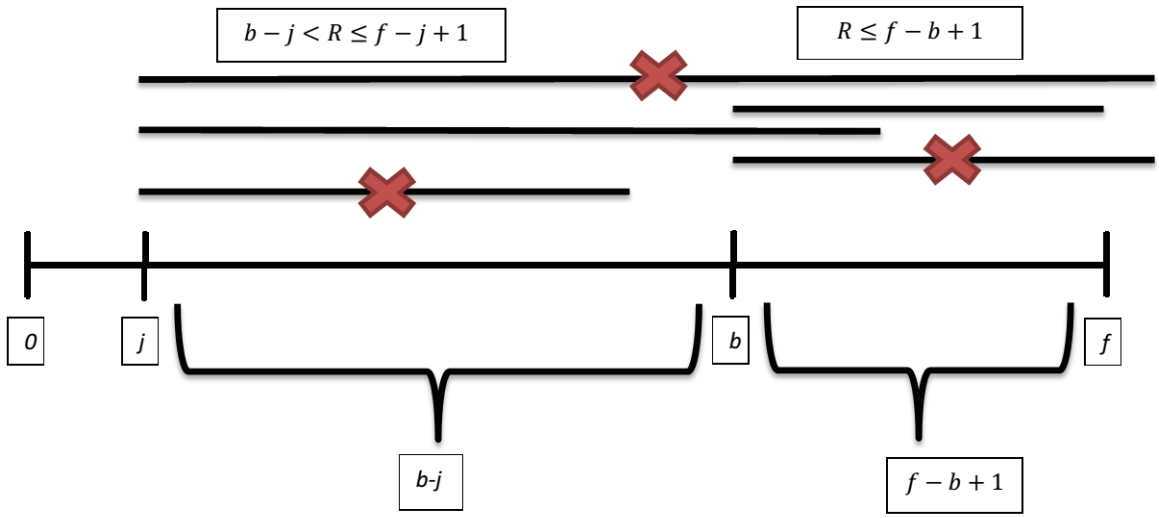

Figura 3.1: Representación esquemática del calculo de la cobertura para la base b

$$Cov_b = \sum_{j=max(b-R+1,1)}^{min(b,f-R+1)} X_{b,j} \sim Poisson(\lambda[min(b, f-R+1) - max(b-R+1, 1)+1]) \quad (1)$$

Simplificando, y asumiendo $f - R + 1 \geq R$ (es decir, $R \leq \frac{(f+1)}{2}$) :

$$Cov_b = \begin{cases} Poisson(\lambda[R]) & si\ f - R + 1 \geq b \geq R \\ Poisson(\lambda[f - b + 1]) & si\ b \geq f - R + 1 \\ Poisson(\lambda[b]) & si\ b \leq R - 1 \end{cases}$$

El esquema de este procedimiento de cálculo de la cobertura para cada base se muestra en la Figura 3.1.

Como vemos, en la Figura 3.1, en las bases j ($j = 1, \ldots, b$) suceden eventos que son los reads que empiezan en la base j, pero solo se tienen en cuenta los reads que caben entre la base j y la última base $f(f - j + 1)$; se añade un $+1$ por el motivo de que la última base cuenta y también puede ser solapada. Como la longitud de los reads es constante, el valor esperado en la base b serán las sumas de los reads desde la base $b - R + 1$, o la base 1 si b está más cerca de la primera base que R, hasta la base b, o la base $f - R + 1$ en el caso de que la base b esté más cerca del final que R.

3.2. Parámetro general del experimento λ

Para el contexto, el parámetro λ es una constante que viene dada por el experimento. Este parámetro se interpreta como el valor medio de N_b, los reads que empiezan en cada base b ($b = 1, \ldots, f - R$). Nosotros observamos el número de reads que empiezan en cada base desde 1 hasta $f - R$ (debido a que los reads desde $f - R + 1$ hasta f son eliminados), N_b ($b = 1, \ldots, f - R$), y estos eventos son independientes e idénticamente distribuidos para todas estas bases. Bajo el supuesto de que estos eventos siguen una $Poisson(\lambda)$, si no conocemos el parámetro λ lo calculamos para el experimento general (conjunto de todas las regiones), recurriendo al estimador de máxima verosimilitud de λ, que es la media del número de reads que empiezan en cada base b ($b = 1, \ldots, f - R$) para todas las regiones k,

$(k = 1, \ldots, K)$, es decir:

$\lambda = \frac{1}{k} \sum_{k=1}^{k} \frac{1}{f-R} \sum_{b=1}^{f-R} N_{b,k}$

donde $N_{b,j}$ denota el número de reads que empiezan en la base b para la región K.

Un problema que tenemos es si alguna región presenta alteraciones en el número de reads por base, λ, de algunas de las bases. Debido a la presencia de una deleción o duplicación, el parámetro λ de esa base o bases será distinto, de modo que estaremos cometiendo error en la media si incluimos esta base o bases con λ diferente. Por tanto, el parámetro es conveniente estimarlo para un conjunto grande de regiones, con el fin de mitigar el efecto de la inclusión de estas bases con un λ diferente en la media. Esta solución es razonable debido a que la presencia esperada de alteraciones es baja. Otra alternativa es utilizar una muestra control que no presente alteraciones para calcular el parámetro λ. Con el valor de λ y la información de la longitud de los reads, R, podremos calcular los valores esperados.

Si disponemos de una región control para la misma región en la que buscamos alteraciones podemos calcular λ con los los reads de dicha región control de la forma:

$\lambda = \frac{1}{f-R} \sum_{b=1}^{f-R} N_b$

3.3. Contraste basado en el modelo

Construimos un test, para contrastar la cobertura en una base cualquiera b de la región de interés. El valor esperado de la variable cobertura en la base Cov_b es, de acuerdo con el modelo (1), $\Lambda = \lambda[min(b, f - R + 1) - max(b - R + 1, 1) + 1]$:

Para una base dada b, la hipótesis nula de interés es $H_0 : \Lambda = \Lambda_0$, la cual se testea frente a una alternativa $H_1 : \Lambda \neq \Lambda_0$, pero para los casos más sencillos de alteraciones, la hipótesis alternativa se reduce a unos posibles valores, $H_1 : \Lambda = \Lambda_1$, donde Λ_0 y Λ_1 son constantes conocidas, permitiéndonos construir un contraste con hipótesis nula y alternativa simples. En la práctica, Λ_0 es el valor de Λ calculado con el modelo (1) en condiciones estándar (sin duplicaciones o deleciones). Respectivamente, Λ_1 denota el valor de Λ para una alteración específica en la base b (deleción o duplicación). La muestra disponible para realizar el test se reduce a una única observación para la variable Cov_b.

Partiendo de estos supuestos construimos el test utilizando el Lema Neyman-Pearson .

3.3.1. Lema Neyman-Pearson

Apoyándonos en que tanto la hipótesis nula como la alternativa son simples podemos recurrir al lema de Neyman-Pearson (N-P) para encontrar el test uniformemente más potente (UMP). Partimos de una muestra de tamaño 1 ($n = 1$) de la variable $Cov_b \sim Poisson(\Lambda)$;$\Lambda \in \Re^+$, ya que solo contamos con una observación de cobertura por cada base.

Nuestras hipótesis son:

$H_0 : \Lambda = \Lambda_0$ y $H_1 : \Lambda = \Lambda_1$

Y Λ_1 puede tomar los valores, en las situaciones más sencillas, que se presenta en el Cuadro 3.1.

El test viene dado por:

$$\phi(x) = \begin{cases} 1 \ si \ \delta(x) > K \\ \gamma \ si \ \delta(x) = K \\ 0 \ si \ \delta(x) < K \end{cases}$$

donde el estadístico $\delta(x)$ es:

$\delta(x) = \frac{[L(\Lambda_1|x)]}{[L(\Lambda_0|x)]}$, x denota la única observación disponible, y $L(\Lambda|x)$ la función de verosimilitud del modelo:

$L(\Lambda|x) = \prod_{i=1}^{n=1} f_i(\Lambda|x) = f(\Lambda|x) = \frac{e^{-\Lambda}\Lambda^x}{x!}$

Rechazaremos la hipótesis nula cuando el cociente $\delta(x)$ sea grande: $\delta(x) > K$.

Desarrollando $\delta(x)$:

$\delta(x) = \frac{f(\Lambda_1|x)}{f(\Lambda_0|x)} = \frac{\frac{e^{-\Lambda_1}\Lambda_1^x}{x!}}{\frac{e^{-\Lambda_0}\Lambda_0^x}{x!}} = e^{\Lambda_0-\Lambda_1}(\frac{\Lambda_1}{\Lambda_0})^x < K$

Vemos que en $\delta(x)$ la única parte aleatoria es x, por lo tanto, podemos simplificar la expresión a:

$x < S$ (si $\Lambda_1 < \Lambda_0$) o bien a : $x > S$ (si $\Lambda_1 > \Lambda_0$)

Pero sabemos que x sigue una $Poisson(\Lambda_0)$ bajo H_0 y al ser una variable discreta el test queda de la siguiente forma (para $\Lambda_1 > \Lambda_0$):

$$\phi(x) = \begin{cases} 1 \ si \ x > S \\ \gamma \ si \ x = S \\ 0 \ si \ x < S \end{cases}$$

Entonces la región de rechazo de tamaño α esta dada por los valores de S y γ que satisfacen:

$\alpha = E[\phi_0(x)] = P_0(x < S) + \gamma P_0(x = S)$

Y la probabilidad P_0 representa la de una $Poisson(\Lambda_0)$. Analogamente para el caso $\Lambda_1 < \Lambda_0$

Tanto para la situación de $\Lambda_1 > \Lambda_0$ como para $\Lambda_1 < \Lambda_0$ vemos que x y γ solo dependen de Λ_0 y S y no de Λ_1. Por tanto, se demuestra que el test anterior es el UMP para $H_0 : \Lambda = \Lambda_0$ frente a $H_1 : \Lambda = \Lambda_1$ en cualquiera de las cuatro situaciones recogidas en el Cuadro 3.1.

En la práctica, para realizar el contraste calculamos el p-valores de x de la forma: $P_0(Poisson(\Lambda_0) \leq x)$ para $\Lambda_1 < \Lambda_0$ y de la forma $P_0(Poisson(\Lambda_0) \geq x)$ para $\Lambda_1 > \Lambda_0$. El contraste de la cobertura de las regiones, usando los p-valores, se encuentra en el script *contraste.R* (página 71), en el que también se incluye un ajuste de los p-valores de Benjamini y Hochberg (1995).

3.3.2. Comparaciones múltiples

En este caso, ya que se realiza un contraste para cada base, estamos ante el problema clásico de tener un número de comparaciones elevado, produciendo que la aplicación reiterada del contraste para un nivel de significación α prefijado pueda conducir a un número de rechazos de la hipótesis nula grande, aunque no existan diferencias reales.

Generalmente en experimentos genéticos en los que realizan comparaciones entre muestras normales y alteradas en los que las diferencias son significativas, como es este caso, y en los que se está dispuesto a aceptar un determinado porcentaje de falsos positivos en los contrastes, se suele emplear el método de Benjamini y Hochberg (1995).

En este método se ordenan los p-valores obtenidos del contraste de cada base de menor a mayor, y se les asignan a estos p-valores los índices $i = 1, \ldots, f$, p_i, y se compara cada p-valor, p_i, con un valor crítico que se obtiene de la forma: $\frac{i}{m} \times q$, donde i es el índice, m es el número total de test que se realizan, y es igual a f, $(m = f)$, y q es la tasa de falsos positivos que estamos dispuestos a asumir. El mayor p-valor, p, que cumpla $p_i \leq \frac{i}{m} \times q$ es significativo y todos los p-valores de índice menor serán significativos también.

El motivo de utilizar este ajuste viene derivado de la naturaleza dependiente de los datos. El método de Benjamini y Hochberg (1995), funciona cuando las múltiples comparaciones son independientes, pero también cuando existe una dependencia positiva entre los tests. Si usamos un método que no tenga esto en cuenta producimos una sobrecorrección de los p-valores, perdiendo mucha potencia estadística, de modo que el usar un método de corrección adecuado se vuelve vital en este contexto, en el que se realizan un gran número de contrastes. En un contexto como este, la cobertura de una base, Cov_b $(b = 1, \ldots, f)$, presenta cierta dependencia positiva de la cobertura de un número j bases anteriores, Cov_{b-j}, bajo la hipótesis nula. Esta última condición es la que garantiza el buen funcionamiento del método. (Benjamini y Yekutieli, 2001).

3.3.3. Contraste aplicado a una región real

Con la teoría anterior y la información necesaria proveniente del experimento, construimos nuestro contraste y lo aplicamos a una muestra real. Primero necesitamos obtener de nuestra región problema N_b (número de reads que empiezan en cada base), la longitud de los reads de cada base, R (constante) y la Cov_b observada para la región problema. Con el número de reads que empiezan en cada base, N_b, calculamos λ con todas las bases desde 1 hasta f. Estos serán los reads esperados para cada base. Después, se suman los λ desde la base $max(b - R + 1, 1)$ hasta la base $min(b, f - R + 1)$, para cada base b, y de esta forma obtenemos el valor esperado bajo H_0.

La región representada en la Figura 3.2 región pertenece a una muestra real sin alteraciones con un tamaño de $f = 312$, un tamaño de reads $R = 101$ y un parámetro experimental λ=1.2547171. Del mismo individuo, pero con la totalidad de las regiones, se obtuvo la información del tamaño de los reads R.

Utilizando el test que construimos en la sección 3.3.2, contrastamos la información observada de la región de la Figura 3.2. Realizamos un contraste de hipótesis nula para cada base b, $H_0 : \Lambda_b = \Lambda_0$, frente a hipótesis alternativa $H_1 : \Lambda_b = \Lambda_1$ con $\Lambda_1 < \Lambda_0$, lo que implica que estamos buscando una deleción. El contraste en el caso de buscar una duplicación, $\Lambda_1 > \Lambda_0$, sería el de signo contrario al siguiente test:

$\Lambda_1 = 0\Lambda_0 = 0$	Deleción homocigosis
$\Lambda_1 = 0{,}5\Lambda_0$	Deleción heterocigosis
$\Lambda_1 = 1{,}5\Lambda_0$	Duplicación heterocigosis
$\Lambda_1 = 2\Lambda_0$	Duplicación homocigosis

Cuadro 3.1: Cuadro con los posibles valores de Λ_1

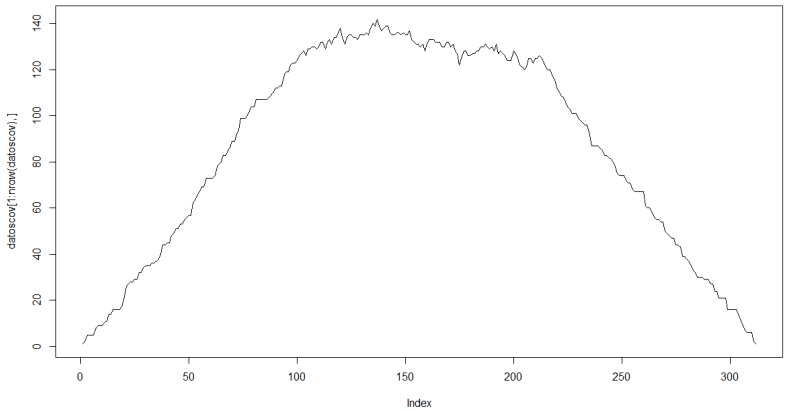

Figura 3.2: Perfil de cobertura de una región sin alteraciones obtenida de una muestra real

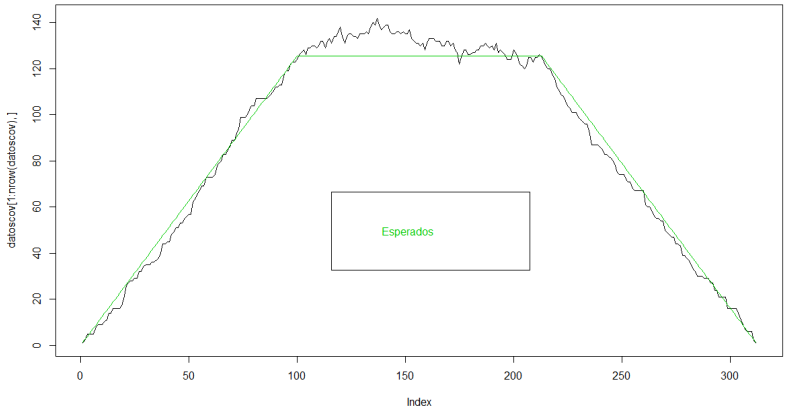

Figura 3.3: Perfil de cobertura de una región sin alteraciones obtenida de una muestra real y la curva de valores esperados bajo H_0 obtenidos para un tamaño de reads constante (color verde)

$$\phi(x) = \begin{cases} 1 \; si \; x < S \\ \gamma \; si \; x = S \\ 0 \; si \; x > S \end{cases}$$

Calculamos los valores esperados bajo la hipótesis nula H_0, mediante el proceso que describimos en la sección 3.3.2, y calculamos los p-valores de los valores de Cov_b observados en una $Poisson(\Lambda_0)$. El resultado obtenido se muestra en la Figura 3.3, en la que vemos cómo se adapta la curva esperada a la curva real, y cómo parece razonable que los valores de la curva real sigan el modelo propuesto, ya que la muestra no presenta alteraciones.

En la Figura 3.4 vemos el resultado del contraste que construimos anteriormente para $\Lambda_1 < \Lambda_0$, utilizando el lema de Neyman-Pearson, y después ajustamos los p-valores para contrastes múltiples mediante Benjamini y Hochberg (1995). Por último se compararán los valores observados de la curva real frente a los esperados, con un nivel de confianza α.

El contraste de la cobertura de las regiones se encuentra en el script *contraste.R* (página 71), en el que también se incluye un ajuste de los p-valores de Benjamini y Hochberg (1995).

3.4. Simuladores

Podemos utilizar el modelo propuesto en la sección 3.1 para construir simuladores de cobertura de regiones sin alteraciones.

Para construir el simulador, primero definimos el tamaño de la región que queremos simular, en este caso el tamaño es f, y las bases b ($b = 1, \ldots, f$); definimos R, que es la longitud de los reads, que asumimos constante. Por último definimos el λ, de dos posibles formas:

Calculamos el λ de una muestra control, como propusimos anteriormente, de la forma ($\hat{\lambda} = \frac{1}{f-R}\sum_{b=1}^{f-R} N_b$), o también podemos especificarlo a través de la altura en el centro que queremos que alcance la simulación. Por ejemplo: si queremos que nuestra curva alcance una cobertura determi-

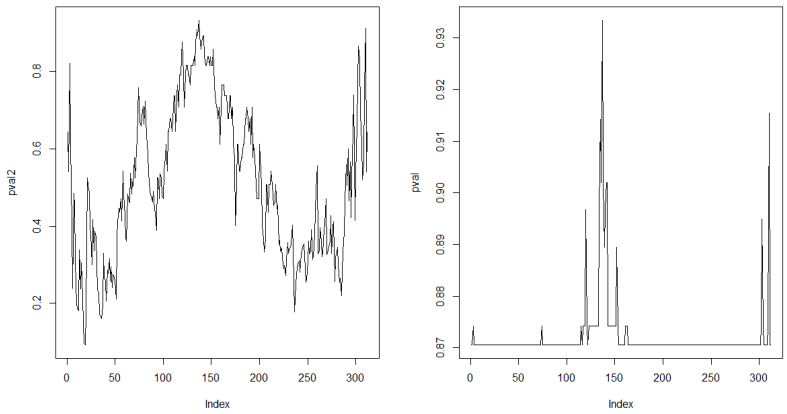

Figura 3.4: Izquierda: p-valores sin ajustar para el test de deleción. Derecha: p-valores ajustados mediante Benjamini y Hochberg (1995). Muestra real de la Figura 3.2

nada en la región central, el λ se estima como: $\lambda = \frac{Cov_{centro}}{R}$.

Con estos elementos empezamos simulando f valores de una variable de $Poisson(\lambda)$, que representa el número de reads que empiezan en cada una de nuestras bases b ($b = 1, ..., f$), que son variables independientes e idénticamente distribuidas. Después, a cada read le asignamos la longitud R.

Seguidamente construimos n vectores de R unos, donde ($b = 1, \ldots, f$) y ($N_b = 1, \ldots, N_b$), que representan a cada read que empieza en cada base b y que tienen una longitud R.

$(\vec{v}_{R_{1,1},1}, \ldots, \vec{v}_{R_{1,N_1},n-N_1}, \ldots, \vec{v}_{R_{f,1},n-N_f}, \ldots, \vec{v}_{R_{f,N_f},n})$.

Estos vectores se colocan en una matriz $M_{n\times f}$ con f columnas, tantas como bases tiene la región, y n filas, tantas como reads hemos creado. Cada vector $\vec{v}_{R_{b,N_b},k}(k = 1, ..., n)$ se coloca en su fila R_{b,N_b} correspondiente al número de read creado y que empieza en la columna b en la que empieza el read. Los reads que son más largos que la distancia de su base b a la base f son eliminados ($\left|\vec{v}_{R_{b,N_b},k}\right| > b-f+1$).

Por último, sumamos las columnas de la matriz y obtenemos de esta forma la cobertura, Cov_b, para cada base b.

Vemos un ejemplo para una región con $f = 312$ bases, un tamaño de reads $R = 101$ bases, una cobertura en la parte central de 130 y un parámetro experimental λ=1.254717. A esta simulación le aplicamos el contraste anterior, del mismo modo que lo hicimos para la muestra real. La región simulada puede verse en la Figura 3.5, donde podemos ver los valores esperados (curva verde), calculados bajo la hipótesis nula H_0, aplicando la ecuación (1); y en la Figura 3.6, donde podemos ver los p-valores del resultado del contraste tanto crudos, como ajustados por el ajuste de Benjamini y Hochberg (1995).

El contraste de la cobertura de las regiones se encuentra en el script *contraste.R*, en el que también se incluye un ajuste de los p-valores de Benjamini y Hochberg (1995); y el script utilizado para obtener esta simulación se encuentra en *Simulador.R* (página 85).

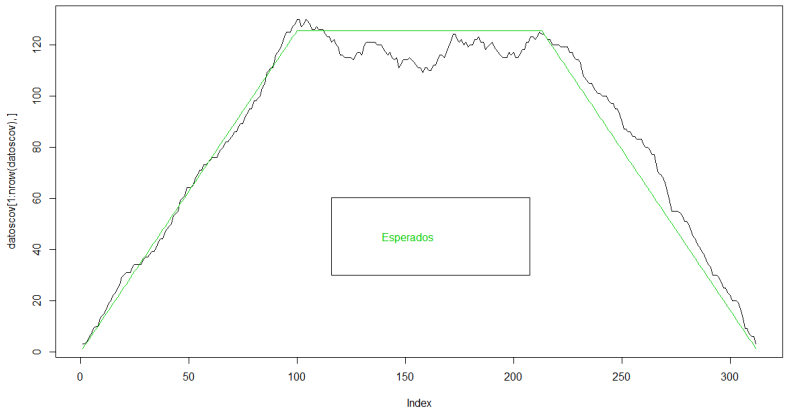

Figura 3.5: Perfil de cobertura de una región de 312 bases simulada con el script *Simulador.R*, y la curva de valores esperados bajo H_0 (color verde)

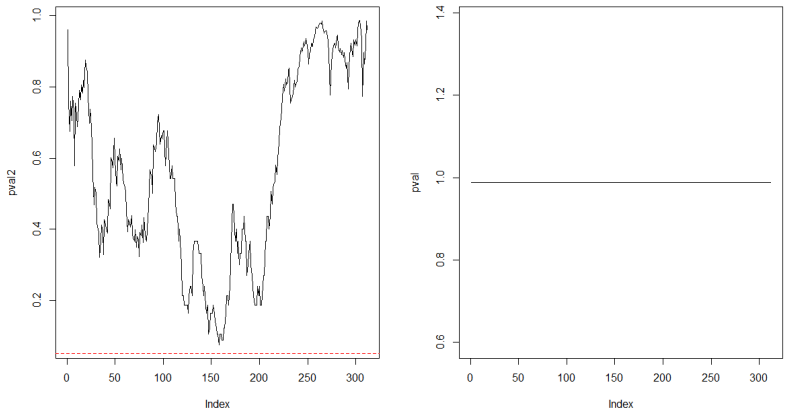

Figura 3.6: Izquierda: p-valores sin ajustar para el test de deleción. Derecha: p-valores ajustados mediante Benjamini y Hochberg (1995). Muestra simulada de la Figura 3.5

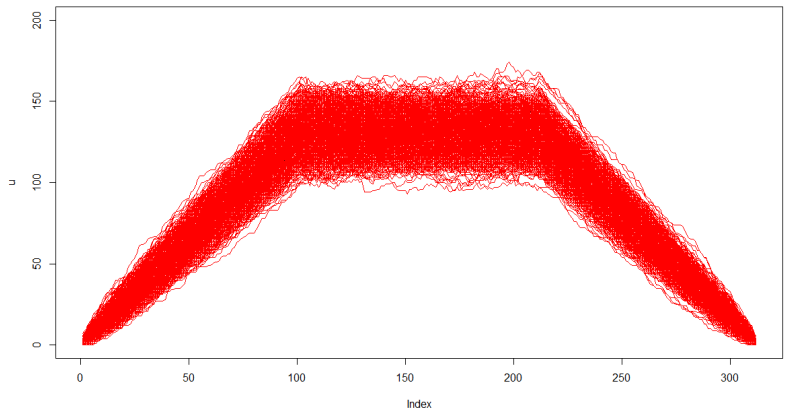

Figura 3.7: Perfil de cobertura de 1000 regiones simuladas de 312 bases

3.4.1. Simulación alternativa

Una forma alternativa para simular el proceso es replicar lo que sucede en un experimento de secuenciación, lo que se realiza de la siguiente manera: primero definimos, R, que marca el tamaño de los reads, estos son constantes, como propusimos anteriormente. Crearemos un número n de vectores de R unos, que representan a cada read, pero a diferencia de antes, estos vectores, $(\vec{v}_{R_1}, \ldots, \vec{v}_{R_n})$, aún no tienen una base de inicio asignada.

El siguiente proceso es repartir los n reads creados entre las distintas f posiciones de la región. Esto se realiza simulando n datos de una uniforme, $t = [f \times U(0,1) + 1]$, donde $[x]$ representa la parte entera de x, que indicará en qué posición se colocan los reads sobre la región objetivo. Los reads que no encajen completamente en la región no se tienen en cuenta para que se reproduzca la forma de pirámide truncada. Al final, obtenemos una matriz $M_{n\times f}$, similar a la obtenida en el proceso de la sección anterior, la diferencia radica en el orden de la simulación.

En el primer proceso, simulamos en primer lugar las posiciones de los reads y luego les asignamos una longitud, mientras que mediante esta segunda forma, primero asignamos una longitud y luego su posición. En nuestra experiencia, ambas formas de simulación obtienen el mismo resultado. El script utilizado para realizar esta simulación se encuentra en *Simulador_2.R* (página 87).

3.5. Intervalo de confianza

Con nuestro simulador operativo, podemos simular tantas curvas como deseemos y esto nos permite también construir intervalos de confianza.

En la Figura 3.7 vemos $m = 1000$ simulaciones de una región de tamaño de $f = 312$ bases, todas con un tamaño de reads, R, de 101 bases, una cobertura en la parte central de 130 y un parámetro experimental λ =1.254717.

Para obtener los extremos del intervalo, se construye una matriz $M_{f\times m}$ donde se colocan cada uno de los m perfiles de cobertura simulados de longitud f, después cada columna, que representa a una base de la región, es ordenada de menor a mayor y nos quedamos con el valor entero más próximo a $(m \times (1 - \frac{\alpha}{2}))$ para el extremo superior del intervalo y $(m \times \frac{\alpha}{2})$ para el valor inferior del intervalo,

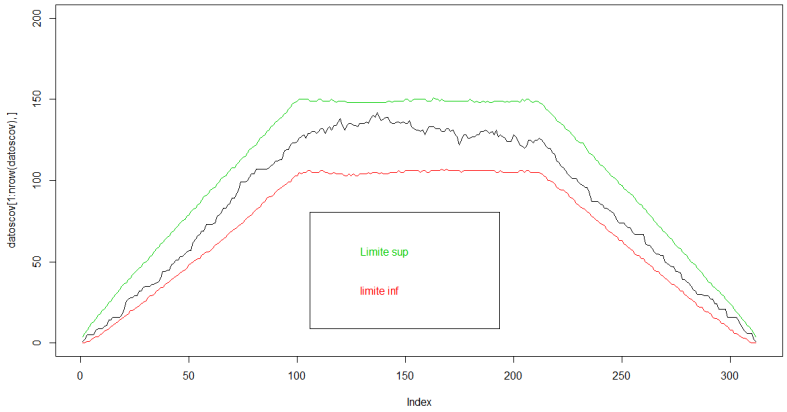

Figura 3.8: Perfil de cobertura de la región real de 312 bases (Figura 3.2) y las curvas de los límites del intervalo con un α del 0.05

siendo α el nivel de confianza prefijado.

Vemos en la Figura 3.8 y en la Figura 3.9, los intervalos confianza tanto para la región real como la simulada en la sección anterior. Los resultados obtenidos son acordes a los contrastes presentados anteriormente, englobando al perfil de cobertura observado, de modo que nos conducen a las mismas conclusiones que antes.

Se aprecia en la región simulada que un fragmento de la cola de la derecha se sale del límite de confianza superior. Esto se espera en un 5 % de las bases, bajo condiciones de independencia en las observaciones, pero la cobertura no es independiente, de modo que el número de bases que se salen del límite es mayor o menor que en condiciones de independencia. Este fenómeno lo veremos también más adelante cuando se analice la validez del contraste.

El script utilizado para realizar estos intervalos de confianza se encuentra en *IC.R* (página 82).

3.6. Simular alteraciones

Para poder calcular la potencia del contraste, previamente, implementamos en el simulador la posibilidad de simular alteraciones. Nos centramos sólo en las 4 posibilidades más simples, que son: deleción en homocigosis, deleción en heterocigosis, duplicación en homocigosis y duplicación en heterocigosis (Cuadro 3.1).

Antes de empezar con la implementación de las alteraciones en los simuladores, hay que recordar que nos situamos en un contexto en el que se ha realizado un alineamiento global y gapped de los reads (Figura 2.2). Esto es importante, ya que permitimos que el read no sea una unidad indivisible (alineamiento gapped), y un read puede encontrarse flanqueando los lados de una alteración y no formando parte de la misma. Esto genera un perfil de cobertura de la alteración muy característico, con un cambio brusco en la cobertura, efecto que no veremos si tratamos el read como una unidad indivisible.

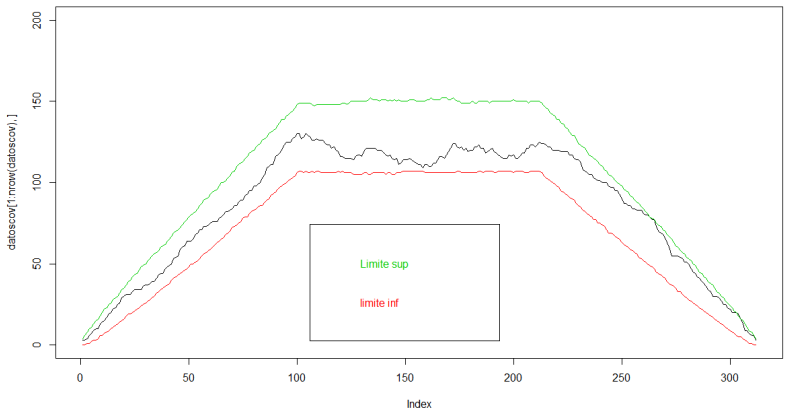

Figura 3.9: Perfil de cobertura de la región simulada de 312 bases (Figura 3.5) y las curvas de los límites del intervalo con un α del 0.05

3.6.1. Deleción

Para implementar en el simulador una deleción en homocigosis, partimos de la matriz $M_{n\times f}$ que obtuvimos en la sección de simulación y que contiene los reads colocados para calcular la cobertura de una región sin alteraciones. Primero seleccionamos la base de inicio y final entre las que queremos simular la deleción en homocigosis, recorremos las n filas de la matriz $M_{n\times f}$, seleccionando aquellas que presenten al menos un uno en el intervalo que delimita la base de inicio y final de la alteración. Después, insertamos un vector de ceros, de igual tamaño a la alteración en la base donde empieza la alteración, desplazando de esta forma los elementos de la fila n de la matriz, $M_{n\times f}$, tantas bases a la derecha como grande es la alteración. Esto nos genera una fila de mayor tamaño que la original, $f + (final\ alteración - inicio\ alteración)$. Para poder sustituir esta nueva fila en la matriz, eliminamos un número de elementos de la fila iguales al tamaño de la alteración, pero esto solo se puede hacer con las filas que no contengan ningún uno en este extremo del final; las filas que sí contengan unos son sustituidas por una fila de f elementos todos ceros, debido a que los reads que no encajen completamente en la región de f bases son eliminados y por lo tanto no aportarán información a la cobertura.

En el caso de simular una deleción en heterocigosis, no todas las filas con al menos un uno en el intervalo que delimita la base de inicio y final de la alteración serán modificadas; esperamos que solo la mitad de ellas se vean alteradas por la deleción. Para reproducir este fenómeno, para cada fila que presente al menos un uno en el intervalo que delimita la alteración arrojamos una Bernoulli con probabilidad 0.5, que nos indicará si esa fila resultará modificada o no, y repetimos el proceso.

Vemos, a continuación, unas simulaciones de los dos casos de deleción y aplicamos el contraste y el intervalo de confianza propuestos en las secciones anteriores. Primero simulamos una deleción en homocigosis ($\Lambda_1 = 0$), de la base 130 a la base 170, contrastamos la presencia de la deleción, $\Lambda_1 < \Lambda_0$, y construimos un intervalo de confianza.

Vemos en la Figura 3.10 anterior cómo la curva sigue, aproximadamente, los valores esperados de cobertura, excepto en la región central, donde se ve claramente la deleción en homocigosis.

Vemos, en la Figura 3.11 que, al realizarse el contraste, aparece una zona que se sale del límite de confianza del 95 % correspondiente a la alteración, tanto para los p-valores como para los p-valores

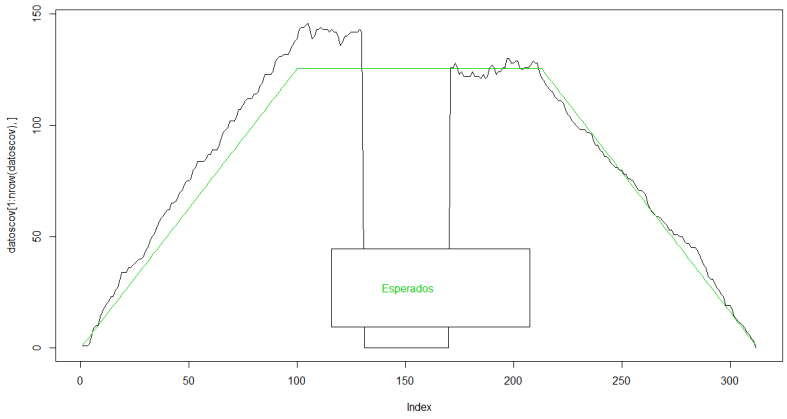

Figura 3.10: Perfil de cobertura de una región simulada de 312 bases con una deleción en homocigosis entre la base 130 y la base 170, y curva de valores esperados bajo H_0 obtenidos usando un tamaño de los reads constante (color verde)

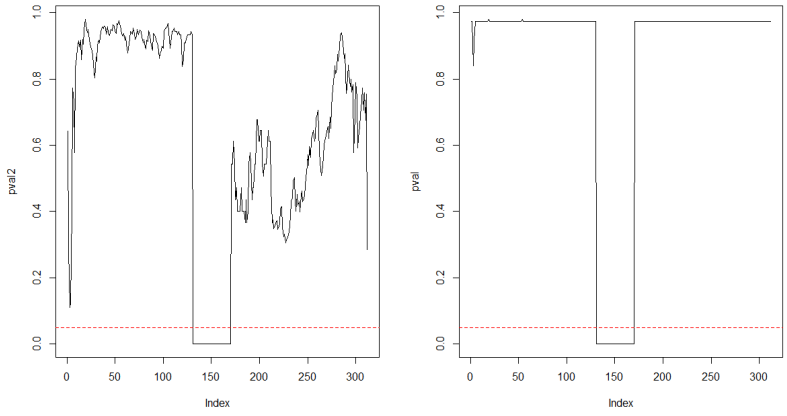

Figura 3.11: Izquierda: p-valores sin ajustar. Derecha: p-valores ajustados mediante Benjamini y Hochberg (1995). Perfil de cobertura de la Figura 3.10

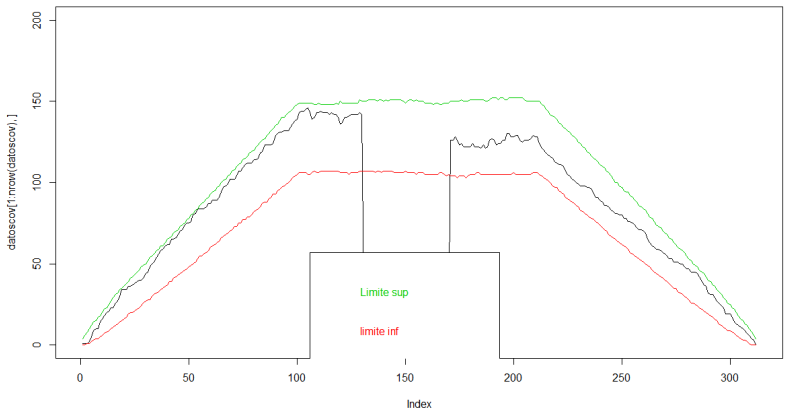

Figura 3.12: Perfil de cobertura de la región simulada de 312 bases con una deleción en homocigosis entre la base 130 y la base 170, y curvas de los límites del intervalo con un α del 0.05

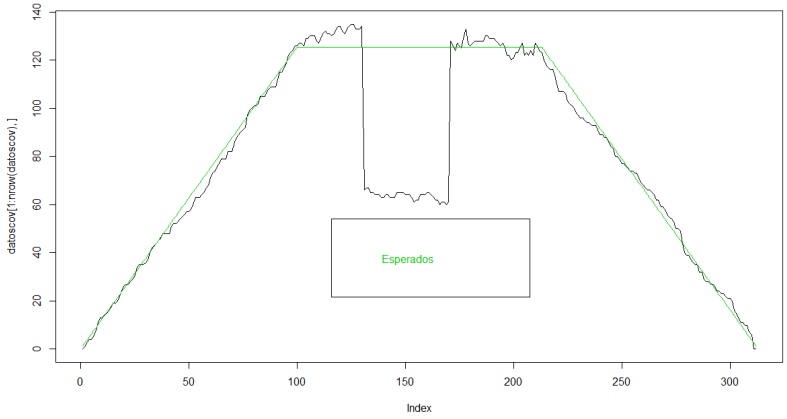

Figura 3.13: Perfil de cobertura de una región simulada de 312 bases con una deleción en heterocigosis entre la base 130 y la base 170, y curva de valores esperados bajo H_0 (color verde)

ajustados por Benjamini y Hochberg (1995), por lo que la deleción en homocigosis se detecta muy claramente.

Construimos un intervalo de confianza para esta simulación. Vemos en la Figura 3.12 anterior como la deleción se sale de los límites de confianza al 95 %.

A continuación, simulamos una deleción en heterocigosis, ($\Lambda_1 = \Lambda_0/2$), de la base 130 a la base 170, contrastamos la presencia de la deleción, $\Lambda_1 < \Lambda_0$ y construimos un intervalo de confianza. Vemos en la Figura 3.13 anterior cómo la curva sigue, aproximadamente, los valores esperados de cobertura, excepto en la región central donde se ve claramente la deleción en heterocigosis.

Vemos, en la Figura 3.14, al realizar el contraste que la deleción se sale del límite de confianza del 95 %, tanto para los p-valores como para los p-valores ajustados y vemos en el la Figura 3.15 como la deleción se sale de los límites de confianza al 95 %.

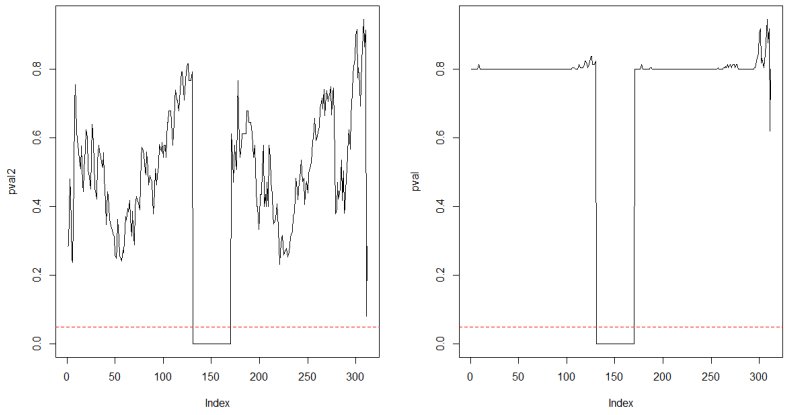

Figura 3.14: Izquierda: p-valores sin ajustar. Derecha: p-valores ajustados mediante Benjamini y Hochberg (1995). Perfil de cobertura de la Figura 3.13

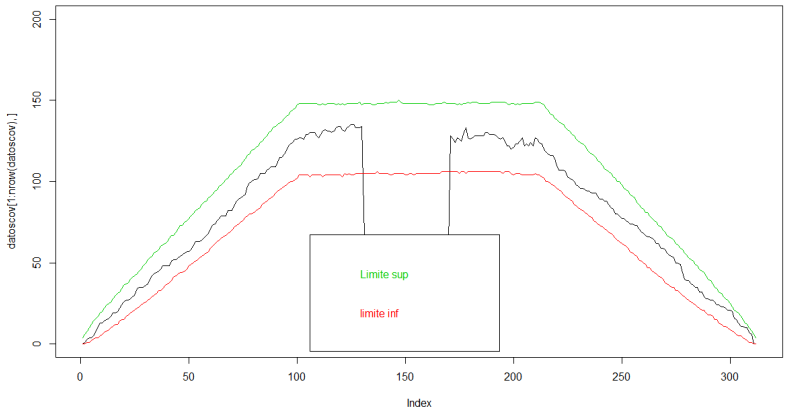

Figura 3.15: Perfil de cobertura de la región simulada de 312 bases con una deleción en heterocigosis entre la base 130 y la base 170 y curvas de los límites del intervalo con un α del 0.05

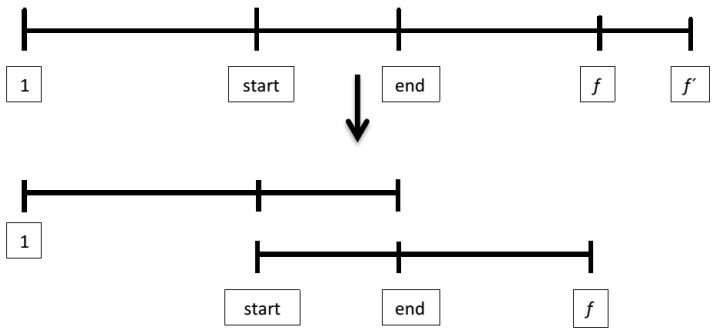

Figura 3.16: Proceso que se realiza en cada fila de la matriz obtenida en la simulación para simular una duplicación en homocigosis en una región de tamaño f

3.6.2. Duplicación

Para implementar en el simulador una duplicación en homocigosis, tratamos de replicar lo que sucede en la realidad. Realizamos una simulación de una región normal pero de una longitud mayor a la región que queremos simular, f más el tamaño de la deleción, $f+(final\ alteración-inicio\ alteración) = f'$. Simulamos de igual modo que en una región normal y obtendremos una matriz $M_{n\times f'}$ de mayor tamaño que la región que queremos simular con la alteración. Separamos cada fila, n, de la matriz $M_{n\times g}$, en un vector que abarca desde el inicio de la fila hasta la base donde termina la duplicación que queremos simular, y otro vector que abarca desde la base en la que empieza la duplicación hasta el final de la fila n. Luego, estos vectores se agrupan en una nueva matriz $M_{2n\times f}$, que tiene el doble de filas que la original y un número de columnas iguales al tamaño de la región que queremos simular, f. De este modo los reads se superponen sobre la zona en la que se localiza la duplicación en homocigosis y aumenta la cobertura en esta zona; en la Figura 3.16 se muestra el procedimiento.

Para simular una duplicación en heterocigosis, el procedimiento de la simulación se divide en dos procesos, un primero en el que simularemos una región normal de f bases sin ninguna alteración, pero con una cobertura, $\dot{\lambda}$, que es la mitad de la normal, $\dot{\lambda} = \lambda/2$, y un segundo proceso en el que simularemos una duplicación como hacíamos para el caso de la duplicación en homocigosis, pero con una cobertura $\dot{\lambda}$. Después, las filas de estos dos procesos se unirán y obtendremos una única matriz, $M_{3n\times f}$, con la duplicación en heterocigosis. Al realizar la simulación en dos pasos, estamos reproduciendo lo que sucede en una duplicación en heterocigosis, que es que una de las dos copias del organismo diploide es normal y la otra presenta una duplicación.

A continuación vemos un par de simulaciones de los dos casos de duplicación y aplicamos el contraste y el intervalo de confianza propuestos anteriormente. Primero simulamos una duplicación en homocigosis ($\Lambda_1 = 2\Lambda_0$), de la base 130 a la base 170, contrastamos la presencia de la duplicación, $\Lambda_1 > \Lambda_0$, y construimos un intervalo de confianza. Vemos en la Figura 3.17 cómo la curva sigue, aproximadamente, los valores esperados de cobertura, excepto en la región central, donde se ve claramente la duplicación en homocigosis, y en una pequeña región después de la duplicación, pero debido a la aleatoriedad; esperamos que un porcentaje de las bases se salgan del valor esperado por azar.

Vemos, en la Figura 3.18 que, al realizar el contraste, aparece una zona que sale del límite de confianza del 95 %, correspondiente a la duplicación, tanto para los p-valores ajustados como sin ajustar, y una zona después de la duplicación que también se sale del limite de confianza para los p-valores sin

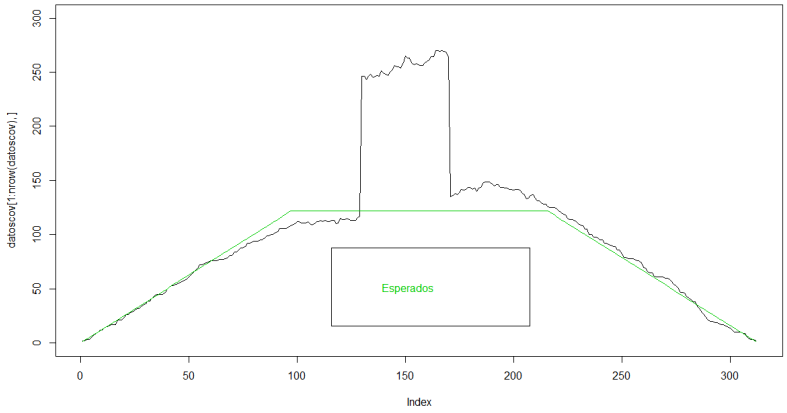

Figura 3.17: Perfil de cobertura de una región simulada de 312 bases con una duplicación en homocigosis entre la base 130 y la base 170, y curva de valores esperados bajo H_0 (color verde)

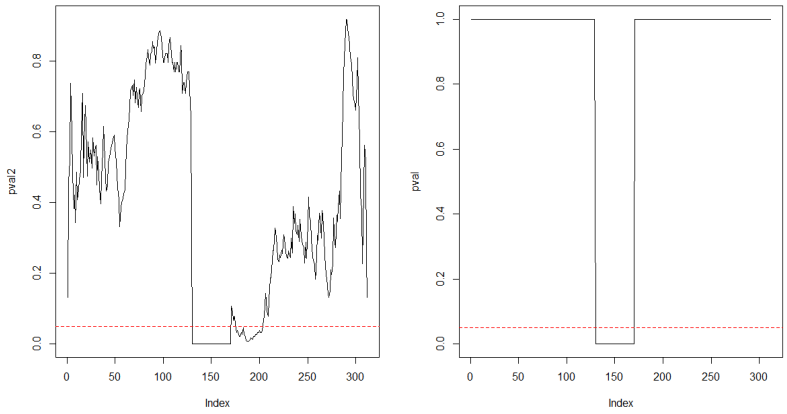

Figura 3.18: Izquierda: p-valores sin ajustar. Derecha: p-valores ajustados mediante Benjamini y Hochberg (1995). Perfil de cobertura de la Figura 3.17

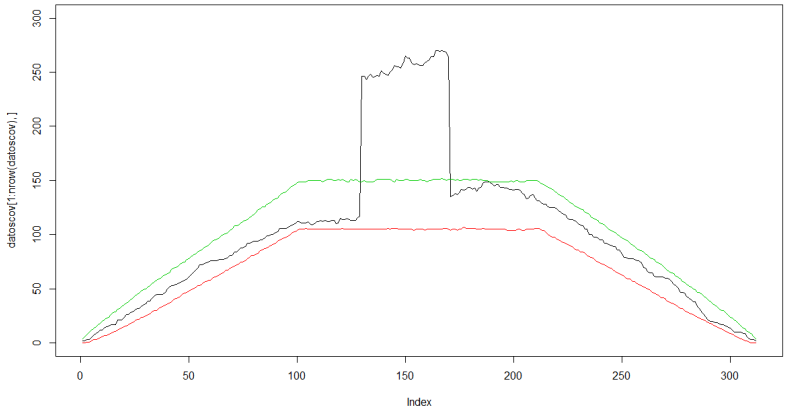

Figura 3.19: Perfil de cobertura de la región simulada de 312 bases con una duplicación en homocigosis entre la base 130 y la base 170 y curvas de los límites del intervalo con un α del 0.05

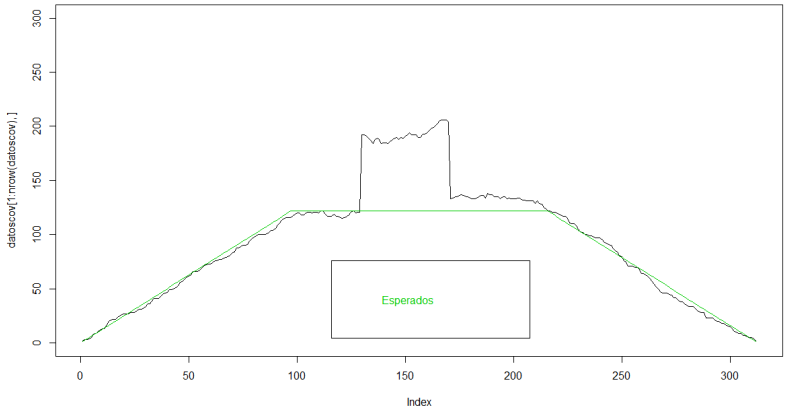

Figura 3.20: Perfil de cobertura de una región simulada de 312 bases con una duplicación en heterocigosis entre la base 130 y la base 170, y curva de valores esperados bajo H_0 (color verde)

ajustar pero no para los p-valores ajustados. La duplicación se detecta muy claramente.

Construimos un intervalo de confianza para esta simulación.

Vemos en la Figura 3.19 cómo la duplicación se sale de los límites de confianza al 95 %; sin embargo, la pequeña zona a la derecha de la duplicación está muy cerca de salirse del límite de confianza, pero permanece dentro.

A continuación, simulamos una duplicación en heterocigosis ($\Lambda_1 = 1{,}5\Lambda_0$) de la base 130 a la base 170. Contrastamos la presencia de la duplicación, $\Lambda_1 > \Lambda_0$, y construimos un intervalo de confianza. Vemos en la Figura 3.20 anterior cómo la curva sigue, aproximadamente, los valores esperados de cobertura, excepto en la región central, donde se ve claramente la duplicación en heterocigosis. Vemos en la Figura 3.21 que, al realizar el contraste, la duplicación se sale del límite de confianza del 95 %, tanto para los p-valores como para los p-valores ajustados.

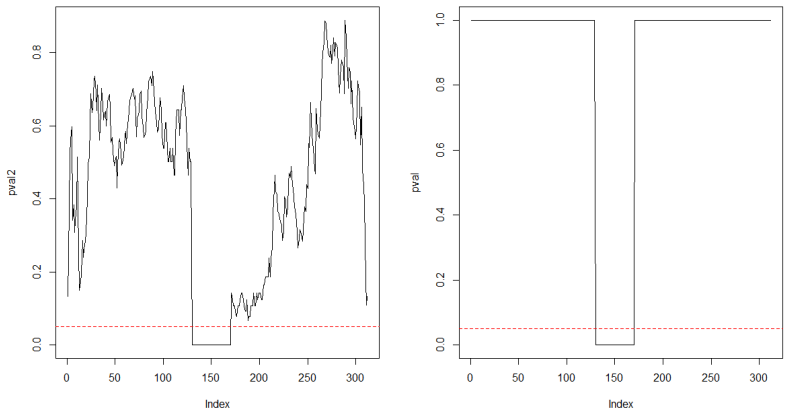

Figura 3.21: Izquierda: p-valores sin ajustar. Derecha: p-valores ajustados mediante Benjamini y Hochberg (1995). Perfil de cobertura de la Figura 3.20

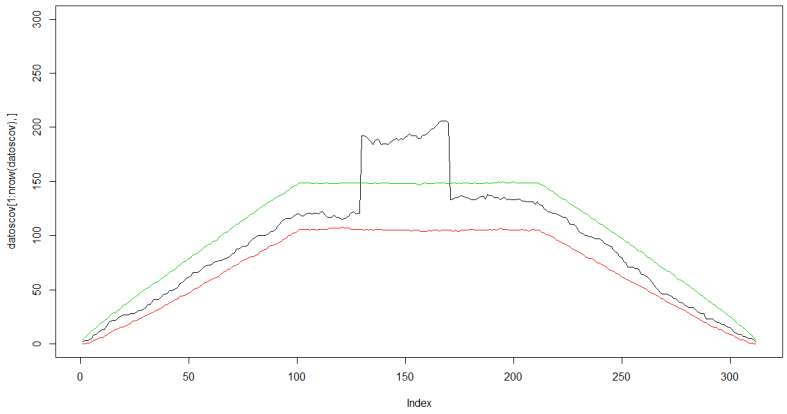

Figura 3.22: Perfil de cobertura de la región simulada de 312 bases con una duplicación en heterocigosis entre la base 130 y la base 170 y curvas de los límites del intervalo con un α del 0.05

Construimos un intervalo de confianza para la simulación. Vemos en Figura 3.22 anterior que solo la duplicación se sale de los límites de confianza al 95 %, de modo que se detecta muy bien la duplicación en heterocigosis.

El script utilizado para simular alteraciones se encuentra en *Simuldor.R* (página 85).

3.6.3. Varianza en las alteraciones y comparación con ejemplos reales

Un concepto importante a tener en cuenta cuando realizamos una simulación de alguna alteración, tanto deleción como duplicación, es que la varianza en la alteración en cuestión es distinta al del resto de la región normal. Esto se ve bien en la Figura 3.23.

Si estamos simulando una deleción en homocigosis, replicamos el efecto que produce un individuo diploide al presentar la falta de las 2 copias de una zona en concreto, y por lo tanto la cobertura en la deleción es 0 siempre, y la varianza en la deleción es 0; al no existir ningún read en la zona,

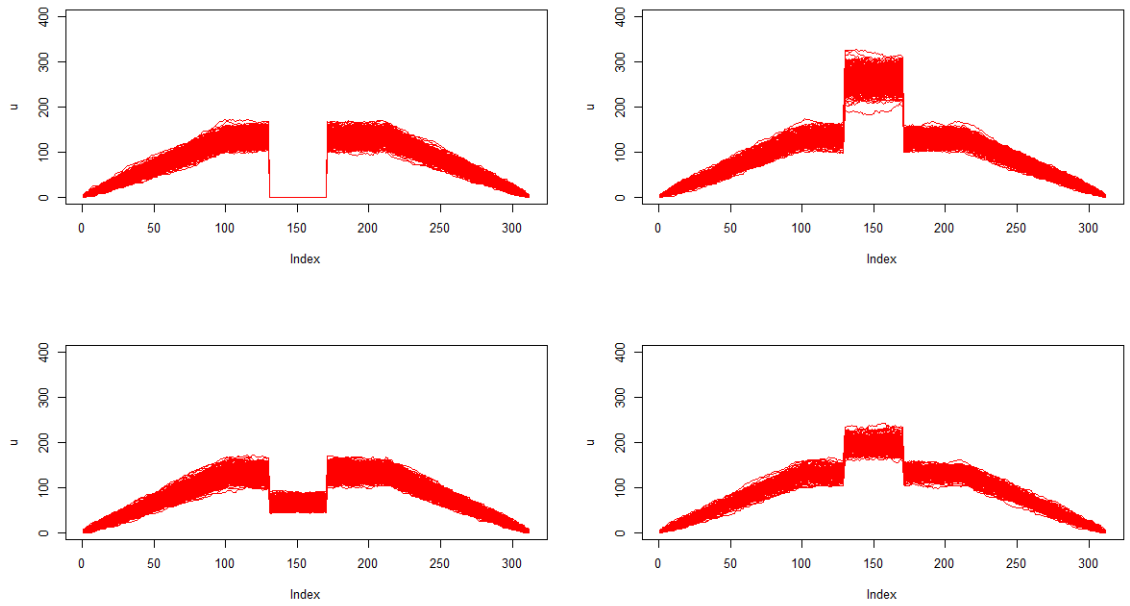

Figura 3.23: Simulaciones de 4 alteraciones, repetidas 500 veces. Arriba a la izquierda: una delecíon en homocigosis. Arriba a la derecha: una duplicación en homocigosis. Abajo a la izquierda: una deleción en heterocigosis. Abajo a la derecha: una duplicación en heterocigosis. En esta figura se aprecia cómo varía la varianza según el tipo de alteración que se simula

lo que se traduce en una Poisson de parámetro 0, mientras que en el resto de la región la varianza es normal y la marca el número de eventos de Poisson que se suman, siendo una varianza creciente desde las colas al centro de la región, donde se alcanza la mayor varianza de la región. Si simulamos una deleción en heterocigosis, la cobertura cae un 50 % en la deleción, al estar simulando un individuo con una sola copia, lo que produce que la varianza también sea menor, ya que sí hay reads en la deleción, pero son la mitad de lo esperado; y el mismo efecto se produce en las duplicaciones, en las que la varianza es mayor que en la zona normal. De modo que esperamos que el método propuesto para simular las alteraciones reproduzca este fenómeno en la varianza, como vemos en la Figura 3.23.

Por último podemos ver como los perfiles de experimentos reales toman la misma forma que las simulaciones llevadas a cabo. En la Figura 3.24, Figura 3.25 y Figura 3.26 vemos cómo cuando aparece una alteración, y se ha utilizado un alineamiento que permita gaps en la colocación de los reads (gapped alignment), se genera un cambio brusco en la cobertura en la zona en la que se encuentra la alteración.

El script utilizado para las simulaciones de esta sección se encuentra en *Simuldor.R* (página 85).

3.7. Validez del contraste

Necesitamos validar el contraste que hemos construido. Primero procedemos a calcular el Error tipo I. Para ello realizamos una simulación intensiva de curvas sin alteraciones, apoyándonos en el simulador que presentamos en este capítulo. Para estudiar el Error tipo I calculamos el porcentaje de veces que el contraste da falsos positivos para cada base de la región; lo que es lo mismo, cuántas veces el p-valor obtenido por el contraste rebasa el nivel del 5 % que hemos prefijado. Por la forma de

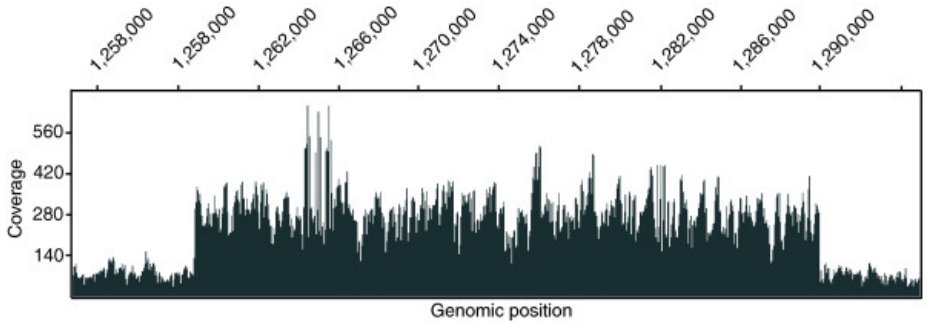

Figura 3.24: Cobertura de una duplicación larga. (Imagen extraída de: Conrad et al., 2009)

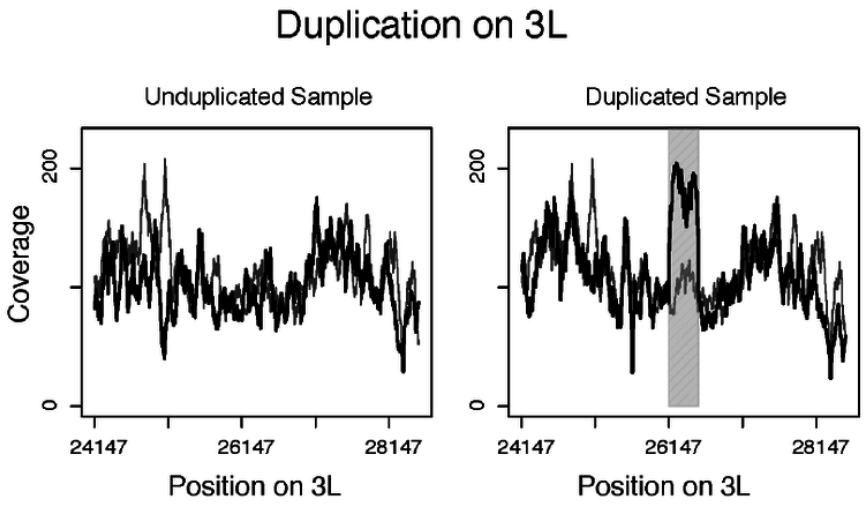

Figura 3.25: Cobertura de una duplicación. (Imagen extraída de: Rebekah et al., 2014)

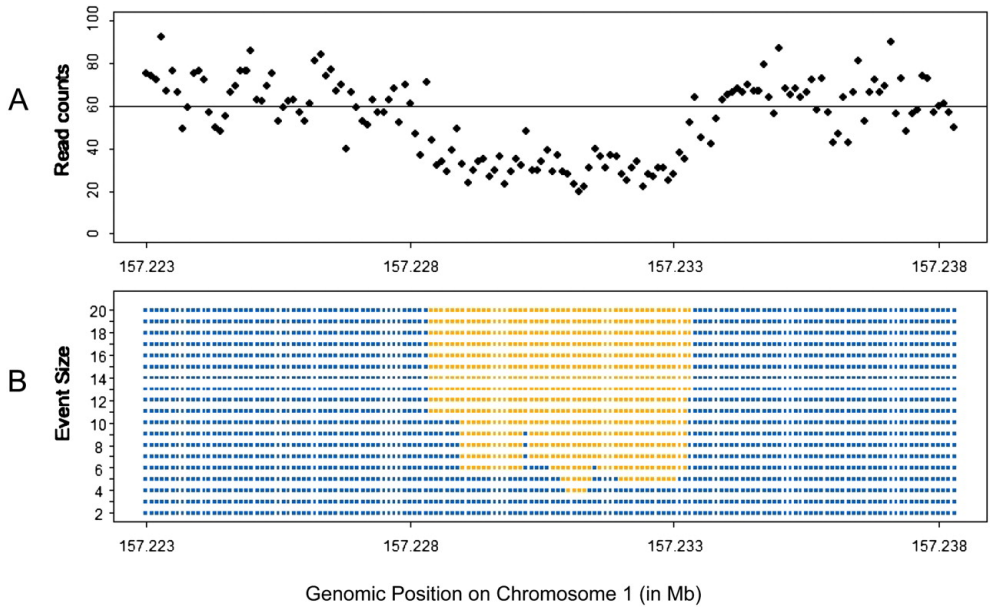

Figura 3.26: Cobertura de una deleción. (Imagen extraída de: Yoon et al., 2009)

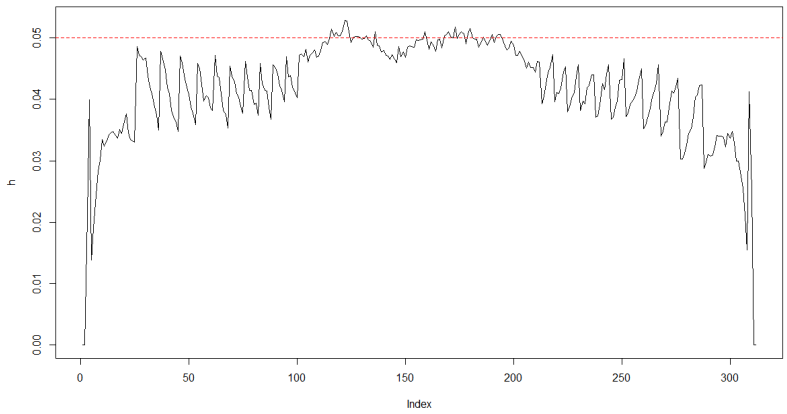

Figura 3.27: Porcentaje de veces que cada base sobrepasa el límite de confianza del 5 % del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases sin alteraciones. En rojo vemos marcado el nivel del 5 %. Se aprecia que el Error Tipo I no es el mismo para todas las bases

sumatorio creciente de variables de Poisson de los datos de cobertura, Cov_b, sabemos que la varianza de las bases en el centro de la región es mayor que en las colas; esto afectará al Error Tipo I, que va a variar desde la región central a las cosas, reduciéndose.

Para estudiar el comprometimiento del Error Tipo I, necesitamos realizar una simulación suficientemente grande. Nosotros hemos recurrido a simular 10000 curvas como un compromiso entre un gran número de simulaciones y tiempo el tiempo computacional. Esta simulación ha sido llevada acabo con unos parámetros de tamaño de los reads $R = 101$, una región de $f = 312$ bases y un parámetro experimental λ =1.254717. El resultado de este estudio de simulación puede verse en la Figura 3.27.

Por la Figura 3.27 vemos que la probabilidad de cometer Error Tipo I varia en función de la zona de la región en la que nos encontremos, llegando a 0 % en la primera y última base y sólo el límite del 5 % en las bases más centrales. Esto nos indica que el contraste es bastante conservador para las base localizadas en las colas y que mantiene controlado el Error Tipo I, siendo la aproximación al 5 % nominal buena, sobre todo en la zona central.

Para estudiar la potencia nos limitamos a los 4 casos de alteración: deleción en homocigosis, deleción en heterocigosis, duplicación en homocigosis y duplicación en heterocigosis (Cuadro 3.1). Para medir la potencia en cada uno de los casos de alteración y para cada una de las bases, simulamos cada una de las alteraciones que afectan a la totalidad de la región de 312 bases y calculamos el porcentaje de veces que cada base es detectada por el contraste, y vemos el efecto en la potencia de la posición de la base en la región.

Primero analizamos el caso de una deleción en homocigosis (en la deleción en homocigosis la cobertura de todas las bases afectadas será de 0). Simulamos una deleción en homocigosis que abarque toda la región para, de este modo, ver la potencia del contraste según la zona de la región en la que se encuentra la deleción. El resultado de una simulación de 10000 regiones lo vemos en la Figura 3.28.

En la Figura 3.28 vemos que la potencia del test es del 100 % para todas las bases excepto para las dos primeras bases de la región. En la Figura 3.29 vemos que si ajustamos los p-valores por Benjamini y Hochberg (1995), la potencia del test no se ve afectada para este caso de alteración.

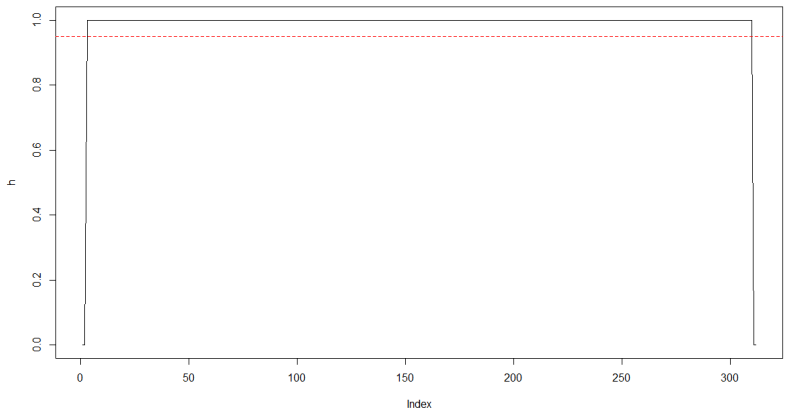

Figura 3.28: Potencia del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases con una deleción en homocigosis que abarca la totalidad de la región. En rojo vemos marcada una potencia del 95 %

Para el caso de una deleción en heterocigosis (en la deleción en heterocigosis la cobertura de todas las bases afectadas será de la mitad de la cobertura normal). Simulamos una deleción en heterocigosis que abarque la totalidad de la región, igual que en el caso anterior, y de este modo ver la potencia del contraste según la zona de la región en la que se encuentre la deleción. El resultado de una simulación de 10000 regiones lo vemos en la Figura 3.30.

En la Figura 3.30 vemos que la potencia del test crece desde el 0 % hasta el 100 % a mucha velocidad, en aproximadamente las 50 primeras bases, alcanzado una potencia del 95 % en las primeras 25 bases. La potencia del test cae a la misma velocidad en las bases de la derecha de la región, como cabe esperar. En la Figura 3.31 vemos que, si ajustamos los p-valores por Benjamini y Hochberg (1995), la potencia del test sólo se ve afectada en las bases del extremo de la región, que pasan a tener una potencia de 0 %.

A continuación vemos la potencia del contraste para el caso de una duplicación en homocigosis, (en la duplicación en homocigosis la cobertura de todas las bases afectadas será del doble de la cobertura normal). Simulamos una duplicación en homocigosis que abarque toda la región, igual que en los casos de lateraciones anteriores. El resultado de una simulación de 10000 regiones lo vemos en la Figura 3.32.

En la Figura 3.32 vemos que la potencia del test crece desde el 0 % hasta el 100 % a más velocidad que en el caso de la deleción en heterocigosis. En tan solo las 25 primeras bases, el test alcanza una potencia del 100 %. La potencia del test cae a la misma velocidad en las bases de la derecha de la región, por la simetría de la región. En la Figura 3.33 vemos que el comportamiento de la potencia si ajustamos los p-valores por Benjamini y Hochberg (1995) no se ve prácticamente afectada.

Por último vemos el comportamiento de la potencia en el caso de una duplicación en heterocigosis (en la duplicación en heterocigosis la cobertura de todas las bases afectadas será de 1.5 veces la de la cobertura normal). Simulamos una duplicación en heterocigosis que abarque toda la región, igual que en los casos anteriores y vemos el resultado de una simulación de 10000 regiones en la Figura 3.34.

En la Figura 3.34 vemos que la potencia del test crece desde el 0 % hasta el 100 % mucho más lento que en los casos anteriores, el test alcanza la potencia del 100 % en las 75 primeras bases, y la potencia

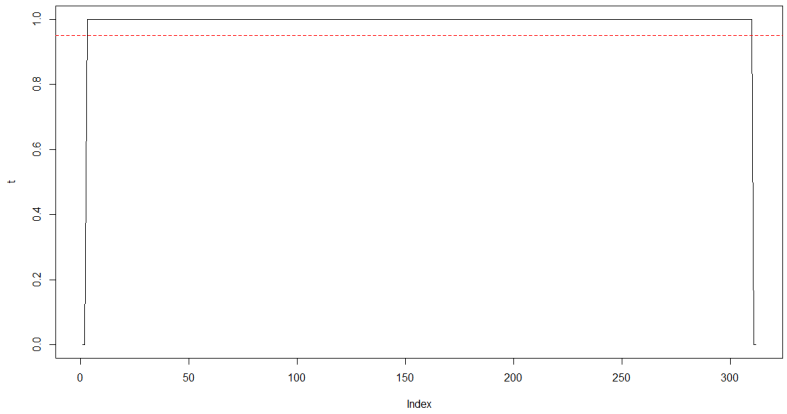

Figura 3.29: Potencia del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases con una deleción en homocigosis que abarca la totalidad de la región, con los p-valores ajustados por Benjamini y Hochberg (1995). En rojo vemos marcada una potencia del 95 %

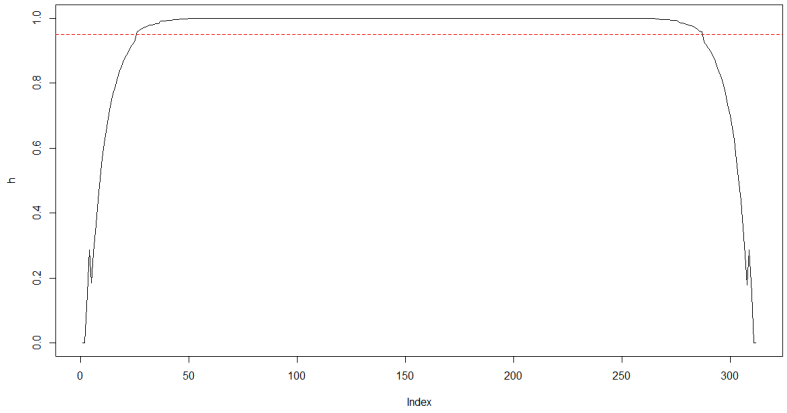

Figura 3.30: Potencia del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases con una deleción en heterocigosis que abarca la totalidad de la región. En rojo vemos marcada una potencia del 95 %

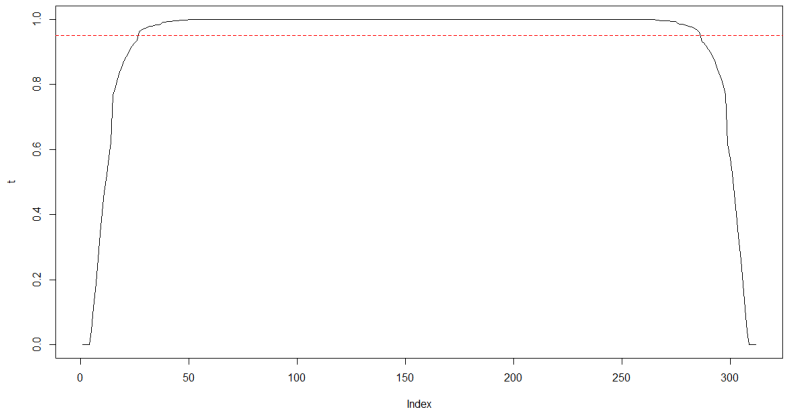

Figura 3.31: Potencia del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases con una deleción en heterocigosis que abarca la totalidad de la región, con los p-valores ajustados por Benjamini & Hochberg (1995). En rojo vemos marcada una potencia del 95 %

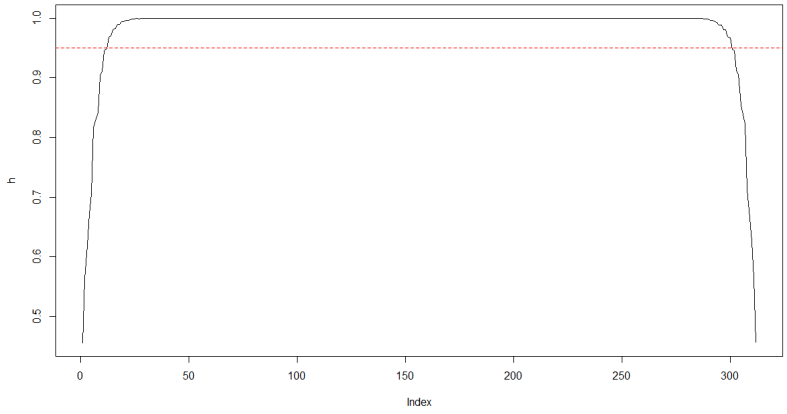

Figura 3.32: Potencia del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases con una duplicación en homocigosis que abarca la totalidad de la región. En rojo vemos marcada una potencia del 95 %

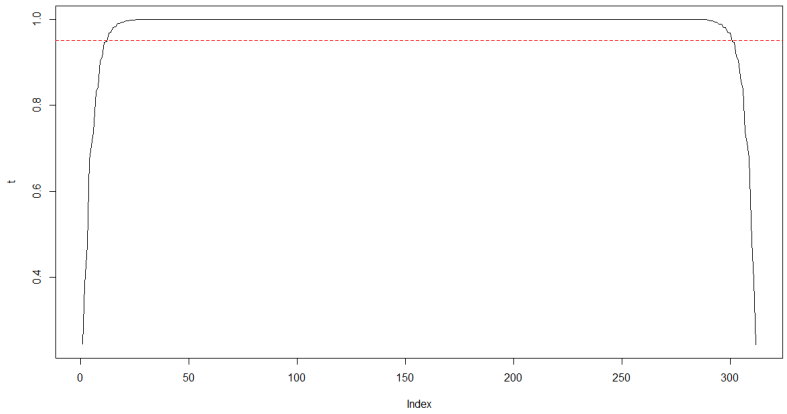

Figura 3.33: Potencia del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases con una duplicación en homocigosis que abarca la totalidad de la región, con los p-valores ajustados por Benjamini y Hochberg (1995). En rojo vemos marcada una potencia del 95 %

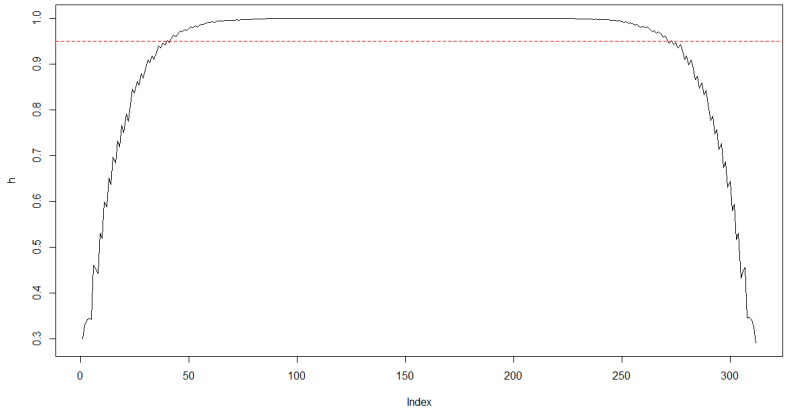

Figura 3.34: Potencia del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases con una duplicación en heterocigosis que abarca la totalidad de la región. En rojo vemos marcada una potencia del 95 %

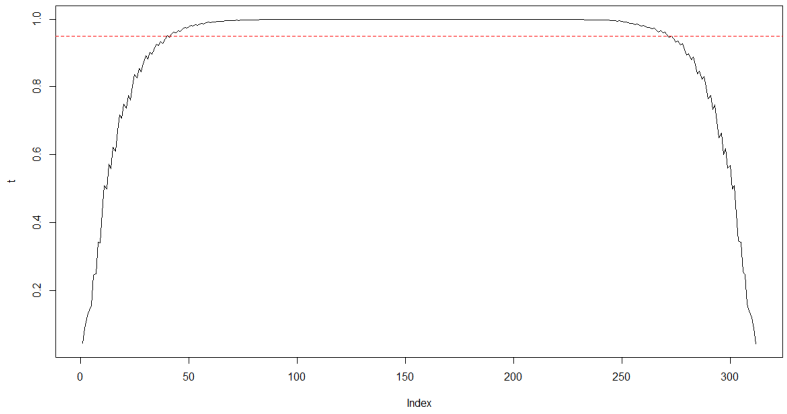

Figura 3.35: Potencia del contraste para una simulación intensiva de 10000 regiones de tamaño de 312 bases con una duplicación en heterocigosis que abarca la totalidad de la región, con los p-valores ajustados por Benjamini y Hochberg (1995). En rojo vemos marcada una potencia del 95 %

del test cae a la misma velocidad en las bases de la derecha de la región. En la Figura 3.35 vemos que el ajuste de los p-valores por Benjamini y Hochberg (1995) afecta ligeramente a las 75 primeras y últimas bases reduciendo la potencia del test en las mismas.

Con estas simulaciones, podemos comprobar que el contraste es valido y funciona bien, incluso con el ajuste de Benjamini y Hochberg (1995), que no daña significativamente la potencia del test. Hay que tener en cuenta que tanto la Potencia la potencia como el Error Tipo I crecen conforme nos acercamos de la posición central de la región, y este efecto se nota sobre todo entre las bases $75 - 50$, tanto la principio como al final del a región. Este fenómeno es debido a que la cobertura de cada base b de la región, Cov_b, se comporta como una suma de Poissons de igual parámetro, y este efecto aditivo crece desde los extremos a la zona central de la región. De modo que cuando buscamos alteraciones en los extremos de las regiones tendremos un contraste con poca potencia, y por lo tanto poca capacidad para detectar alteraciones. El script necesario para replicar estar pruebas se encuentra en *errores_contraste.R* (página 79).

3.8. Estadístico resumen

El problema de identificar una alteración en una región problema se puede enfocar desde una perspectiva global en lugar de analizar la región a nivel de base; sí nos centramos en una región en concreto, podemos construir un estadístico que resuma la información de la cobertura observada en la región de interés y que compare estos valores con los esperados bajo condiciones de no alteración, considerando

la longitud de los reads, R, constante. Esperamos que si sumamos la cobertura Cov_b, $\sum_{b=1}^{f} Cov_b$, para las f bases de la región y la comparamos con la suma de las f cobertura esperadas bajo condiciones de no alteración podamos identificar cuando la región problema presente alguna alteración.

De modo que nuestro estadístico es de la forma:

$$T = \frac{\sum_{b=1}^{f} Cov_b \ observada}{\sum_{b=1}^{f} Cov_b \ esperada}$$

El estadístico resumen T, tal como se ha definido, es el cociente entre la suma de coberturas observadas para las bases $1, \ldots, f$ y la correspondiente cantidad esperada bajo la hipótesis nula. Por lo que respecta al numerador, éste coincide con la suma de las cantidades N_j (variables de *Poisson* independientes de parámetro λ), tantas veces como contribuciones tengan a las coberturas de las distintas bases. Ocurre que cada una de estas variables N_j aparece repetida R veces en la suma, si $1 \leq j \leq f-R+1$, y cero veces en otro caso. Por tanto, y en virtud de la reproductividad del modelo de *Poisson*, el numerador del estadístico resumen es R veces una *Poisson* de parámetro $\lambda \times (f - R + 1)$. Por lo que respecta al denominador, éste es precisamente R veces el parámetro de la citada distribución de *Poisson*. Por tanto, el estadístico resumen es una *Poisson* re-escalada:

$$T = \frac{Poisson(\lambda \times (f-R+1))}{(\lambda \times (f-R+1))}$$

En condiciones habituales (parámetro característico, $\lambda \times (f - R + 1)$, grande), T se aproxima bien por un modelo gaussiano, pero, también, podemos aproximar su distribución por Montecarlo. Para aproximar la distribución del estadístico bajo condiciones de no alteración por Montecarlo, necesitamos realizar una simulación con el suficiente número de regiones, calcular los valores esperados bajo condiciones de no alteración y ver cómo se comporta el estadístico T.

Para probarlo realizamos una simulación de 5000 regiones normales de tamaño $f = 312$, tamaño de los reads de $R = 101$ y un parámetro experimental λ =1.254717; para cada simulación hacemos el ratio entre la suma de las coberturas de la simulación y la suma de las coberturas esperadas. La suma de 312 elementos es lo suficientemente grande para comprobar a qué forma converge este estadístico. En la Figura 3.36 vemos el histograma del estadístico T obtenido por Montecarlo, vemos que obtenemos una campana centrada en el 1, efecto esperado bajo condiciones de no alteración. Si pasamos un test de Shapiro-Wilk de normalidad obtenemos un valor de $W = 0.99944$, y un p-valor de 0.1385, de modo que el estadístico T pasa un contraste de normalidad; como cometamos más arriba, podríamos utilizar la aproximación normal a este estadístico. Pero también podemos construir un intervalo por el método de Montecarlo, con las 5000 regiones normales de $f = 312$ bases simuladas obtenemos 5000 replicas del estadístico T, a este vector lo llamamos $\vec{T^*}$, después ordenamos $\vec{T^*}$ y nos quedamos con los extremos; inferior $B \times \frac{\alpha}{2}$ y superior $B \times \frac{1-\alpha}{2}$ con el nivel α que prefijemos y siendo $B = número\ de\ realizaciones\ de\ T^*$. De este modo obtenemos un intervalo de confianza por Montecarlo.

Para nuestro ejemplo de simulación obtenemos un intervalo para T, (0.8796992 , 1.1240601). Para una muestra de una región que obtengamos calcularemos T y veremos si se encuentra dentro de este intervalo.

En este punto, tenemos un estadístico T y un intervalo en el que se encuentra bajo condiciones de normalidad, pero si analizamos un poco en profundidad el estadístico T, vemos que en su forma se introduce la información de todas las bases de la región. Cada base presenta una varianza marcada por la posición que ocupa en la región, ya que su valor de cobertura viene dado por acumulación de variables de Poisson, su varianza crece desde los extremos a la parte central de la región; otro aspecto es que al incluir toda la información de un número de bases suficientemente grande el efecto de una

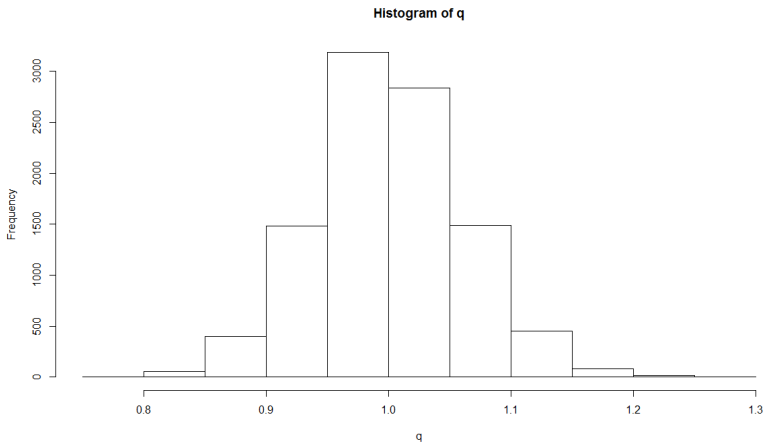

Figura 3.36: Histograma del estadístico T para 5000 simulaciones con $f = 312$, $R = 101$ y λ =1.254717

alteración que afecte a una única base puede verse diluido, de modo que necesitamos conocer como se comporta el tamaño de cada uno de los tipos de alteraciones que podemos detectar con este estadístico, y además ver como afecta a la capacidad de detección la posición de la alteración.

Para estudiar el comportamiento del estadístico T a la hora de detectar alteraciones, simulamos regiones alteradas en las que variamos progresivamente el tamaño de la alteración y realizamos este procedimiento para cada uno de los 4 tipos de alteraciones. También variaremos el lugar donde se encuentra la alteración, para poder evaluar la capacidad de detección del estadístico T cuando la alteración se encuentra en el centro de la región, como cuando se encuentra en una en los extremos.

Primero simulamos una deleción en homocigosis en el centro de una región de tamaño $f = 312$ bases, tamaño de los reads de $R = 101$ y un parámetro experimental λ =1.254717. El tamaño mínimo de la deleción en homocigosis en el centro de la región que somos capaces de detectar es de 20 bases, con un valor de T =0.8751954, de modo que para una región de tamaño $f = 312$ el limite de detección de una deleción en homocigosis que se encuentre en la región central es de 20 bases. Un detalle importante es que en una deleción en homocigosis la cobertura en la alteración siempre vale 0, efecto que no sucede en las otras tipologias de alteración y que hace más fácil detectar deleciónes en homocigosis.

Repetimos el proceso con los mismos parámetros de simulación para el caso de una deleción en heterocigosis, y calculamos cual es el porcentaje de detección al variar la longitud de la alteración. En las dos primeras filas del Cuadro 3.2 vemos el resultado de la deleción en homocigosis y heterocigosis.

Repetimos el procedimiento, con los mismos parámetros de simulación, para los casos de duplicación, tanto en homocigosis como en heterocigosis, y el resultado para las 4 posibles tipologias de alteración se muestran en el Cuadro 3.2.

Por los resultados del Cuadro 3.2, vemos lo que intuitivamente ya esperábamos, esto es, que según el tipo de alteración que estemos buscando, necesitamos que esta sea de mayor o menor longitud, siendo los casos de alteración en homocigosis los que requieren tamaños menores para ser detectados.

Repetimos el proceso anterior, con los mismos parámetros de simulación, pero esta vez colocando las alteraciones en una de las colas, de modo que podamos ver si al estar la alteración localizada en un extremo de la región, donde la cobertura es ascendente o descendente, afecta al tamaño de la al-

teración que podemos llegar a detectar con el estadístico T. Los resultados se muestran en el Cuadro 3.3.

Vemos al comparar el Cuadro 3.3 en el que se encuentran las alteraciones en un extremo de la región y el Cuadro 3.2 en el que la alteración se encuentra en el centro de la región, que la localización de la alteración afecta a la capacidad de detección, siendo más difícil detectar alteraciones en las colas, y que este efecto es mayor en las alteraciones del tipo deleción que en las del tipo duplicación, que permanecen prácticamente iguales. En estos ejemplos hemos simulado una región de 312; bases, si el tamaño de la región aumenta, también aumentará el tamaño necesario para detectar la alteración, y del mismo modo también afectarán factores como la longitud de los reads, R, ya que afectan a la forma del perfil de cobertura afectando a cuán pesadas son las colas de la región. También es importante la cobertura general del experimento ya que, conforme más bajos sean los valores de cobertura, más difícil será detectar las anomalías con el estadístico T.

Los scrips necesarios para replicar estos datos se encuentran en el archivo *contraste_global.R* (página 74).

Tamaño alteración	20	40	60	80	100	120	140
Deleción homocigosis	1	1	1	1	1	1	1
Deleción heterocigosis	0.089	0.401	0.961	1	1	1	1
Duplicación homocigosis	0.335	0.859	0.991	1	1	1	1
Duplicación heterocigosis	0.113	0.349	0.605	0.813	0.952	0.992	1

Cuadro 3.2: Valores de detección, en tanto por uno, al variar el tamaño de la alteración en los casos de una duplicación y deleción en heterocigosis y homocigosis, al localizar la alteración en el centro de la región

Tamaño alteración	20	40	60	80	100	120	140
Deleción homocigosis	0.032	0.058	0.265	0.703	0.985	1	1
Deleción heterocigosis	0.029	0.042	0.083	0.181	0.458	0.967	1
Duplicación homocigosis	0.316	0.827	0.994	1	1	1	1
Duplicación heterocigosis	0.113	0.329	0.601	0.834	0.963	0.993	1

Cuadro 3.3: Valores de detección, en tanto por uno, al variar el tamaño de la alteración en los casos de una duplicación y deleción en heterocigosis y homocigosis, al localizar la alteración en un extremo de la región

Capítulo 4

Modelo con R aleatorio

4.1. Modelo: R aleatorio

En este capítulo reconstruimos el modelo anterior permitiendo, ahora, que el tamaño de los reads, R, no sea constante, sino que sea una variable aleatoria, con función de distribución $P(R \leq r)$, que viene marcada por el experimento. El soporte de R es $\{r_{min}, \ldots, r_{max}\}$, donde $r_{min} = 1$ y $r_{max} = f$. Entonces, para cada base j ($j = 1, \ldots, f$) tenemos N_j, el número de reads que empiezan en la base j, y $R_{j,i}$ ($i = 1, \ldots, N_j$), la longitud de cada uno de los i reads de la base j ($1 \leq j \leq f$, $1 \leq i \leq N_j$).

Ahora permitimos que los reads $R_{j,i}$ tengan un tamaño aleatorio comprendido entre $r_{min} = 1$ y $r_{max} = f$; esta situación no es habitual, pero nos permite introducir toda la casuística en el modelo. Además, en esta situación, al contrario que en el caso de R constante, cualquier base j puede contribuir a la cobertura todas las b bases. De este modo tenemos que tener en cuenta en el sumatorio de Cov_b, todas las bases, y no sólo las que nos marque el valor constante de R asumido en el Capítulo 3.

Para calcular la cobertura de una base b en particular introducimos una variable binaria, $N_{b,j,i}$, que indica si cada read de longitud $R_{j,i}$, de entre los N_j reads que empiezan en la base j, cubre la base b en cuestión:

$$N_{b,j,i} = \begin{cases} 1 & si\ j + R_{j,i,r} - 1 \leq f\ y\ j + R_{j,i,r} - 1 \geq b \geq j \\ 0 & en\ otro\ caso \end{cases}$$

Calculamos entonces la cobertura de la base b, Cov_b, como sigue:

$$Cov_b = \sum_{j=1}^{f} \sum_{i=1}^{N_j} N_{b,j,i} = \sum_{j=1}^{f} \sum_{i=1}^{N_j} \mathbf{1}(j + R_{j,i,r} - 1 \leq f,\ j + R_{j,i,r} - 1 \geq b \geq j)$$

Muchos de los elementos que sumamos son nulos, así que reescribimos el sumatorio para contar sólo las bases que tienen probabilidad positiva de aportar información a la cobertura de la base b, Cov_b:

$$Cov_b = \sum_{j=max(b-r_{max}+1,1)}^{min(b,f-r_{min}+1)} \sum_{i=1}^{N_j} N_{b,j,i} =$$

$$= \sum_{j=max(b-r_{max}+1,1)}^{min(b,f-r_{min}+1)} \sum_{i=1}^{N_j} \mathbf{1}(j + R_{j,i,r} - 1 \leq f,\ j + R_{j,i,r} - 1 \geq b \geq j)$$

A diferencia del contexto con R constante, permitimos que el tamaño de los reads R varíe. Por tanto, los elementos $N_{b,j,i}$ del sumatorio $\sum_{i=1}^{N_j} N_{b,j,i}$, siguen una ley de probabilidad; que $N_{b,j,i}$ valga 1 ó 0, viene marcado por la probabilidad $p_{b,j}$, que es la probabilidad de obtener un read suficientemente

largo para llegar desde la base j a la base b y no tan largo como para salirse de la región de tamaño f, $p_{b,j} = P(f-j+1 \geq R_{j,i} \geq b-j)$. Esto significa que $N_{b,j,i} \sim Ber(p_j)$ y, además, todos los $N_{b,j,i}$ tienen el mismo $p_{b,j}$. Por lo tanto, $N_{b,j} = \sum_{i=1}^{N_j} N_{b,j,i}$ es la suma de N_j variables Bernoulli independientes e idénticamente distribuidas.

Así pues, condicionadamete a N_j, la variable $N_{b,j} \sim Binomial(N_j, p_j)$ donde ponemos $p_j = p_{b,j}$ por simplicidad notacional. Teniendo en cuenta que $N_j \sim Poisson(\lambda)$, la función de masa de probabilidad incondicional de $N_{b,j}$ es:

$P(N_{b,j} = x) = \sum_{n_j=x}^{\infty} \binom{n_j}{x} p_j^x (1-p_j)^{n_j-x} \ \frac{e^{-\lambda}\lambda^{n_j}}{n_j!};\ x = 0, 1, 2, \ldots$

Haciendo un cambio de variable, $n_j = \dot{n} + x$ y simplificando términos:

$P(N_{b,j} = x) = \sum_{\dot{n}=0}^{\infty} \binom{\dot{n}+x}{x} p_j^x (1-p_j)^{\dot{n}} \ \frac{e^{-\lambda}\lambda^{\dot{n}+x}}{(\dot{n}+x)!} = \sum_{\dot{n}=0}^{\infty} \frac{1}{\dot{n}!x!} p_j^x (1-p_j)^{\dot{n}} \ \frac{e^{-\lambda}\lambda^{\dot{n}+x}}{1} = \frac{e^{-\lambda}(\lambda p_j)^x}{x!} \sum_{\dot{n}=0}^{\infty} \frac{(1-p_j)^n \lambda^{\dot{n}}}{\dot{n}!}$

Como $e^{(1-p_j)\lambda} = \sum_{n=0}^{\infty} \frac{(1-p_j)^n \lambda^n}{n!}$ podemos simplificar la expresión anterior:

$P(N_{b,j} = x) = \frac{e^{-\lambda}(\lambda p_j)^x}{x!} \sum_{\dot{n}=0}^{\infty} \frac{(1-p_j)^n \lambda^{\dot{n}}}{\dot{n}!} = \frac{e^{-\lambda}(\lambda p_j)^x}{x!} e^{(1-p_j)\lambda} = \frac{e^{-\lambda p_j}(\lambda p_j)^x}{x!};\ x = 0, 1, 2, \ldots$

La expresión anterior demuestra, por tanto, que la distribución incondicional de $N_{b,j}$ es una variable de $Poisson(\lambda p_j)$, en la que el parámetro experimental λ es modificado por la probabilidad p_j. Además, se ve claramente que, si $p_j = 1$, entonces $N_{b,j} \sim Poisson(\lambda)$, lo que significa que si R es constante (longitud de los reads), obtenemos los resultados del Capítulo 3.

De este modo llegamos a que la cobertura en una determinada base b, Cov_b es la suma de variables $Poisson(\lambda p_j)$ independientes:

$Cov_b = \sum_{j=max(b-r_{max}+1,1)}^{min(b,f-r_{min}+1)} N_{b,j}$.

De modo que, por la reproductividad de la Poisson;

$$Cov_b \sim Poisson(\lambda \sum_{j=max(b-r_{max}+1,1)}^{min(b,f-r_{min}+1)} p_j) \qquad (2)$$

Como antes indicamos, queremos permitir que R tome los valores mínimo de $r_{min} = 1$ y máximo de $r_{max} = f$; por tanto, la ecuación anterior queda de la forma:

$Cov_b \sim Poisson(\lambda \sum_{j=1}^{b} p_j)$

En los pasos anteriores demostramos que, sí permitimos que el tamaño de los reads, R, sea aleatorio, llegamos a una estructura igual al contexto de R constante, pero en la que el parámetro de la distribución se ve modificado por las probabilidades obtenidas de la distribución de R. Esta característica, además, nos garantiza que podemos usar el mismo contraste propuesto en el Capítulo 3 para el contexto de R constante, pero modificando el cálculo de la región de rechazo del contraste con las probabilidades p_j.

4.1.1. Longitud de los reads R

Si queremos incorporar en el modelo la posibilidad de reads aleatorios, tenemos que estudiar la distribución de la variable longitud de los reads, R, que utilizaremos para obtener el valor de Cov_b esperado. Para conocer el comportamiento en distribución del tamaño de los reads, R, recurrimos a la

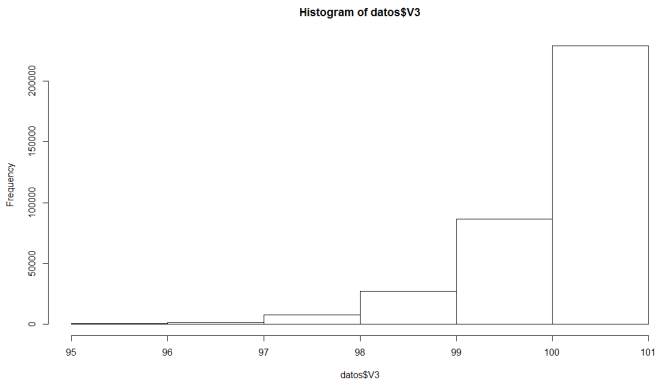

Figura 4.1: Histograma de la longitud de los reads, R, filtrados, de una muestra real

información obtenida de todas las regiones de una muestra normal, sin ninguna alteración.

La distribución de los reads nos da la probabilidad de la longitud de los reads, p_j. Los reads obtenidos de los experimentos de secuenciación tienen una longitud máxima igual al número de ciclos del secuenciador, de modo que en un proceso 100 % eficaz, todos los reads serán del mismo tamaño. Sin embargo, aparecen reads de longitud menor y mayor debido a un proceso de adiciones de bases que pueden adelantarse o retrasarse a su ciclo de incorporación: estos fenómenos se llaman *prephasing* y *phasing*.

El efecto del *prephasing* y *phasing* se puede mitigar mediante la aplicación de correcciones. Las muestras suelen presentar solamente un pequeño número de secuencias diferentes, produciendo distribuciones degeneradas en el número de ciclos llevados a cabo en el secuenciador. Al secuenciar, por ejemplo, una región de $0,5 - 1,5Mb$ (dependiendo de la librería de captura utilizada), el número de secuencias diferentes puede ser menor que en un experimento de secuenciación del genoma completo. En la Figura 4.1 observamos el histograma de los reads ya filtrados de una muestra real.

Vemos cómo en el histograma de la Figura 4.1 presenta una densidad casi degenerada en el valor 101. Esta asimetría tan fuerte después de los procesos de corrección es la forma más habitual de densidad de R. Los tamaños de los reads del histograma son los que se utilizarán para calcular la cobertura.

Dado que desconocemos si el comportamiento en distribución de la longitud de los reads, R, pertenece a alguna familia paramétrica, podemos optar por un método no paramétrico para calcular las probabilidades que utilizaremos. Una solución sencilla es utilizar la distribución empírica $(Fn(x) = \frac{1}{n}\sum_{i=1}^{n} I(X_i \leq x))$ de la longitud de los reads observados (R) para obtener los p_j. A lo largo de este tema usaremos la distribución empírica de la longitud de los reads, $(Fn_R(\cdot))$ para las técnicas propuestas en este capítulo.

Los scrips necesarios para replicar los resultados de esta sección se encuentran en el archivo *densidad_reads.R* (página 78).

4.2. Simuladores

A la hora de incluir la posibilidad de que la longitud de los reads sea aleatoria a la hora de construir un simulador de coberturas, hay que tener en cuenta ciertas consideraciones, aunque en esencia el procedimiento es el mismo que en el contexto de longitud de los reads constante (Capítulo 3).

Para construir el simulador seguimos el mismo procedimiento para el contexto de R constante. Primero, definimos el tamaño de la región f que queremos simular; definimos $M_e(R)$, que es la mediana de la longitud de los reads; calculamos la distribución empírica de R, $Fn_R(\cdot)$, usando datos reales (podríamos tratar de simular la variable de la longitud de los reads). Definimos el λ, de dos posibles formas. O bien calculamos el λ de una muestra real, como propusimos en el anterior modelo, de la forma $\hat{\lambda} = \frac{1}{f-r_{min}} \sum_{b=1}^{f-r_{min}} N_b$ $(b = 1, \ldots, f - r_{min})$, o bien especificamos $\hat{\lambda}$ a través de la altura en el centro que queremos que alcance la simulación. Por ejemplo: si queremos que nuestra curva alcance una cobertura determinada Cov_{centro} en la región central, el λ se estima como: $\lambda = Cov_{centro}/M_e(R)$.

Con estos elementos empezamos simulando f valores de una variable de $Poisson(\lambda)$ independientes e idénticamente distribuidos, que representan el número de reads que empiezan en cada una de nuestras bases b $(b = 1, ..., f)$; es lo que llamaremos N_b. Después, a través de la distribución empírica, o a través de una simulación, arrojamos un número n de tamaños de reads; n es la suma de todos los eventos de todas las f bases, $n = \sum_{b=1}^{f} N_b$. Para arrojar datos de longitud de los reads desde la empírica con los datos de nuestra muestra real hacemos lo siguiente. Para cada índice i $(i = 1, \ldots, n)$ de la muestra de longitudes de los reads que tenemos, arrojamos una uniforme $U_i \sim U(0, 1)$ y obtenemos nuestra remuestra $X_i^* = X_{[nU_i]+1};$, de esta forma vamos seleccionado n tamaños de regiones remuestreados de los datos reales.

Seguidamente construimos un número n de vectores de X_i^* unos, $R_{b,1}, \ldots, R_{b,N_b}$, donde $b = 1, \ldots, f$ y $N_b = 0, \ldots, N_b$, que representan a cada read que empieza en cada base b y que tienen una longitud R_{b,N_b} $(\vec{v}_{R_{1,1},1}, \ldots, \vec{v}_{R_{1,N_1},n-N_1}, \ldots, \vec{v}_{R_{f,1},n-N_f}, \ldots, \vec{v}_{R_{f,N_f},n})$. Estos vectores se colocan en una matriz $M_{n\times f}$ con f columnas, tantas como bases tiene la región, y n filas, tantas como reads hemos creado. Cada vector $\vec{v}_{R_{b,N_b},k}(k = 1, ..., n)$ se coloca en su fila R_{b,N_b} correspondiente al número de read creado, y en la columna b en la que empieza el read. Los reads que son más largos que la distancia de su base b a su base f son eliminados ($\left|\vec{v}_{R_{b,N_b},k}\right| > b - f + 1$).

Por último, sumamos las columnas de la matriz y obtenemos de esta forma la cobertura, Cov_b, para cada base b.

Vemos un ejemplo para una región de tamaño $f = 312$ bases, un tamaño mediano de reads de $M_e(R) = 101$, una cobertura en la parte central de 130 y un parámetro experimental λ =1.254717. Esta región simulada puede verse en la Figura 4.2.

El procedimiento para poder simular alteraciones no difiere del presentado en el Capítulo 3 por lo que no lo describimos aquí. El script necesario para simular estas curvar se encuentra en el archivo *SimuladorR_aleatorio*. En este script también se incluye la posibilidad de generar alteraciones.

4.3. Intervalo de confianza

Una vez definido el proceso de simulación cuando tenemos un tamaño de reads, R, aleatorio, podemos generar un intervalo de confianza aplicando MonteCarlo. Para probar esta técnica generamos 1000 réplicas de una región de tamaño $f = 312$ con tamaños de reads aleatorios y obteniendo dichos tamaños de reads arrojando valores de la distribución empírica del tamaño de los reads, $Fn_R(\cdot)$ y con un parámetro experimental λ =1.254717. En la Figura 4.3 vemos las 1000 simulaciones de una región.

Para obtener los extremos del intervalo por MonteCarlo, construimos una matriz $M_{f\times m}$ donde se colocan cada uno de los m (en este ejemplo $m = 1000$) perfiles de cobertura simulados de longitud f. Después, cada columna, que representa una base de la región de tamaño f, es ordenada de menor

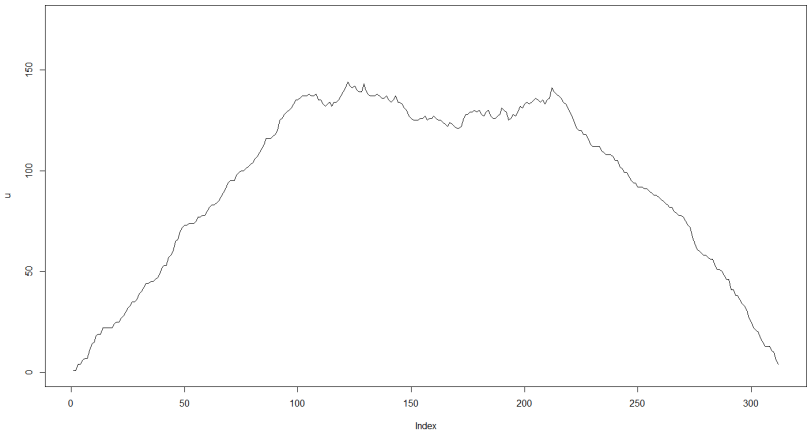

Figura 4.2: Perfil de cobertura de una región simulada de tamaño 312 bases, un tamaño mediano de reads de $M_e(R) = 101$ y un parámetro experimental λ =1.254717; simulada con el script *SimuladorR_aleatorio.R* en el que la longitud de los reads, R, se calcula a través de la distribución empírica de una muestra real

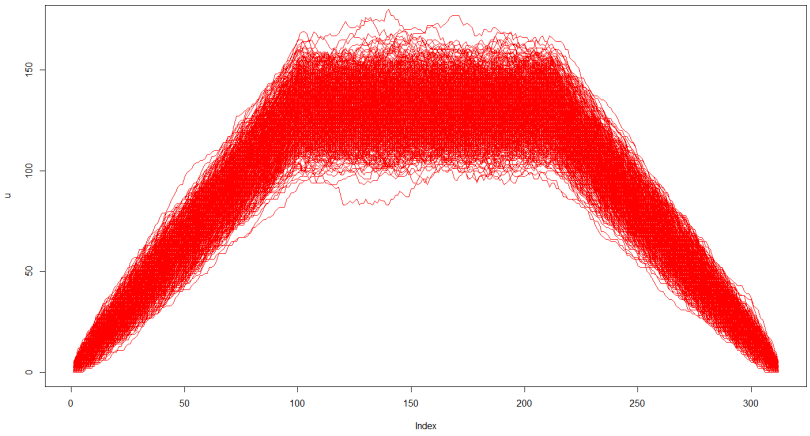

Figura 4.3: Perfil de cobertura de 1000 regiones simuladas de 312 bases (escenario de la Figura 4.2)

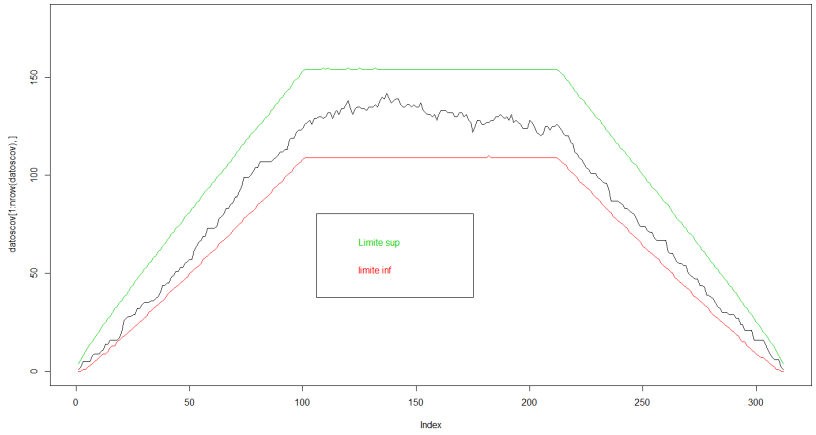

Figura 4.4: Perfil de cobertura de la región real de 312 bases de la Figura 3.2 y curvas de los límites del intervalo con un α del 0.05

a mayor y nos quedamos con el valor entero más próximo a $(m \times (1 - \frac{\alpha}{2}))$ para el extremo superior del intervalo y $(m \times \frac{\alpha}{2})$ para el valor inferior del intervalo, siendo α el nivel de confianza prefijado. A continuación aplicamos esta técnica a la región real de la Figura 3.2 (longitud de la región $f = 312$, λ =1.254717 y tamaño mediano de los reads $R = 101$), y después realizamos lo mismo con la región simulada de la Figura 4.2 (longitud de la región $f = 312$, λ =1.254717 y tamaño mediano de los reads $R = 101$). Vemos en la Figura 4.4 y en la Figura 4.5, los intervalos confianza tanto para la región real como la simulada.

En la Figura 4.4 vemos cómo la región real se encuentra completamente dentro de los limites de confianza de MonteCarlo que hemos generado, mientras que en la Figura 4.5 vemos cómo la región simulada presenta una pequeña zona en la cola de la derecha de la región que se sale del limite superior del intervalo de confianza de MonteCarlo. Sabemos que el perfil de cobertura lo hemos simulado sin alteraciones, así que el hecho de que ese fragmento se salga de los límites de confianza es debido al azar.

El script utilizado para simular regiones con R aleatorio y alteraciones se encuentra en *ICR_aleatorio.R* (página 83).

4.4. Contraste basado en el modelo

En el Capítulo 4 demostramos que al permitir que la longitud de los reads, R, sea aleatoria, la cobertura para una determinada base b, Cov_b, es una variable de $Poisson(\lambda p_j)$ de modo que, para el caso de longitud de los reads aleatorio , Cov_b tiene la misma estructura que para el caso de tamaño de los reads constante, por lo que podemos volver a aplicar el contraste anterior pero calculando los valores esperados bajo el modelo (2):

$$Cov_b \sim Poisson[\lambda \sum_{j=max(b-r_{max}+1,1)}^{min(b,f-r_{min}+1)} p_j] = Poisson(\Lambda)$$

Como dejamos que R tome cualquier valor natural entre 1 y f (es decir, $r_{min} = 1$ y $r_{max} = f$), las condiciones del sumatorio desde $j = max(b - r_{max} + 1, 1)$ hasta $min(b, f - r_{min} + 1)$ pasan a ser desde $j = 1$ hasta b, de modo que:

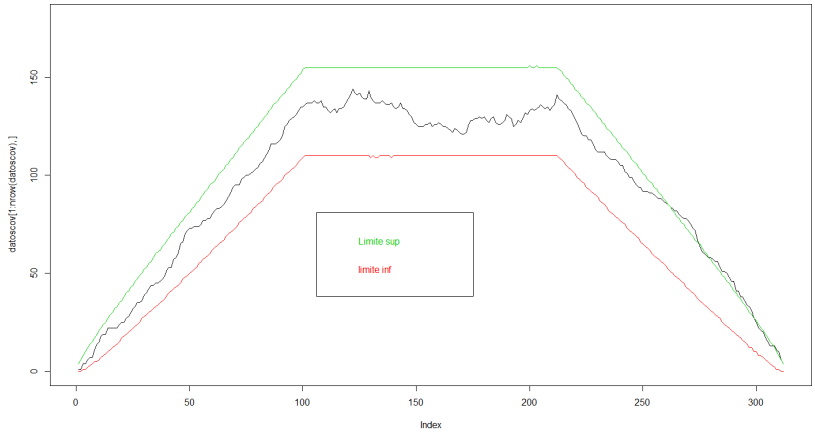

Figura 4.5: Perfil de cobertura de la región simulada de 312 bases de la Figura 4.2 y curvas de los límites del intervalo con un α del 0.05

$$Cov_b \sim Poisson[\lambda \sum_{j=1}^{b} p_j] = Poisson(\Lambda)$$

Nuestra hipótesis nula es $H_0 : \Lambda = \Lambda_0$, la cual testeamos frente a una alternativa $H_1 : \Lambda \neq \Lambda_0$, pero para los casos más sencillos de alteraciones (ver Cuadro 3.1), la hipótesis alternativa se reduce a unos posibles valores, $H_1 : \Lambda = \Lambda_1$, donde Λ_0 y Λ_1 son constantes conocidas, permitiéndonos construir un contraste con hipótesis nula y alternativa simples, aplicar el lemma de Neyman-Pearson al igual que en el Capítulo 3 y así obtener el test UMP. Recordemos que contamos con una única observación, x, para el test.

La alteración más sencilla es la que corresponde a una deleción en homocigosis, en la cual el valor de Λ_1 es $\Lambda_1 = 0$, lo que implica $\Lambda 1 < \Lambda_0$. El test de Neyman-Pearson queda de la siguiente manera:

$$\phi(x) = \begin{cases} 1 \; si \; x < S \\ \gamma \; si \; x = S \\ 0 \; si \; x > S \end{cases}$$

donde el valor S se debe de escoger de forma que $\alpha = E[\phi(x)/H_0] = P_0(x < S) + \gamma P_0(x = S)$, y las probabilidades P_0 se refieren a una $Poisson(\Lambda_0)$.

En la práctica al igual que lo hicimos en el Capítulo 3 vamos a utilizar los p-valores calculados como $P_0(Poisson(\Lambda_0) \leq x)$ para el caso de deleciones $\Lambda_1 < \Lambda_0$ y $P_0(Poisson(\Lambda_0) \geq x)$ para el caso de duplicaciones $\Lambda_1 > \Lambda_0$.

Una vez presentado el contraste, lo aplicamos a la misma muestra real que utilizamos para el contexto de R constante, se trata de una región de tamaño $f = 312$ bases un tamaño de reads aleatorio con un valor mediano de read de $R = 101$ y un parámetro experimental λ =1.254717.

El primer paso es calcular los valores esperados de acuerdo a la ecuación (2). Para ello, partimos de λ que es el número esperado de reads que empiezan en cada base b $(1, \ldots, f)$; después calculamos la distribución empírica de R, $Fn_R(\cdot)$, de la cual se obtiene las probabilidades p_j =

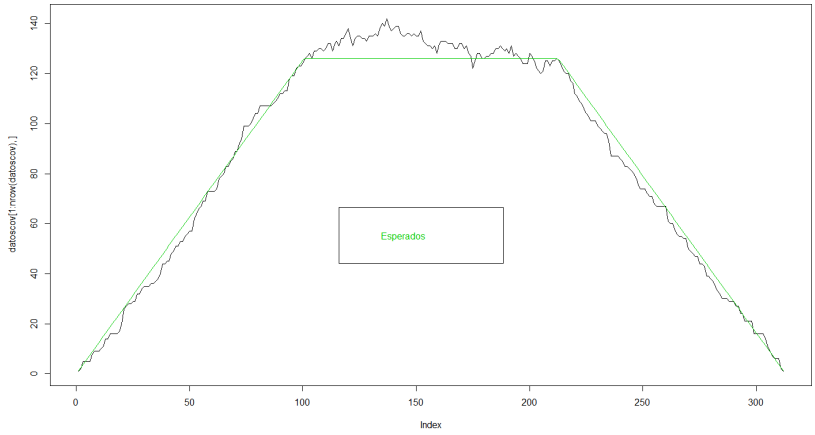

Figura 4.6: Perfil de cobertura de una región sin alteraciones obtenida de la muestra real de la Figura 3.2 y curva de valores esperados bajo H_0 obtenidos de la distribución empírica del tamaño de los reads (color verde)

$(1 - Fn_R(b - j)) - (1 - Fn_R(f - j + 1))$ (probabilidad de que el llegue el read desde j hasta b pero no salga de la región de tamaño f). Para todas las bases j $(1, \ldots, b - 1)$ calculamos $[p_j \times \lambda]$, después sumamos todos estos elementos y de esta forma obtenemos el valor esperado bajo H_0. Este el proceso se repite para todas y cada una de las bases, $b = 1, \ldots, f$.

La región representada en la Figura 4.6 pertenece a la misma muestra real sin alteraciones utilizada en el Capítulo 3, con un tamaño mediano de reads de 101, una región de longitud $f = 312$ bases, en la que el parámetro experimental es λ =1.254717. Calculamos los valores esperados bajo la hipótesis nula H_0 aplicando la ecuación (2), mediante el proceso descrito en esta sección. Finalmente obtenemos el resultado mostrado en la Figura 4.6.

En la Figura 4.6 se aprecia cómo se adapta la curva esperada a la curva real, y parece razonable que los valores de la curva real sigan el modelo propuesto, ya que sabemos que la muestra no presenta alteraciones.

En la Figura 4.7 vemos el resultado del contraste, usando p-valores, para $\Lambda_1 < \Lambda_0$, y después los p-valores ajustados mediante Benjamini y Hochberg (1995) para comparaciones múltiples. Si comparamos las figuras 4.7 y 4.6 con las figuras 3.4 y 3.3 (correspondientes al modelo simplificado en el que R se asume constante), no se aprecian diferencias importantes.

El script necesario para llevar a cabo este estudio se encuentra en el archivo *contraste_R_aleatorio.R* (página 88).

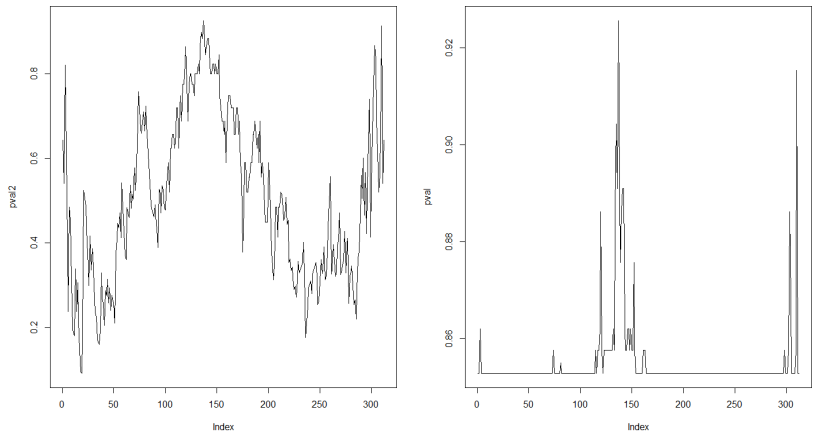

Figura 4.7: Izquierda: p-valores sin ajustar suponiendo que la distribución de la variable R se obtiene usando la empírica. Derecha: p-valores ajustados mediante Benjamini y Hochberg (1995). Muestra real de la Figura 3.2

Capítulo 5

Enfoque funcional

Podemos enfocar el problema de localizar una región con anomalías desde un punto de vista de datos funcionales, donde el perfil de coberturas de una región es tratado como un dato funcional generando una única curva.

Para poder conocer si nuestra curva problema de tamaño f presenta o no alguna alteración, la comparamos con un número de curvas de igual longitud, simuladas a partir del escenario son alteraciones. Para simular curvas recurrimos a los simuladores presentados en los Capítulos 3 y 4 (tanto el simulador del Capítulo 3, que usa un tamaño de reads constante [*Simulador.R*] (página 85), como el simulador del Capítulo 4, que utiliza un tamaño de reads aleatorio [*SimuladorR_aleatorio.R*] (página 88)).

Dada la naturaleza de los datos, parece razonable que las diferencias entre las curvas se manifestaran como diferencias en el área bajo las mismas, por lo que parece razonable trabajar en un espacio L_2, aunque se pueden considerar espacios con otras normas. Esto nos permite calcular determinados estadísticos como la mediana. El proceso se detalla en la sección siguiente.

5.1. Detección de una región alterada

Con un conjunto de regiones que estará compuesto de nuestra curva problema, en la que deseamos conocer si presenta alguna alteración, y un número suficiente de curvas simuladas, procedemos a calcular las profundidades de todas las curvas. La profundidad nos marcará como de cercana es una curva a la curva media del proceso; por lo tanto las curvas más profundas serán curvas sin alteraciones.

De forma general, una región que ha sido generada por un proceso distinto a las demás, presentara un valor de profundidad menor. El conjuto de curvas simuladas han sido generadas todas por el mismo proceso, de modo que las profundidades de las mismas serán grandes. Si nuestra curva problema presenta una profundidad menor que la menos profunda de las curvas simuladas, ésta presenta una alteración.

Para que el método sea consistente, necesitamos simular un número suficiente de regiones sin alteración; cuanto mayor sea el conjunto de curvas simuladas, más fiable es el método.

El primer paso es decidir que medida de profundidad debemos aplicar a este contexto. Aunque existe un abanico de posibilidades, utilizaremos la profundidad modal (Cuevas et al., 2007), ya que, en nuestra experiencia, arroja buenos resultados.

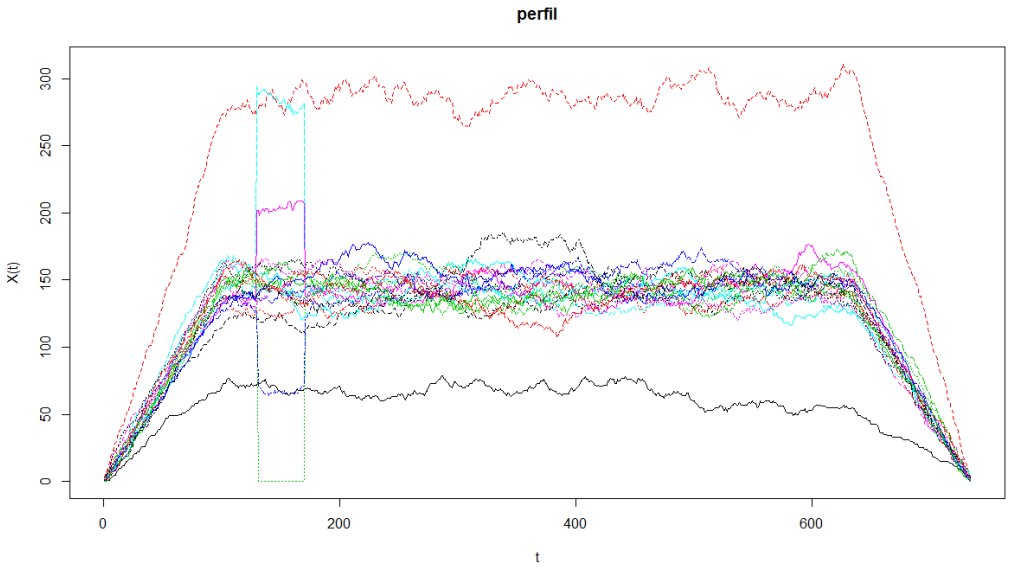

Figura 5.1: 21 Regiones simuladas de tamaño $f = 734$. Se aprecian las curvas que difieren de la mayoría: son las curvas que presentan alteraciones (ver texto)

De forma general, la profundidad modal es una medida de cuántos puntos hay en la vecindad y ésta se asemeja al estimador de la densidad tipo kernel (Parzen-Rosenblatt) pero con una ventana que no necesariamente tiende a cero conforme aumenta el número de datos.

Una vez obtenidas las profundidades de las curvas, esperamos (com ose ha dicho) que las curvas sin alteraciones serán las curvas más profundas, y las curvas con alguna alteración sean datos funcionales cada vez menos profundos; pero cuanto mayor espacio de la región ocupe la alteración y en los casos de alteraciones en homocigosis, menor será la profundidad, y por lo tanto serán más fáciles de diferenciar de las curvas simuladas.

A continuación probaremos este método. Para ello simulamos 21 regiones de tamaño $f = 734$ bases, con un tamaño de reads $R = 101$ constante y un parámetro experimental λ =1.254717. De las 21 curvas simuladas hay una curva con una deleción en homocigosis que ocupa toda la región de 734 bases, una curva con una duplicación en homocigosis que también ocupa la totalidad de la región; y también 4 curvas con las 4 tipologias de alteración (deleción en homocigosis, deleción en heterocigosis, duplicación en homocigosis y duplicación en heterocigosis), pero que sólo abarcan un tamaño de 40 bases; de la base 130 a la base 170. El resto de las 15 regiones son simulaciones sin alteración. Podemos ver estas curvas en la Figura 5.1.

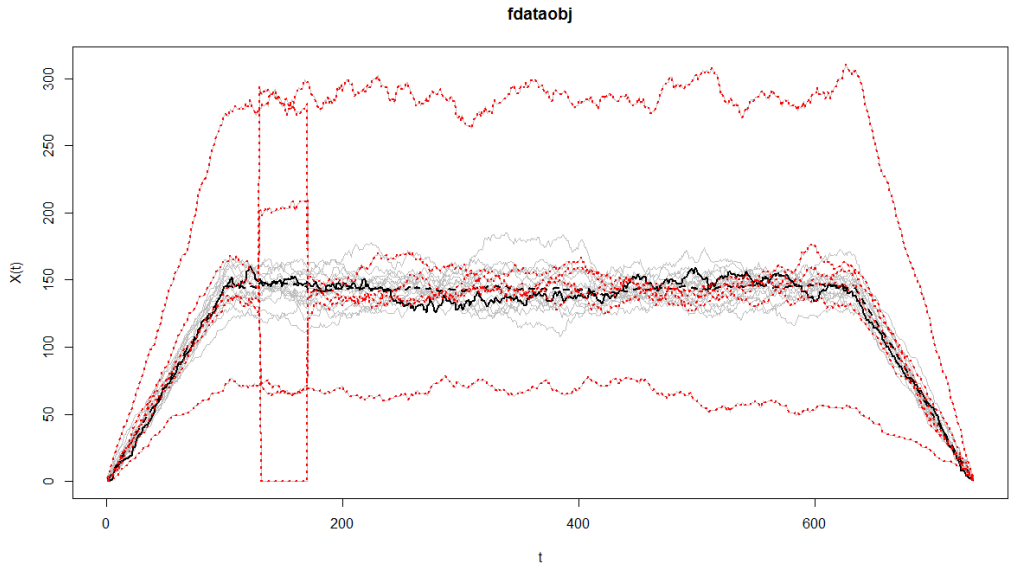

Figura 5.2: 21 Regiones simuladas de tamaño $f = 734$, 6 de las cuales con alteraciones (ver texto). Marcadas en rojo, las 6 regiones con menor profundidad modal. Se aprecia que las regiones que presentan alteraciones son las de menor profundidad modal. La curva discontinua representa la cobertura mediana de las 21 regiones y la curva negra continua es la curva con mayor profundidad modal

Hemos calculado la profundidad modal de las 21 curvas; en la Figura 5.2 podemos ver marcadas en rojo las curvas menos profundas y, como esperábamos, vemos cómo las curvas marcadas en rojo se corresponden con las curvas en las que simulamos una alteración.

En la Figura 5.2 vemos cómo, para la simulación anterior, parece que el enfoque funcional ofrece resultados prometedores para detectar regiones alteradas, incluso para alteraciones relativamente pequeñas, ya que incluso el método llega a detectar bien las regiones con una alteración de tan solo 40 bases.

En la simulación anterior hemos incluido varias curvas con alteraciones para probar el método pero, en la práctica se tiene una única curva real y un conjunto de curvas sin alteración, simuladas; si la curva real presenta algún tipo de anomalía, sea grande o pequeña, ésta se manifestará con una profundidad menor.

Procedemos ahora a realizar el análisis funcional con la región real normal que hemos utilizado a lo largo de este trabajo (Figura 3.2). Es una región de tamaño $f = 312$ con reads de tamaño $R = 101$ constante y un parámetro experimental λ =1.254717. Simulamos, además, 10 regiones normales de

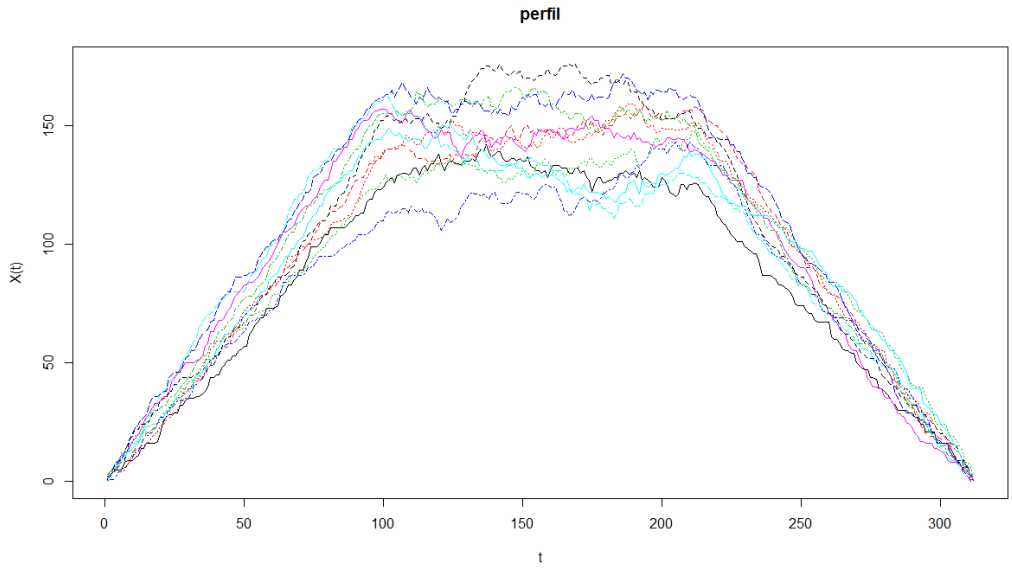

Figura 5.3: 10 Regiones sin alteración simuladas de tamaño 312 y la región real no simulada de la Figura 3.2 (negro). No se aprecia que exista una curva que difiera de la mayoría, sugiriendo que nuestra curva problema no presenta alteraciones

tamaño $f = 312$, con los mismos parámetros de R y λ. Realizamos el análisis funcional de estas 11 curvas y mostramos el resultado obtenido en la Figura 5.3. Al igual que antes calculamos sus profundidades modales y seleccionamos la curva que tiene menos profundidad modal. La curva con menor profundidad modal puede verse en rojo en la Figura 5.4. Al tratarse de una de las curvas simuladas, se concluye que la curva real no presenta alteraciones. Estos resultados prometedores abren la posibilidad de construir un test formal a nivel α para la profundidad.

En resumen, por estos dos ejemplos anteriores, uno completamente simulado y otro con una región normal real sin alteraciones, que el enfoque funcional del problema ofrece resultados prometedores y que, además, son acordes con los distintos métodos utilizados en este trabajo.

El script necesario para la realización de estos ejemplo, se encuentra en el archivo *DF.R* (página 78) que hace uso la librería *fda.usc* (Manuel Febrero Bande et al. 2016).

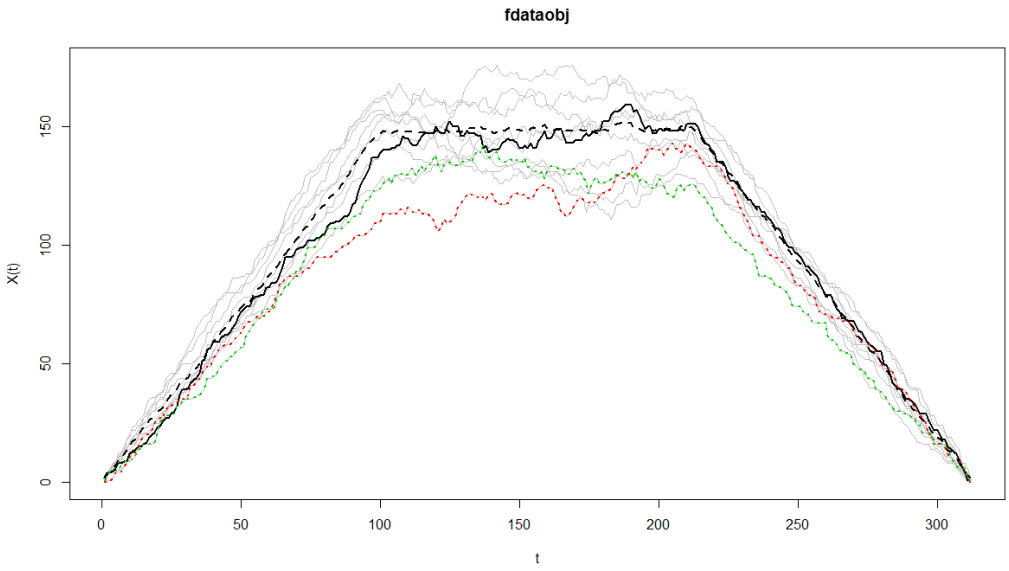

Figura 5.4: Marcada en rojo, la región con menor profundidad modal (de entre las curvas de la Figura 5.3), que es una de las curvas simuladas. En negro, las curvas con mayor profundidad modal. La región verde es nuestra región problema. No se aprecia ninguna región que presente alteración, ni siquiera la de menor profundidad modal. La curva discontinua representa la cobertura mediana de las 10 regiones

Capítulo 6

Algoritmo CONTRA

Con la información de la cobertura, que es el conteo de cuántas veces se ha secuenciado cada base, o basándose en otras medidas, se desarrollan diferentes algoritmos de análisis para poder localizar distintas alteraciones genéticas. Si nos centramos en las alteraciones genéticas CNVs (variaciones en el número de copias), estas alteraciones producen cambios en los perfiles de cobertura de las muestras.

Cuando una muestra presenta una duplicación en heterocigosis, la cobertura aumenta un 50 % y, si es en homocigosis, se produce un aumento del 100 %. Mientras que si es una deleción en heterocigosis, la cobertura cae un 50 %, y si se trata de una deleción en homocigosis, la cobertura cae un 100 %.

Cuando usamos un ADN diploide, como el humano, tenemos 2 copias para cada secuencia de ADN; cuando tenemos una deleción en heterocigosis, sólo nos falta una de las copias, y cuando se produce el proceso de secuenciación, los reads para esa zona serán un 50 % menores. De forma contraria, si la duplicación es en heterocigosis, tenemos 3 copias para una secuencia de ADN, por lo que la cobertura de esa región aumenta un 50 %. En los casos de homocigosis, si estamos ante una deleción, no habrá copias para una determinada región, lo que significa que no existirán reads para esa región y la cobertura será de cero, y en el caso de la duplicación tendremos 4 copias de la región, lo que implica aumentar un 100 % la cobertura de esa región.

La realidad puede ser más compleja, ya que estas duplicaciones pueden estar compuestas por repeticiones en tándem, y podemos tener más de 4 copias para una secuencia, y las conoceremos como amplificaciones, y casi todas las situaciones intermedias que podamos imaginar como el caso de inversiones.
Visualmente, esto se aprecia como valles o picos en los perfiles de cobertura.

El grueso de las herramientas se enfoca en comparar estos perfiles, entre un caso y un control, y contrastar la existencia de alguna de estas alteraciones.

6.1. CONTRA

Se necesita un método robusto para analizar las variaciones en el número de copias (CNVs). El enfoque habitual al problema consiste en comprar los perfiles de cobertura de muestras problema con los de muestras control. Para poder comparar los perfiles, en general, los algoritmos existentes usan un control o pool de controles que saben con certeza que no presentan alteraciones en las regiones seleccionadas, y el control o controles son comparados frente a la muestra problema.

En este capítulo tratamos de explicar al funcionamiento y problemas del algoritmo CONTRA (Li

et al., 2012), que ha sido la herramienta con la que se ha empezado a tratar el problema de detección de CNVs en el marco de este trabajo, y el cual es la base de partida del proyecto desarrollado por Gradiant en colaboración con el CHUS.

CONTRA trata de calcular el número de copias ganadas y pérdidas para cada región de estudio basándose en la cobertura normalizada. La estrategia se basa en calcular el log-ratio a nivel de base entre un control o pool de controles y la muestra problema (también llamada caso). La razón de trabajar con el log-ratio entre el caso y el control tiene como fin eliminar el sesgo que produce el contenido en GC, del que hablamos en el Capítulo 2. Según los autores de CONTRA, trabajar con los log-ratios permite eliminar este sesgo, sin necesidad de recurrir a una técnica específica. El sesgo GC, o sesgo por el contenido en guanina-citosina, se produce en experimentos de secuenciación de nueva generación, en especial en Illumina. Cuando el contenido GC es muy alto o muy bajo en una región, la cobertura se reduce.

El algoritmo, después del cálculo de los log-ratios, corrige la diferencia en el tamaño de las librerías producidas para el caso y el control. Esto se produce cuando el caso y el control no vienen de un experimento de secuenciación con los mismos parámetros experimentales; como por ejemplo el tamaño de los reads o el nivel de cobertura. Por último, el algoritmo estima las varianzas y la media de los log-ratios para cada región de la muestra para poder hacer un contraste de las múltiples regiones, suponiendo normalidad.

Hasta la fecha, se han publicado un número limitado de métodos para el análisis de los datos de CNVs en experimentos TR (targeted resequiencing). Una de esas herramientas es ExomeCNV (Sathirapongsasuti et al., 2011), la cual se ha diseñado específicamente para la captura de todo el exoma e implica el modelado de log-ratio del numero de reads en la región, en lugar de la cobertura por base; utilizando la transformación Geary-Hinkley (Hayya, J. et al., 1975), obtiene la normalidad de estos log-ratios y utiliza esta cualidad para realizar un contraste. El método, sin embargo, no tiene en cuenta una serie de factores sesgadores, como las diferencias en el tamaño de la librería entre el caso y el control y el porcentaje de cobertura, que si son tenido en cuenta en el algoritmo CONTRA. El algoritmo ExomeCNV da buenos resultados en estudios poblacionales, mientras que resulta poco eficaz al trabajar con la muestra de un único individuo.

6.2. Pasos detallados del algoritmo

En esta sección detallamos con más profundidad los pasos que lleva a cabo el algoritmo CONTRA.

El punto de partida del algoritmo es obtener un control, que será el perfil de cobertura de un individuo normal, sin alteraciones en las regiones de interés. Este es el contexto habitual en el diagnóstico clínico pero, si carecemos de un control, CONTRA contempla la posibilidad de crear una "baseline" como control, construida con múltiples individuos de la siguiente forma. La "baseline" debe capturar la variación técnica de una plataforma, pero no variaciones debido a CNVs o polimorfismos de número de copias (CNP) en las muestras. Por lo tanto, la selección de las muestras debe apuntar a sujetos de origen diferente (por ejemplo, no relacionados genéticamente) para que se puedan diluir las alteraciones debidas a los individuos, como los CNP o SVN (variaciones de un único nucleótido). Para la creación de la "baseline", primero definimos el tamaño de la librería para cada elemento de la "baseline" de la siguiente manera:

$$L_s = N_s \times read\_length \times \%target \qquad (3)$$

donde L_s es el tamaño de la librería de la muestra s, N_s es el número total de reads para la muestra

s y, el $\%target$ es la eficacia del proceso de secuenciación. Se observó que, si bien el porcentaje de las lecturas en el objetivo, $\%target$, es estable a través de las muestras, esto puede variar de forma significativa en algunos casos, debido a las condiciones experimentales y reducción de la eficiencia de captura. Dicha variación técnica se elimina mediante la incorporación de porcentaje en el objetivo en la ecuación (3).

La definición de tamaño de la librería en (3) tiene en cuenta las muestras secuenciadas con diferentes longitudes en los reads, y permite que muestras de distintos procesos experimentales sean agrupados juntos en un control.

A continuación se calcula la cobertura a nivel de base para cada base específica en cada muestra s de la forma:

$$d_b = c_b \times \frac{\bar{L}}{L_s} \qquad (4)$$

donde d_b es la cobertura ajustada, c_b es la cobertura cruda de la muestra s en la base específica b, y $\bar{L}$ es la media geométrica de todos los L_s del conjunto de control.

Una vez calcualda la cobertura ajustada para cada base del control, se calcula una media truncada ($\bar{d_b}$) eliminando los valores atípicos (10 % en ambos extremos), y así se consiguen eliminar las CNP que son específicas de un pequeño número de individuos en el control. Y de esta forma obtenemos un perfil de cobertura ajustada que será nuestro control.

Si L_s es consistente a través de las muestras, la varianza de $\bar{d}$ es inversamente proporcional al tamaño del conjunto de control, lo que mejora la estabilidad del análisis de log-ratios cuando el número de elementos del control es grande.

En el supuesto de que se disponga de una muestra control, no es necesario crear una "baseline"; simplemente calcularemos la cobertura ajustada para el control usando la ecuación (5) de abajo, y así obtenemos el perfil de cobertura del control, pasando a ser $\bar{L}$ la media geométrica del tamaño de la librería entre el caso y el control. El uso de la media geométrica truncada para el escalado se ha discutido en el contexto del análisis de RNA-seq (Robinson et al., 2010):

$$d_b = c_b \times \frac{\bar{L}}{L_{control}} \qquad (5)$$

El segundo paso es obtener el perfil de cobertura ajustada de nuestra muestra problema, para lo cual se reescala la cobertura bruta con la ecuación siguiente:

$$d_b = c_b \times \frac{\bar{L}}{L_{caso}} \qquad (6)$$

Las regiones de interés con una cobertura bruta inferior a un umbral prefijado son eliminadas (por defecto, se requiere al menos 10 bases con $c_b > 10$).

El siguiente paso del algoritmo consiste en calcular los log-ratios con la cobertura del caso y el control ajustados (d_b del caso y el control), que es lo que se llamará lr_b:

$$lr_b = log_2 \frac{d_b CASO}{d_b CONTROL} \qquad (7)$$

Para cada región de interés, se calcula el log-ratio a nivel de región (RLR), $lr_{RLR} = \bar{lr_b}$, tomando la media del log-ratio a nivel de base para cada región.

Como veíamos en el Capítulo 3 con el estudio de la varianza, la varianza de d_b no es constante a lo largo de la región, debido a que, al tratarse de sumas de procesos de Poisson, la varianza aumenta con-

forme aumentamos el número de variables de Poisson que se suman. Al desplazarnos desde los extremos de la región hacia el centro, esto se traduce en mayor varianza en la parte central en comparación con las colas de la región. Esto produce que, al calcular la media para la región, estemos obviando esta naturaleza en los perfiles de cobertura. De modo que los lr_b heredan una heterogeneidad en su varianza a lo largo de la región y, al hacer la media de ellos, estamos cometiendo error en la varianza del estimador.

La razón para utilizar el log-ratio a nivel de base es eliminar al máximo el efecto del contenido GC, un sesgo que se ha observado en procesos de secuenciación de segunda generación, en particular Illumina (Aird et al., 2011).

La razón por la que el log-ratio a nivel de base elimina el sesgo GC es porque, tanto en el caso como en el control, o en la "baseline", el sesgo GC se localiza en las mismas zonas. De esta forma el ratio no se ve alterado, pero del mismo modo que comentábamos que para lr_b la varianza varía al variar el nivel de cobertura, en las zonas donde se produce el sesgo GC el parámetro de la Poisson es menor, y por tanto la varianza es menor que en una zona normal.

6.2.1. Corrección del tamaño de la librería

Los autores de CONTRA (Li et al., 2012) aseguran que el log-ratio a nivel de base es linealmente dependiente del log-ratio de la cobertura cuando los tamaños de librería entre el caso y el control son desiguales. Para eliminar el sesgo, se realiza un ajuste lineal entre log-ratio y la log-cobertura utilizando todos los RLR. La línea de ajuste se corresponde con la neutralidad en el número de copias, y esta se resta de cada RLR en la etapa de corrección. El modelo lineal es de la siguiente forma:

$$\frac{1}{reglen}\sum_{b=1}^{reglen} log_2 \frac{d_b CASO}{d_b CONTROL} = a + b \times log_2\left(\frac{\frac{1}{reglen}\sum_{b=1}^{reglen} d_b CASO + \frac{1}{reglen}\sum_{b=1}^{reglen} d_b CONTROL}{2}\right) \quad (8)$$

Se tienen tantos datos como regiones. Se calcula la media de los log-ratios para cada región, esto es, la media de los log-ratio a nivel de base para cada región, para el caso ($d_b CASO$) y el control ($d_b CONTROL$). También se calcula la media para las coberturas ajustadas d_b y se plantea el modelo lineal (8). Con el modelo (8) ajustado se resta el ajuste a los log-ratios:

$$Adjusted.Logratios = Logratios - (a + b \times log.cov.mean)$$

siendo:

$$log.cov.mean = log_2\left(\frac{\frac{1}{reglen}\sum_{b=1}^{reglen} d_b CASO + \frac{1}{reglen}\sum_{b=1}^{reglen} d_b CONTROL}{2}\right)$$

Sin embargo, aplicando este procedimiento a una muestra vemos que no parece muy razonable un modelo lineal para los datos. Es más, ni siquiera parece haber un dependencia lineal en ellos. Esto se aprecia en las Figuras 6.1 y 6.2, donde se ve la presencia de valores atípicos.

6.2.2. Contraste

Los autores de CONTRA sostienen que los RLR corregidos se distribuyen normalmente para las regiones con similar cobertura. Por lo tanto, los RLR siguen una distribución normal, $N(\mu_d, \sigma_d)$, donde el subíndice d corresponde a la región de interés.

Esta afirmación la han validado con QQ-plots realizados sobre distintas plataformas de captura y a distintos niveles de cobertura, como se ve en la figura 6.3. Sin embargo estos gráficos no son suficientes para poder asegurar que el comportamiento de los log-ratios es normal, y de hecho vemos que muchos

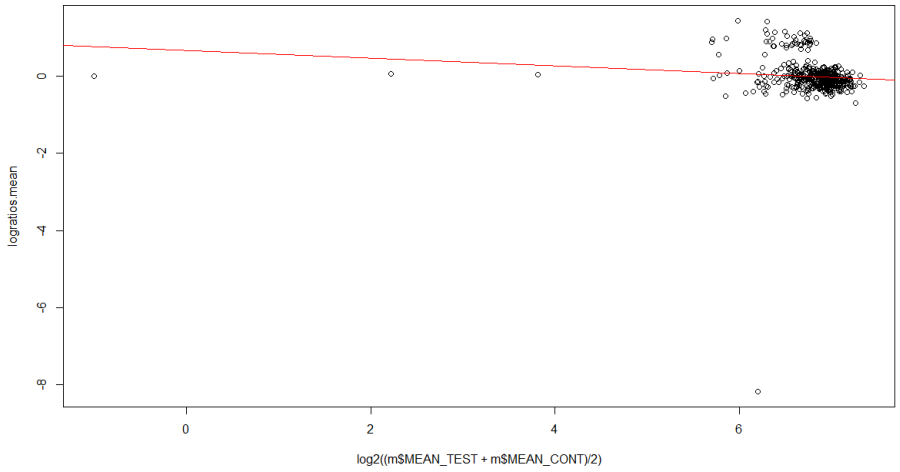

Figura 6.1: Ajuste lineal entre la media de los log-ratios para cada región, frente a la media del logaritmo de la cobertura ajustada del caso y el control

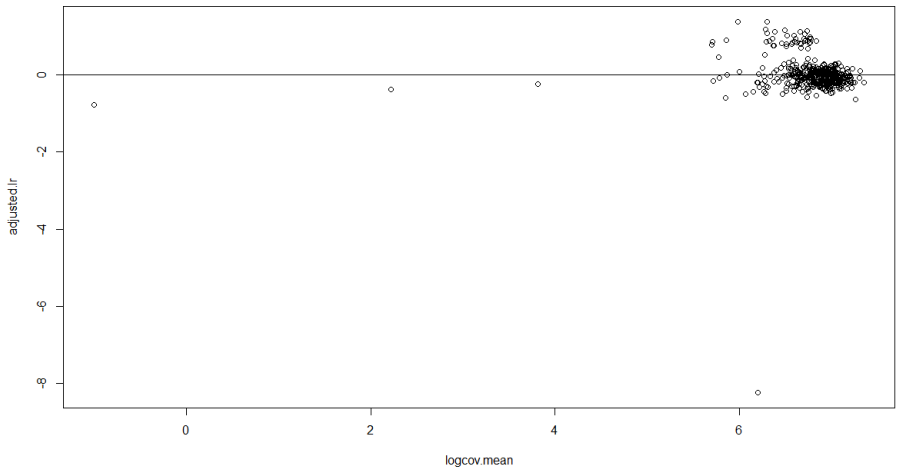

Figura 6.2: Ajuste lineal entre la media de los log-ratios para cada región, frente a la media del logaritmo de la cobertura ajustada del caso y el control, después de restar el ajuste de la Figura 6.1

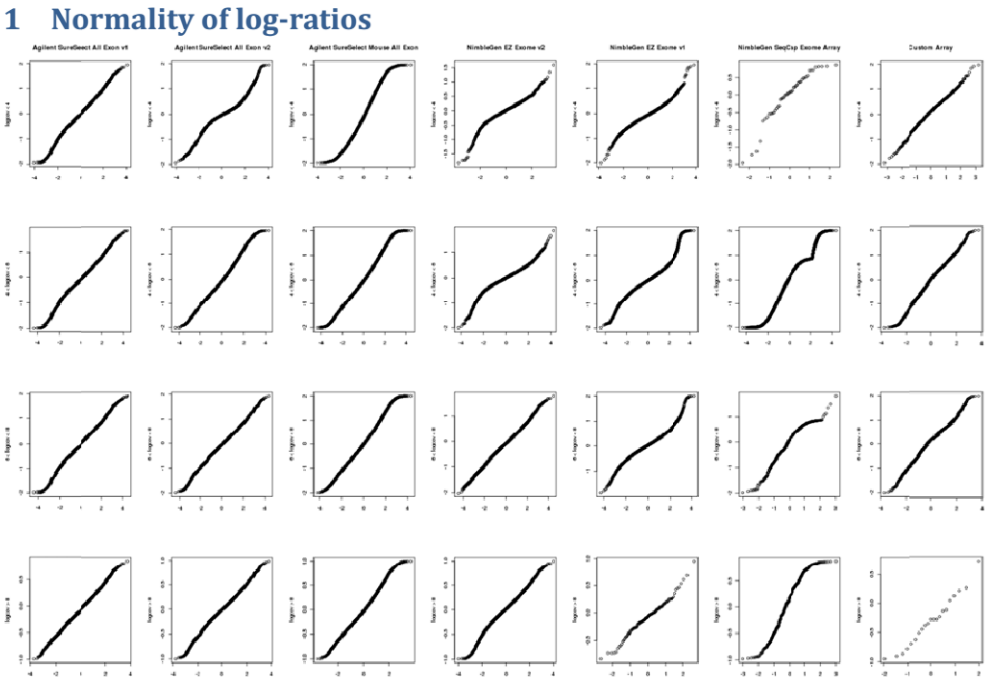

Figura 6.3: QQ-plots de los log-ratios para distintas regiones de diferentes plataformas de captura y distintos niveles de cobertura. Fuente: Li et al., 2012

de ellos no siguen un comportamiento normal. A su vez, no se apoyan en ningún otro tipo de análisis de la normalidad aparte de los QQ-plots. Esta suposición es la base del contraste llevado a cabo por el algoritmo, y puede que no sea correcta, o al menos no está validada.

Los parámetros de la normal, para el contraste, se son estimados de la siguiente manera. La media , μ_d se estima por la media de los *RLR*. Los autores de CONTRA comentan que dicha media estará próxima a cero después de la corrección del tamaño de la librería. Hay que destacar que después de la corrección de la librería esperamos que la media sea cero exactamente, ya que el algoritmo calcula la media para el contraste como:

$$Adjusted.Logratios = a + b \times log.cov.mean$$

Pero, como se ha realizado el ajuste (8), sabemos que los log-ratios ajustados (*Adjusted.Logratios*) ya son independientes de la log-cobertura (*log.cov.mean*).

Por otro lado, la desviación estándar, σ_d, se calcula utilizando el siguiente procedimiento:

(I) Las regiones se agrupan en bins, en función de su similitud en la cobertura media (*log.cov.mean*).

(II) Se calcula la desviación estándar de los log-ratios de todas las regiones de cada bin; es lo que llamamos *sd.logratios.perbin*, y se calcula la cobertura media (*log.cov.mean*) para cada bin; es lo que llamamos *log.cov.mean.perbin*.

(III) Se realiza un ajuste entre el logaritmo de la cobertura media de los bins y la desviación estándar de los log-ratios de cada bin, en escala logarítmica, obtenidos en el paso (II).

$$log_2 sd.logratios.perbin = a + b \times log.cov.mean.perbin$$

(IV) La σ_d se estima entonces para cada región con la curva anterior en función del logaritmo de su cobertura media dentro del bin (*log.cov.mean.perbin*), como se muestra en la figura 6.4.

El ajuste para estimar σ_d que se presenta aquí es lineal, pero puede realizarse de diversas formas como se ve en la figura 6.4. El ajuste lineal es más sensible a los outliers requiriendo que los bins tengan un mínimo de 1000 regiones para evitar el efecto de estos outliers, mientras que el ajuste exponencial es menos sensible a los outliers, pero comete más error en ambas colas y la relación recíproca es más sensible a cambios pequeños-medianos en la cobertura.

Por último se calculan los p-valores para un contraste bilateral para cada región, en base a la normal obtenida en los pasos anteriores; este p-valor se ajusta por las múltiples comparaciones mediante la aplicación de la corrección de pruebas múltiples de Benjamini y Hochberg (Benjamini y Hochberg, 1995). También el algoritmo permite al usuario establecer algunos límites arbitrarios en los recuentos de reads crudos en el caso y/o control debido al hecho de que los datos de secuenciación son muy ruidosos e impredecibles con recuentos bajos de lectura. Esto es útil cuando se espera que el control normal sea diploide, pero tiene regiones con muy pocos o ningún read, probablemente debido a errores de captura o artefactos de secuenciación. Excluir estas regiones es una etapa de filtrado recomendada.

6.3. Enfoque heurístico para la predicción de grandes CNV

Para la detección de grandes CNV que abarcan varias regiones, CONTRA propone un enfoque heurístico del problema. En primer lugar se realiza una segmentación binaria circular (Olshen et al., 2004) sobre los *RLR*, utilizando diferentes parámetros para conseguir diferentes resoluciones de seg-

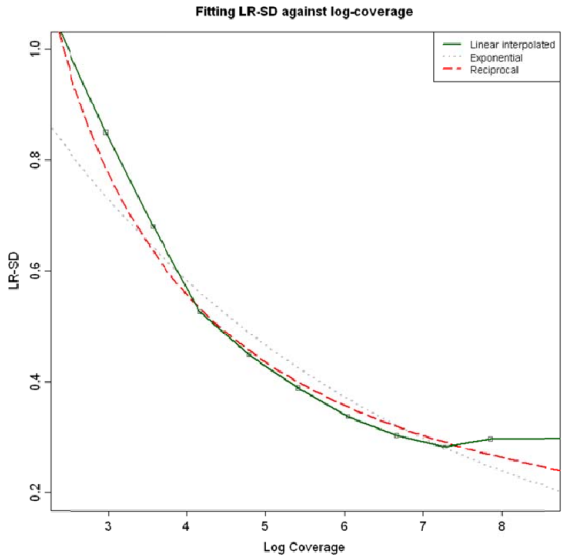

Figura 6.4: Ajuste lineal entre la media de los sd log-ratios dentro de cada bin, frente a la media del logaritmo de la cobertura ajustada del caso y el control dentro cada bin

mentación.

A partir de la resolución más basta (regiones más grandes), CONTRA aplica tres criterios para llamar a un segmento significativo:

(I) Si el segmento tiene un log-ratio $> 0,3$ tiene ganancia en el número de copias, o $< 0,3$ tiene pérdidas en el número de copias.

(II) Al menos la mitad de las regiones cubiertas por el segmento debe haber sido significativa en las predicciones de la CNVs.

(III) Y la dirección de la CNV (es decir, pérdida o ganancia) debe ser consistente entre las regiones que abarque.

Hay que mencionar que en las aplicaciones que hemos manejado a lo largo del proyecto, la detención de CNVs que abarcan varias regiones no ha sido necesaria.

Capítulo 7

Conclusión

Durante mi estancia en Gradiant, he tomado contacto con el proyecto GRIDD (desarrollado en Gradiant con la colaboración del CHUS). El punto de partida ha sido el algoritmo CONTRA, que se puede obtener del repositorio https://sourceforge.net/projects/contra-cnv/.

Conforme avanzaba el proyecto, se hizo palpable la necesidad de validar el algoritmo CONTRA, ya que es la base del proyecto. Contábamos con un número limitado de muestras reales de las que desconocíamos la presencia de todas las CNVs y de alteraciones distintas a las CNVs; así que para poder validar la fiabilidad de las herramientas, necesitábamos ampliar la cantidad de muestras, y optamos por dos posibles soluciones. La primera fue intentar conseguir muestras de las bases de datos públicas como HapMap (https://hapmap.ncbi.nlm.nih.gov/) o 1000 genomes (http://www.1000genomes.org/); pero no conseguimos reunir las muestras necesarias, por motivos de formato y por motivos de información disponible al respecto de las muestras. Y la segunda solución fue intentar simular nuestras propias muestras; para lo cual realizamos un análisis de los diversos simuladores disponibles hasta la fecha, pero no conseguimos una herramienta 100 % funcional ni que nos permitiera simular la casuística completa que plantea el contexto de detección de CNVs.

Por otro lado, el feedback que obtuvimos del CHUS nos indicaba que CONTRA no estaba detectando las anomalías de forma adecuada, debido a que, como vimos en el capítulo anterior, algunos de los planteamientos de este algoritmo no son adecuados.

Sumado a estos problemas, nos encontramos que el simulador utilizado en el artículo de CONTRA para generar muestras no goza de soporte, y que las alteraciones simuladas con esta herramienta generan perfiles poco parecidos a los que se obtendrían en la realidad, y con comportamientos anormales en las coberturas.

Por estos motivos este trabajo vuelve a la base de los experimentos NGS, e intenta construir una herramienta desde cero, que permita solucionar los problemas encontrados, obtener una herramienta fiable para su uso, y extender la capacidad de detección a la de anomalías de hasta una única base y la de simular muestras.

En este trabajo se presenta toda la base teórica necesaria para poder construir un software adecuado para la detección de anomalías en los perfiles de cobertura a nivel de base. En el futuro se pretende desarrollar este software en un paquete de R con el fin de obtener una herramienta lista para su prueba en un entorno real.

Apéndice A

Código de R de los scripts utilizados y datos

Los scripts de R referenciados en esta memoria así como los archivos de datos reales y simulados se pueden descargar en:

https://drive.google.com/open?id=0ByUqtq9myp8qVGs4UzBlVktHR3c.

Para hacer la memoria autcontenida, reproducimos a continuación los scripts, excepto *contra.R* por su longitud.

Script *contraste.R* (página 19):

```
################################
#                              #
#     CONTRASTE R CONSTANTE    #
#                              #
################################

###################
#nombres de archivo
#regrealnormal312.txt
#regrealnormal312.csv
#L_CTE.txt
#L_CTE_2.txt
#L_CTE.txt
#L_CTE_delhomo.txt
#L_CTE_delhetero.txt
#L_CTE_duphomo.txt
#L_CTE_duphetero.txt

#carpeta de trabajo
setwd("C:/Users/Jackie/Downloads/muestras/")

####################
#archivo de cobertura
datoscov=read.table("regrealnormal312.txt")

####################
#archivo con longitud de reads y posición inicial
datos=read.table("Read_length.txt")

#comportamiento de la longitud de reads R
hist(datos$V3,breaks=7)
mean(datos$V3)
median(datos$V3)
var(datos$V3)
length(datos$V3)

###################
#obtener lambda experimental
data=read.csv2("regrealnormal312.csv")
x=table(data[,1])
b=max(data[,1])-length(x)
lambda=mean(c(x,rep(0,b)))

###################
#representación de la cobertura de la región
plot(datoscov[1:nrow(datoscov),],type="l",ylim=c(0,300))

###################
#parámetros para calcular los esperados
#longitud de la region
```

```
longitud_region=length(datoscov[,1])

#longitud mediana de los reads de la region
longitud_media_read=median(data[,2])

#numero de reads
numero_reads=length(data$LONG)

#altura de la cobertura en el centro de la región
inicio_centro=longitud_media_read+1
final_centro=longitud_region-longitud_media_read

#cobertura de la región
cobertura_media=mean(datoscov[inicio_centro:final_centro,])

#forma alternativa para obtener el parámetro lambda
#lambda=(datoscov/longitud_media_read)


################################################################################
#calculo de los valores esperados con R constante
esperadoRCTE=function(reglength=634,readlength=100,coverage=120,lambda=1){

  R=readlength
  f=reglength
  b=seq(1,f,1)
  r=numeric(f)
  r=rep(lambda,f)
  c=numeric(f)

  for (b in 1:f){
    if(b<=1){
      c[b]=r[b]}

    if(b>1){
      ini=max(1,b-R+1)
      fin=min(b,f-R+1)
      c[b]=sum(r[ini:fin])}

  }

  lines(c,type="l",col=3)
  return(c)
}

c=esperadoRCTE(longitud_region,longitud_media_read,cobertura_media,lambda)
################################################################################

###################
#leyenda
legend(longitud_region/2-40, cobertura_media/2+1, c("Esperados"), text.col = c(3))

###################
#obtener p-valores

#nivel alfa
alfa=0.05

#valores observados
con=c(datoscov[,1])

#p-valores
pval2=ppois(con,c,lower.tail =T)#p-valores si h_1 < h_0 (p.ej. H1: Lambda=0)
#pval2=ppois(con,c,lower.tail =F)#p-valores si h_1 > h_0 (p.ej. H1: Lambda=Lambda*2)

#error tipo I (alpha)/potencia
mean(as.numeric(pval2<=alfa))

#p-valores ajustados BH
pval = p.adjust(pval2, method="BH")

#representacion de los p-valores
par(mfcol=c(1,2))
plot(pval2,type="l")
abline(h=.05,lty=2,col=2)
plot(pval,type="l")
abline(h=.05,lty=2,col=2)

###################
#forma alterantiva del calculo de p-valores

#probabilidades P[x=X]
probs2=round(dpois(con,c),3)
#probabilidades P[x<X]
probs=round(ppois(con,c,lower.tail=T),3)

#valor más extremo que se puede alcanazar sin rechazar
s=round(max(probs[probs<=alfa]),4)
#s=0
t=qpois(s,c)

#constante gamma
k=t
gam=(alfa-(s))/(t)

for(o in 1:length(gam)){
  if(gam[o]==Inf){
    gam[o]=0}
  if(gam[o]==-Inf){
    gam[o]=0}
}
por=cbind(con,k,gam)
```

```
#bases en las que se rechaza
nor=numeric()
for(u in 1:length(por[,1])){
  if(por[u,1]<por[u,2]){
    nor=rbind(nor,por[u,])}
}
nor

#bases en las que decidimos si se rechaza o no
nor2=numeric()
for(u in 1:length(por[,1])){
  if(por[u,1]==por[u,2]){
    nor2=rbind(nor2,por[u,])}
}
nor2

#decisión
rbinom(1,1,nor2[1,3]*nor2[1,2])
rbinom(1,1,nor2[2,3]*nor2[2,2])
rbinom(1,1,nor2[3,3]*nor2[3,2])
#.....
```

Script *contraste_R_aleatorio.R* (página 52):

```
#################################
#                               #
#     CONTRASTE R ALEATORIO     #
#                               #
#################################

###################
#nombres de archivo
#regrealnormal312.txt
#regrealnormal312.csv
#L_CTE.txt
#L_CTE_2.txt
#L_CTE.txt
#L_CTE_delhomo.txt
#L_CTE_delhetero.txt
#L_CTE_duphomo.txt
#L_CTE_duphetero.txt

#carpeta de trabajo
setwd("C:/Users/Jackie/Downloads/muestras/")

####################
#archivo de cobertura
datoscov=read.table("regrealnormal312.txt")

####################
#archivo con longitud de reads y posición inicial
datos=read.table("Read_length.txt")

#comportamiento de la longitud de reads R
hist(datos$V3,breaks=7)
mean(datos$V3)
median(datos$V3)
var(datos$V3)
length(datos$V3)
h=datos$V3

###################
#obtener lambda experimental
data=read.csv2("regrealnormal312.csv")
x=table(data[,1])
b=max(data[,1])-length(x)
lambda=mean(c(x,rep(0,b)))

###################
#representación de la cobertura de la región
plot(datoscov[1:nrow(datoscov),],type="l",ylim=c(0,300))

###################
#parámetros para calcular los esperados
#longitud de la region
longitud_region=length(datoscov[,1])

#longitud mediana de los reads de la region
longitud_media_read=median(data[,2])

#numero de reads
numero_reads=length(data$LONG)

#altura de la cobertura en el centro de la región
inicio_centro=longitud_media_read+1
final_centro=longitud_region-longitud_media_read

#cobertura de la región
cobertura_media=mean(datoscov[inicio_centro:final_centro,])

#forma alternativa para obtener el parámetro lambda
#lambda=(datoscov/longitud_media_read)

#################################################################
#calculo de los valores esperados con R aleatorio

esperadonoparFn=function(reglength=634,readlength=100,coverage=120,lambda=1){
  #probabilidades de la no paramétrica
  Fn=ecdf(h)
  summary(Fn)
  #cobertura esperada
  b=seq(1,reglength,1)
```

```
  f=reglength
  c=numeric(reglength)
  for (b in 1:f){
    if(b<=1){
      i=1
      y=Fn(f-b+1)
      c[b]=sum(y*lambda)}
    if(b>1){
      i=seq(1,b-1,1)
      x=numeric(b-1)
      for(i in 1:b-1){
        x[i]=(1-Fn(b-i))-(1-Fn(f-i+1))
        y=Fn(f-b+1) }
      c[b]=sum(x*lambda,y*lambda)}}
  lines(c,type="l",col=3)
  return(c)
}

c=esperadonoparFn(longitud_region,longitud_media_read,cobertura_media,lambda)
#####################################################################

###################
#leyenda
legend(longitud_region/2-40, cobertura_media/2+1, c("Esperados"), text.col = c(3))

###################
#obtener p-valores

#nivel alfa
alfa=0.05

#valores observados
con=c(datoscov[,1])

#p-valores
pval2=ppois(con,c,lower.tail =T)#p-valores si h_1 < h_0 (p.ej. H1: Lambda=0)
#pval2=ppois(con,c,lower.tail =F)#p-valores si h_1 > h_0 (p.ej. H1: Lambda=Lambda*2)

#error tipo I (alpha)/potencia
mean(as.numeric(pval2<=alfa))

#p-valores ajustados BH
pval = p.adjust(pval2, method="BH")

#representacion de los p-valores
par(mfcol=c(1,2))
plot(pval2,type="l")
abline(h=.05,lty=2,col=2)
plot(pval,type="l")
abline(h=.05,lty=2,col=2)

###################
#forma alterantiva del calculo de p-valores

#probabilidades P[x=X]
probs2=round(dpois(con,c),3)
#probabilidades P[x<X]
probs=round(ppois(con,c,lower.tail=T),3)

#valor más extremo que se puede alcanazar sin rechazar
s=round(max(probs[probs<=alfa]),4)
#s=0
t=qpois(s,c)

#constante gamma
k=t
gam=(alfa-(s))/(t)

for(o in 1:length(gam)){
  if(gam[o]==Inf){
    gam[o]=0}
  if(gam[o]==-Inf){
    gam[o]=0}
}
por=cbind(con,k,gam)

#bases en las que se rechaza
nor=numeric()
for(u in 1:length(por[,1])){
  if(por[u,1]<por[u,2]){
    nor=rbind(nor,por[u,])}
}
nor

#bases en las que decidimos si se rechaza o no
nor2=numeric()
for(u in 1:length(por[,1])){
  if(por[u,1]==por[u,2]){
    nor2=rbind(nor2,por[u,])}
}
nor2

#decisión
rbinom(1,1,nor2[1,3]*nor2[1,2])
rbinom(1,1,nor2[2,3]*nor2[2,2])
rbinom(1,1,nor2[3,3]*nor2[3,2])
#.....
```

Script *contraste_global.R* (página 48):

```
################################
#                              #
```

```
#       CONTRASTE GLOBAL        #
#                               #
#################################

#parámetros para la simulación

#carpeta de trabajo
setwd("C:/Users/Jackie/Downloads/muestras/")

#longitud de la region
longitud_region=312

#longitud mediana de los read de la regián
longitud_media_read=101

#cobertura central
cobertura_media=130

#lambda experimental
lambda=1.254717

#réplicas
B=100

#nivel de significación
alfa=0.05

############################################
#simulador con reads de longitud (Constante)
############################################

#alteraciones
#Normal---------------0
#DEL HOMOCIGOSIS------1
#DEL HETEROCIGOSIS----2
#DUP HOMOCIGOSIS------3
#DUP HETEROCIGOSIS----4

simulCTE=function(readlen=100,regionlen=1000,coverage=60,lambda="No",repe=10,
                  alter=0,start=NULL,end=NULL){

  if(lambda!="No"){
    coverage=lambda
  }
  if(lambda=="No"){
    readlen=readlen
    coverage=(coverage/readlen)
  }

  if(alter==4){
    coverage=coverage/2
  }

  for(j in 1:repe){
    readlen=readlen
    readsbase=rpois(regionlen,coverage)

    longreads=rep(readlen,sum(readsbase))

    y=0
    for(t in 1:length(readsbase)){
      re=rep(t,readsbase[t])
      y=c(y,re)}
    y=y[-1]
    datos=data.frame(y,longreads)

    #parámetros
    RLR=regionlen
    L=datos[,2]
    #posición de inicio de cada read
    x=datos[,1]
    #número de reads
    N=length(x)

    #matriz con los reads
    m=matrix(0,nrow=N,ncol=max(L))
    for (i in 1:length(x)){
      m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
    head(m)

    #colocamos los reads en su posición
    s=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=datos[,1]
    for (r in 1:N){
      if(numcerosinic[r]<=1){
        s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
      if(numcerosinic[r]>1){
        if(L[r]+numcerosinic[r]-1<=RLR){
          s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}

    #DEL HOMO
    if(alter==1){
      for(i in 1:length(s[,1])){

        if(sum(s[i,start:end])>0){
          p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

          if(sum(p[(RLR-(end-start)):RLR])>0){
            p=rep(0,RLR)}

          if(sum(p[(RLR-(end-start)):RLR])==0){
            p=p[1:RLR]}
```

```
      s[i,]=p}
  }}

#DEL HETERO
if(alter==2){
  for(i in 1:length(s[,1])){

    if(sum(s[i,start:end])>0){
      m=rbinom(1,1,0.5)

      if(m==1){p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

      if(sum(p[(RLR-(end-start)):RLR])>0){
        p=rep(0,RLR)}
      if(sum(p[(RLR-(end-start)):RLR])==0){
        p=p[1:RLR]}}


      if(m==0){p=s[i,]}
      s[i,]=p}
  }}

#DUP HOMO
if(alter==3){
  readsbase=rpois(regionlen+(end-start),coverage)
  longreads=rep(readlen,sum(readsbase))
  y=0
  for(t in 1:length(readsbase)){
    re=rep(t,readsbase[t])
    y=c(y,re)}
  y=y[-1]
  datos=data.frame(y,longreads)

  #parámetros
  RLR=regionlen+(end-start)
  L=datos[,2]
  #posición de inicio de cada read
  x=datos[,1]
  #número de reads
  N=length(x)

  #matriz con los reads
  m=matrix(0,nrow=N,ncol=max(L))
  for (i in 1:length(x)){
    m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
  head(m)

  #colocamos los reads en su posición
  s=matrix(0,nrow=N,ncol=RLR)
  numcerosinic=datos[,1]
  for (r in 1:N){
    if(numcerosinic[r]<=1){
      s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
    if(numcerosinic[r]>1){
      if(L[r]+numcerosinic[r]-1<=RLR){
        s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
  o=numeric(regionlen)
  for(k in 1:length(s[,1])){
    p=c(s[k,1:end],rep(0,(regionlen-end)))
    q=c(rep(0,start-1),s[k,end:RLR])
    o=rbind(o,p,q)
  }
  s=o
}

#DUP HETERO
if(alter==4){
  readsbase=rpois(regionlen+(end-start),coverage)

  longreads=rep(readlen,sum(readsbase))
  y=0
  for(t in 1:length(readsbase)){
    re=rep(t,readsbase[t])
    y=c(y,re)}
  y=y[-1]
  datos=data.frame(y,longreads)

  #parámetros
  RLR=regionlen+(end-start)
  L=datos[,2]
  #posición de inicio de cada read
  x=datos[,1]
  #número de reads
  N=length(x)

  #matriz con los reads
  m=matrix(0,nrow=N,ncol=max(L))
  for (i in 1:length(x)){
    m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
  head(m)

  #colocamos los reads en su posición
  se=matrix(0,nrow=N,ncol=RLR)
  numcerosinic=datos[,1]
  for (r in 1:N){
    if(numcerosinic[r]<=1){
      se[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
    if(numcerosinic[r]>1){
      if(L[r]+numcerosinic[r]-1<=RLR){
        se[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
  op=numeric(regionlen)
  for(k in 1:length(se[,1])){
    p=c(se[k,1:end],rep(0,(regionlen-end)))
```

```
        q=c(rep(0,start-1),se[k,end:RLR])
        op=rbind(op,p,q)
      }
      s=rbind(op,s)
    }


    u=colSums(s,dims=1L)
    #calculamos la cobertura de cada base
    if(j==1){
      c=rbind(u)}
    if(j>1){c=rbind(c,u)}

  }
  return(c)
}

u=simulCTE(readlen=longitud_media_read,regionlen=longitud_region,
           coverage=cobertura_media,lambda=lambda,repe=B,alter=0,start=1,end=312)

#####################################################################
# calculo de los esperados
#####################################################################
esperadoRCTE=function(reglength=634,readlength=100,coverage=120,lambda=1){
  #esperado
  R=readlength
  f=reglength
  b=seq(1,f,1)
  r=numeric(f)
  r=rep(lambda,f)
  c=numeric(f)

  for (b in 1:f){
    if(b<=1){
      c[b]=r[b]}

    if(b>1){
      ini=max(1,b-R+1)
      fin=min(b,f-R+1)
      c[b]=sum(r[ini:fin])}
  }
  return(c)
}

c=esperadoRCTE(reglength=longitud_region,readlength=longitud_media_read,
                    coverage=cobertura_media,lambda=lambda)
#####################################################################

#densidad del estadístico
q=numeric(0)
for(i in 1:B){
q[i]=sum(u[i,])/sum(c)}
plot(table(q))
hist(q)

#intervalo de confianza Montecarlo
estadistico_boot_ordenado=sort(q)
indice_inf=floor(B*alfa/2)
pto_crit_inf=estadistico_boot_ordenado[indice_inf]
indice_sup=floor(B*(1-alfa/2))
pto_crit_sup=estadistico_boot_ordenado[indice_sup]

limite_inf=pto_crit_sup
limite_sup=pto_crit_inf
IC<-c(limite_inf,limite_sup)
IC

#contraste normalidad
shapiro.test(q)

#archivo de cobertura real
datoscov=read.table("regrealnormal312.txt")
#resultado del test
t1=sum(datoscov)/sum(c)

############################
#Comprobar método
############################
#alteraciones
#Normal---------------0
#DEL HOMOCIGOSIS------1
#DEL HETEROCIGOSIS----2
#DUP HOMOCIGOSIS------3
#DUP HETEROCIGOSIS----4

#Inicio y final de la alteración
start=1
end=120
alter=4

alt=simulCTE(readlen=longitud_media_read,regionlen=longitud_region,
           coverage=cobertura_media,lambda=lambda,repe=B,alter=alter,start=start,end=end)
plot(alt[1,],type="l")

t2=numeric(0)
for(i in 1:B){
  t2[i]=sum(alt[i,])/sum(c)}

t3=mean(as.numeric(t2>IC[1]))

IC;t3
```

Script *densidad_reads.R* (página 47):

```
################################
#                              #
#       DENSIDAD DE READS      #
#                              #
################################

#carpeta de trabajo
setwd("C:/Users/Jackie/Downloads/muestras/")

#datos de reads
datos=read.table("Read_length.txt")
is.numeric(datos$V3)
datos=datos[datos$V3>=96,]
max(datos$V3)
hist(datos$V3,breaks=7)
median=median(datos$V3)
vari=var(datos$V3)
length(datos$V3)

#comapración con distitnas densidades paramétricas
x=seq(96,median,1)
j=ppois(x, median)
k=pgamma(x, shape=(median)^2/vari,scale=(vari)/(median))
l=pnorm(x,median,vari)
m=punif(x,96,median)

v=cbind(x,j,k,l,m)

n=rmultinom(5000,length(x),v[,2])
#n=rmultinom(5000,length(x),v[,3])
#n=rmultinom(5000,length(x),v[,4])
#n=rmultinom(5000,length(x),v[,5])
s=rowSums(n)
v2=cbind(x,s)
y=0
for(t in 1:length(v2[,1])){
  re=rep(v2[,1],v2[,2])
  y=c(y,re)}
y=y[-1]
hist(y,breaks=4)
h=datos$V3

ks.test(y,h,alternative = c("two.sided"))
```

Script *DF.R* (página 56) :

```
############################################
#                                          #
#       DATOS FUNCIONALES R COSNTANTE      #
#                                          #
############################################

#relación de la muestra de datos.csv

#1A 734 sim 1-734 DEHE
#2A 734 sim 1-734 DUHO
#3A 734 sim 130-170 DEHO
#4A 734 sim 130-170 DEHE
#5A 734 sim 130-170 DUHO
#6A 734 sim 130-170 DUHE
#7N 734 sim
#8N 734 sim
#9N 734 sim
#10N 734 sim
#11N 734 sim
#12N 734 sim
#13N 734 sim
#14N 734 sim
#15N 734 sim
#16N 734 sim
#17N 734 sim
#18N 734 sim
#19N 734 sim
#20N 734 sim
#21N 734 sim

#relación de la muestra de datos2.csv
#1N 312 real
#2:10N 312 sim

#carpeta de trabajo
setwd("C:/Users/Jackie/Downloads/muestras/")

#archivo de cobertura
datos=read.csv2("datos2.csv",header = T)


#####################################################################
#Datos funcionales
#####################################################################
library(fda.usc)

#tamaño de la región
reglen=734
#reglen=312

ini=1
fin=ncol(datos)
head(datos)
```

```
tail(datos)
t=seq(1,reglen,1)#puntos de discretización
perf=fdata(t(datos[,ini:fin]),argvals=t)

plot.fdata(perf,main="perfil")

#espacio L2

D2 = metric.lp(perf) #Distance L2 between curves
crit2 = apply(D2^2, 1, sum)
media=which.min(crit2);media
plot.fdata(perf[media,],col=4)

#mediana
crit=apply(D2,1,sum)
mediana=which.min(crit);mediana
plot.fdata(perf[mediana,],col=4)

plot(perf,lty=1,col="gray50")
lines(func.mean(perf),col=2,lwd=2)#media hilbertiana
lines(perf[mediana],col=2,lty=2,lwd=2)
legend("topleft", c("media", "mediana"), lwd = 2, lty = 1:2,col = 2)

fmd = depth.FM(perf)
md = depth.mode(perf)
rpd = depth.RP(perf, nproj = 10)
rtd = depth.RT(perf)
print(cur <- c(fmd$lmed, md$lmed, rpd$lmed, rtd$lmed))
md2 = depth.mode(perf, h = md$hq * 2)
plot(md$dep, md2$dep, pch = 19, col = 2, xlab = "MD(h)", ylab = "MD(2*h)")

plot(perf, type = "n")
lines(perf[cur], lwd = 2, lty = 1:4, col = 1:4)
legend("topleft", c("FMD", "MD", "RPD", "RTD"), lwd = 2, lty = 1:4,col = 1:4)


#dispersión
barx = func.mean(perf)
print(vsn <- mean(metric.lp(perf, barx)))

#bandas bootstrap
par(mfrow = c(1, 3))
m1b = fdata.bootstrap(perf, statistic = func.mean, nb = 100,
                      smo = 0.1, draw = TRUE,alpha=0.05)
m2b = fdata.bootstrap(perf, statistic = func.trim.mode,
                      nb = 100, smo = 0.1, draw = TRUE)
m3b = fdata.bootstrap(perf, statistic = func.med.mode,
                      nb = 100, smo = 0.1, draw = TRUE)

bw.ucv(perf$data[media,])
bw.SJ(perf$data[media,])
bw.nrd0(perf$data[media,])
bw.nrd0(perf$data[media,])

out1 = fdata.bootstrap(perf, statistic = func.mean,
                       nb = 100,draw = TRUE)

#outliers distitnas profundidades
out.FM = outliers.depth.trim(perf, nb = 100, smo = 0.1, trim = 0.03,dfunc = depth.FM)
out.RP = outliers.depth.pond(perf, nb = 100, smo = 0.1, dfunc = depth.RP)
out.hMw = outliers.depth.pond(perf, nb = 100, smo = 0.1, dfunc = depth.mode)
out.hMt = outliers.depth.trim(perf, nb = 100, smo = 0.1, trim = 0.03,dfunc = depth.mode)
#outlier

md1 = depth.mode(perf);min(md1$dep)
sort(md1$dep)
mc = c("black", "red", "blue")
plot(perf, type = "l", col = "gray", lty = 1)
lines(c(md1$median), col = mc, lwd = 2,lty = 1)
lines(c(md1$mtrim), col = mc, lwd = 2,lty = 2)
lines(perf[4,],col = 2, lwd = 2, lty = 3)
legend("topright", c("Deepest", "T_25%", "Out"), lty = 1:3, lwd = 2)
```

Script *errores_contraste.R* (página 40):

```
##############################################
#                                            #
#       ERRORES CONTRASTE R CONSTANTE        #
#                                            #
##############################################

#parámetros
#longitud de la región
longitud_region=312

#longitud mediana de los read de la región
longitud_media_read=101

#cobertura
cobertura_media=130

#lambda experimental
lambda=1.254717

#repeticiones
B=5000

##############################################
#simulador con reads de longitud (Constante)
##############################################

#alteraciones
```

```
#Normal---------------0
#DEL HOMOCIGOSIS------1
#DEL HETEROCIGOSIS----2
#DUP HOMOCIGOSIS------3
#DUP HETEROCIGOSIS----4

simulCTE=function(readlen=100,regionlen=1000,coverage=60,lambda="No",repe=10,
                  alter=0,start=NULL,end=NULL){

  if(lambda!="No"){
    coverage=lambda
  }
  if(lambda=="No"){
    readlen=readlen
    coverage=(coverage/readlen)
  }

  if(alter==4){
    coverage=coverage/2
  }

  for(j in 1:repe){
    readlen=readlen
    readsbase=rpois(regionlen,coverage)

    longreads=rep(readlen,sum(readsbase))

    y=0
    for(t in 1:length(readsbase)){
      re=rep(t,readsbase[t])
      y=c(y,re)}
    y=y[-1]
    datos=data.frame(y,longreads)

    RLR=regionlen
    L=datos[,2]
    x=datos[,1]
    N=length(x)

    m=matrix(0,nrow=N,ncol=max(L))
    for (i in 1:length(x)){
      m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
    head(m)

    s=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=datos[,1]
    for (r in 1:N){
      if(numcerosinic[r]<=1){
        s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
      if(numcerosinic[r]>1){
        if(L[r]+numcerosinic[r]-1<=RLR){
          s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}

    #DEL HOMO
    if(alter==1){
      for(i in 1:length(s[,1])){

        if(sum(s[i,start:end])>0){
          p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

          if(sum(p[(RLR-(end-start)):RLR])>0){
            p=rep(0,RLR)}

          if(sum(p[(RLR-(end-start)):RLR])==0){
            p=p[1:RLR]}

          s[i,]=p}
      }}

    #DEL HETERO
    if(alter==2){
      for(i in 1:length(s[,1])){

        if(sum(s[i,start:end])>0){
          m=rbinom(1,1,0.5)

          if(m==1){p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

          if(sum(p[(RLR-(end-start)):RLR])>0){
            p=rep(0,RLR)}
          if(sum(p[(RLR-(end-start)):RLR])==0){
            p=p[1:RLR]}}


          if(m==0){p=s[i,]}
          s[i,]=p}
      }}

    #DUP HOMO
    if(alter==3){
      readsbase=rpois(regionlen+(end-start),coverage)
      longreads=rep(readlen,sum(readsbase))
      y=0
      for(t in 1:length(readsbase)){
        re=rep(t,readsbase[t])
        y=c(y,re)}
      y=y[-1]
      datos=data.frame(y,longreads)

      RLR=regionlen+(end-start)
      L=datos[,2]
      x=datos[,1]
      N=length(x)
```

```
      m=matrix(0,nrow=N,ncol=max(L))
      for (i in 1:length(x)){
        m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
      head(m)

      s=matrix(0,nrow=N,ncol=RLR)
      numcerosinic=datos[,1]
      for (r in 1:N){
        if(numcerosinic[r]<=1){
          s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
        if(numcerosinic[r]>1){
          if(L[r]+numcerosinic[r]-1<=RLR){
            s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
      o=numeric(regionlen)
      for(k in 1:length(s[,1])){
        p=c(s[k,1:end],rep(0,(regionlen-end)))
        q=c(rep(0,start-1),s[k,end:RLR])
        o=rbind(o,p,q)
      }
      s=o
    }

    #DUP HETERO
    if(alter==4){
      readsbase=rpois(regionlen+(end-start),coverage)

      longreads=rep(readlen,sum(readsbase))
      y=0
      for(t in 1:length(readsbase)){
        re=rep(t,readsbase[t])
        y=c(y,re)}
      y=y[-1]
      datos=data.frame(y,longreads)

      RLR=regionlen+(end-start)
      L=datos[,2]
      x=datos[,1]
      N=length(x)

      m=matrix(0,nrow=N,ncol=max(L))
      for (i in 1:length(x)){
        m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
      head(m)

      se=matrix(0,nrow=N,ncol=RLR)
      numcerosinic=datos[,1]
      for (r in 1:N){
        if(numcerosinic[r]<=1){
          se[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
        if(numcerosinic[r]>1){
          if(L[r]+numcerosinic[r]-1<=RLR){
            se[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
      op=numeric(regionlen)
      for(k in 1:length(se[,1])){
        p=c(se[k,1:end],rep(0,(regionlen-end)))
        q=c(rep(0,start-1),se[k,end:RLR])
        op=rbind(op,p,q)
      }
      s=rbind(op,s)
    }


    u=colSums(s,dims=1L)
    if(j==1){
      c=rbind(u)}
    if(j>1){c=rbind(c,u)}

  }
  return(c)
}

u=simulCTE(readlen=longitud_media_read,regionlen=longitud_region,
           coverage=cobertura_media,lambda=lambda,repe=B,alter=0,start=130,end=170)


#################################################################
#esperados
#################################################################
esperadonoparCTE=function(reglength=634,readlength=100,coverage=120,lambda=1){

  #esperado
  R=readlength
  f=reglength
  b=seq(1,f,1)
  r=numeric(f)
  r=rep(lambda,f)
  c=numeric(f)

  for (b in 1:f){
    if(b<=1){
      c[b]=r[b]}

    if(b>1){
      ini=max(1,b-R+1)
      fin=min(b,f-R+1)
      c[b]=sum(r[ini:fin])}

  }

  return(c)
}
```

```
c=esperadonoparCTE(reglength=longitud_region,readlength=longitud_media_read,
                   coverage=cobertura_media,lambda=lambda)
#################################################################

h=numeric(ncol(u))
t=numeric(ncol(u))

for(o in 1:ncol(u)){
  pval2=ppois(u[,o],c[o],lower.tail =T)#p-valores si h_1 < h_0 (p.ej. H1: Lambda=0)
  #pval2=ppois(con,c,lower.tail =F)#p-valores si h_1 > h_0 (p.ej. H1: Lambda=Lambda*2)

  pval=p.adjust(pval2, method="BH")

  h[o]=mean(as.numeric(pval2<=0.05))#error tipo I (alpha) en las simulaciones
  t[o]=mean(as.numeric(pval<=0.05))
}


par(mfcol=c(1,2))
plot(h,type="l")
abline(h=.05,lty=2,col=2)
plot(t,type="l")
abline(h=.05,lty=2,col=2)
```

Script *IC.R* (página 23):

```
################################################
#                                              #
#       INTERVALO CONFIANZA R CONSTANTE        #
#                                              #
################################################

###################
#nombres de archivo
#regrealnormal312.txt
#regrealnormal312.csv
#L_CTE.txt
#L_CTE_2.txt
#L_CTE.txt
#L_CTE_delhomo.txt
#L_CTE_delhetero.txt
#L_CTE_duphomo.txt
#L_CTE_duphetero.txt
#L_Fn_R_alea.txt


#carpeta de trabajo
setwd("C:/Users/Jackie/Downloads/muestras/")

#######################################################
#Datos de la región
#######################################################
#archivo de cobertura
datoscov=read.table("regrealnormal312.txt")

#archivo de longitud de reads y posición
datosl=read.table("Read_length.txt")
hist(datosl$V3,breaks=7)
mean(datosl$V3)
median(datosl$V3)
var(datosl$V3)
length(datosl$V3)
h=datosl$V3


###########
#obtener lambda
data=read.csv2("regrealnormal312.csv")
x=table(data[,1])
b=max(data[,1])-length(x)
lambda=mean(c(x,rep(0,b)))

#tamaño mediano de los read
read_long=median(datos$V3)

#tamaño de la region
longitud_region=length(datoscov[,1])

#numero de reads
numero_reads=length(data$LONG)

#altura en el centro de la región
inicio_centro=read_long+1
final_centro=longitud_region-read_long
cobertura_media=mean(datoscov[inicio_centro:final_centro,])


#representación de la región
plot(datoscov[1:nrow(datoscov),],type="l",ylim=c(0,180))

#repeticiones
b=1000

#nivel alfa
alfa=0.05


#####################################################
#Simulacion de densidad de reads (Constante)
#####################################################
simuldensCTE=function(readlen=100,regionlen=2141,coverage=265,repe=10,alfa=0.05){
  coverage=(coverage/readlen)
```

```
  for(j in 1:repe){
    readsbase=rpois(regionlen,coverage)
    #longreads=rpois(sum(readsbase),readlen)
    longreads=rep(readlen,sum(readsbase))
    y=0
    for(t in 1:length(readsbase)){
      re=rep(t,readsbase[t])
      y=c(y,re)}
    y=y[-1]
    datos=data.frame(y,longreads)
    #parámetros
    RLR=regionlen
    L=datos[,2]
    #posición de inicio de cada read
    x=datos[,1]
    #número de reads
    N=length(x)
    #matriz con los reads
    m=matrix(0,nrow=N,ncol=max(L))
    for (i in 1:length(x)){
      m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
    head(m)
    #colocamos los reads en su posición
    s=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=datos[,1]
    for (r in 1:N){
      if(numcerosinic[r]<=1){
        s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
      if(numcerosinic[r]>1){
        if(L[r]+numcerosinic[r]-1<=RLR){
          s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
    u=colSums(s,dims=1L)
    #calculamos la cobertura de cada base
    if(j==1){
      c=rbind(u)}
    if(j>1){c=rbind(c,u)}
    #nos quedamos con la región
    #u=u[u>0]
  }
  IC=numeric()
  for(k in 1:ncol(c)){
    ordenado=sort(c[,k])
    indice_inf=floor(repe*alfa/2)
    pto_crit_inf=ordenado[indice_inf]
    indice_sup=floor(repe*(1-alfa/2))
    pto_crit_sup=ordenado[indice_sup]
    IC_b=c(pto_crit_inf,pto_crit_sup)
    IC<-cbind(IC,IC_b)}
  return(IC)}

h=simuldensCTE(readlen=read_long,regionlen=longitud_region,coverage=coverage,repe=1000)
lines(h[2,],type="l",col=3)
lines(h[1,],col=2)

######################################################

#leyenda
legend(longitud_region/2-50, cobertura_media/2+15, c("Limite sup","limite inf"), text.col = c(3,2))
```

Script *IC R_aleatorio.R* (página 50):

```
################################################
#                                              #
#       INTERVALO CONFIANZA R ALEATORIO        #
#                                              #
################################################

###################
#nombres de archivo
#regrealnormal312.txt
#regrealnormal312.csv
#L_CTE.txt
#L_CTE_2.txt
#L_CTE.txt
#L_CTE_delhomo.txt
#L_CTE_delhetero.txt
#L_CTE_duphomo.txt
#L_CTE_duphetero.txt
#L_Fn_R_alea.txt


#carpeta de trabajo
setwd("C:/Users/Jackie/Downloads/muestras/")

######################################################
#Datos de la región
######################################################
#archivo de cobertura
datoscov=read.table("L_Fn_R_alea.txt")

#archivo de longitud de reads y posición
datosl=read.table("Read_length.txt")
hist(datosl$V3,breaks=7)
mean(datosl$V3)
median(datosl$V3)
var(datosl$V3)
length(datosl$V3)
h=datosl$V3


###########
#obtener lambda
```

```
data=read.csv2("regrealnormal312.csv")
x=table(data[,1])
b=max(data[,1])-length(x)
lambda=mean(c(x,rep(0,b)))

#tamaño mediano de los read
read_long=median(datos$V3)

#tamaño de la region
longitud_region=length(datoscov[,1])

#numero de reads
numero_reads=length(data$LONG)

#altura en el centro de la región
inicio_centro=read_long+1
final_centro=longitud_region-read_long
cobertura_media=mean(datoscov[inicio_centro:final_centro,])


#representación de la región
plot(datoscov[1:nrow(datoscov),],type="l",ylim=c(0,180))

#repeticiones
b=1000

#nivel alfa
alfa=0.05


#########################################################
#Simulacion de densidad de reads (Fn)
#########################################################
simuldensfn=function(regionlen=2141,coverage=150,repe=10,alfa=0.05){
  readlen=mean(datosl$V3)
  coverage=(coverage/readlen)
  for(j in 1:repe){
    readsbase=rpois(regionlen,coverage)
    muestra<-c(datosl$V3)
    n=sum(readsbase)
    remuestra<-numeric(n)
    for (s in 1:n){
      o=runif(1)
      w=floor(o*n)+1
      remuestra[s]=muestra[w]}
    longreads=remuestra
    y=0
    for(t in 1:length(readsbase)){
      re=rep(t,readsbase[t])
      y=c(y,re)}
    y=y[-1]
    datos=data.frame(y,longreads)
    #parámetros
    RLR=regionlen
    L=datos[,2]
    #posición de inicio de cada read
    x=datos[,1]
    #número de reads
    N=length(x)
    #matriz con los reads
    m=matrix(0,nrow=N,ncol=max(L))
    for (i in 1:length(x)){
      m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
    head(m)
    #colocamos los reads en su posición
    s=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=datos[,1]
    for (r in 1:N){
      if(numcerosinic[r]<=1){
        s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
      if(numcerosinic[r]>1){
        if(L[r]+numcerosinic[r]-1<=RLR){
          s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
    u=colSums(s,dims=1L)
    #calculamos la cobertura de cada base
    if(j==1){
      c=rbind(u)}
    if(j>1){c=rbind(c,u)}
    #nos quedamos con la región
    #u=u[u>0]
  }
  #intervalo
  IC=numeric()
  for(k in 1:ncol(c)){
    ordenado=sort(c[,k])
    indice_inf=floor(repe*alfa/2)
    pto_crit_inf=ordenado[indice_inf]
    indice_sup=floor(repe*(1-alfa/2))
    pto_crit_sup=ordenado[indice_sup]
    IC_b=c(pto_crit_inf,pto_crit_sup)
    IC<-cbind(IC,IC_b)}
  return(IC)}

h=simuldensfn(regionlen=longitud_region,coverage=cobertura_media,repe=B,alfa=alfa)

lines(h[2,],type="l",col=3)
lines(h[1,],col=2)
#########################################################

#leyenda
legend(longitud_region/2-50, cobertura_media/2+15, c("Limite sup","limite inf"), text.col = c(3,2))
```

Script *Simulador.R* (página 20):

```
################################
#                              #
#     Simulador R constante    #
#                              #
################################

#######################################
#simulador con reads de longitud (Constante)
#######################################

#alteraciones
#Normal---------------0
#DEL HOMOCIGOSIS------1
#DEL HETEROCIGOSIS----2
#DUP HOMOCIGOSIS------3
#DUP HETEROCIGOSIS----4

simulCTE=function(readlen=100,regionlen=1000,coverage=60,lambda="No",repe=10,
                  alter=0,start=NULL,end=NULL){
  if(lambda!="No"){
    coverage=lambda
  }
  if(lambda=="No"){
    readlen=readlen
    coverage=(coverage/readlen)
  }

  if(alter==4){
    coverage=coverage/2
  }

  for(j in 1:repe){
    readlen=readlen
    readsbase=rpois(regionlen,coverage)

    longreads=rep(readlen,sum(readsbase))

    y=0
    for(t in 1:length(readsbase)){
      re=rep(t,readsbase[t])
      y=c(y,re)}
    y=y[-1]
    datos=data.frame(y,longreads)

    #parámetros
    RLR=regionlen
    L=datos[,2]
    #posición de inicio de cada read
    x=datos[,1]
    #número de reads
    N=length(x)

    #matriz con los reads
    m=matrix(0,nrow=N,ncol=max(L))
    for (i in 1:length(x)){
      m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
    head(m)

    #colocamos los reads en su posición
    s=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=datos[,1]
    for (r in 1:N){
      if(numcerosinic[r]<=1){
        s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
      if(numcerosinic[r]>1){
        if(L[r]+numcerosinic[r]-1<=RLR){
          s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}

    #DEL HOMO
    if(alter==1){
      for(i in 1:length(s[,1])){

        if(sum(s[i,start:end])>0){
          p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

          if(sum(p[(RLR-(end-start)):RLR])>0){
            p=rep(0,RLR)}

          if(sum(p[(RLR-(end-start)):RLR])==0){
            p=p[1:RLR]}

          s[i,]=p}
      }}

    #DEL HETERO
    if(alter==2){
      for(i in 1:length(s[,1])){

        if(sum(s[i,start:end])>0){
          m=rbinom(1,1,0.5)

          if(m==1){p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

          if(sum(p[(RLR-(end-start)):RLR])>0){
            p=rep(0,RLR)}
          if(sum(p[(RLR-(end-start)):RLR])==0){
            p=p[1:RLR]}}


          if(m==0){p=s[i,]}
          s[i,]=p}
```

```
    }}

  #DUP HOMO
  if(alter==3){
    readsbase=rpois(regionlen+(end-start),coverage)
    longreads=rep(readlen,sum(readsbase))
    y=0
    for(t in 1:length(readsbase)){
      re=rep(t,readsbase[t])
      y=c(y,re)}
    y=y[-1]
    datos=data.frame(y,longreads)

    #parámetros
    RLR=regionlen+(end-start)
    L=datos[,2]
    #posicion de inicio de cada read
    x=datos[,1]
    #numero de reads
    N=length(x)

    #matriz con los reads
    m=matrix(0,nrow=N,ncol=max(L))
    for (i in 1:length(x)){
      m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
    head(m)

    #colocamos los reads en su posición
    s=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=datos[,1]
    for (r in 1:N){
      if(numcerosinic[r]<=1){
        s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
      if(numcerosinic[r]>1){
        if(L[r]+numcerosinic[r]-1<=RLR){
          s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
    o=numeric(regionlen)
    for(k in 1:length(s[,1])){
      p=c(s[k,1:end],rep(0,(regionlen-end)))
      q=c(rep(0,start-1),s[k,end:RLR])
      o=rbind(o,p,q)
    }
    s=o
  }

  #DUP HETERO
  if(alter==4){
    readsbase=rpois(regionlen+(end-start),coverage)

    longreads=rep(readlen,sum(readsbase))
    y=0
    for(t in 1:length(readsbase)){
      re=rep(t,readsbase[t])
      y=c(y,re)}
    y=y[-1]
    datos=data.frame(y,longreads)

    #parámetros
    RLR=regionlen+(end-start)
    L=datos[,2]
    #posicion de inicio de cada read
    x=datos[,1]
    #número de reads
    N=length(x)

    #matriz con los reads
    m=matrix(0,nrow=N,ncol=max(L))
    for (i in 1:length(x)){
      m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
    head(m)

    #colocamos los reads en su posición
    se=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=datos[,1]
    for (r in 1:N){
      if(numcerosinic[r]<=1){
        se[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
      if(numcerosinic[r]>1){
        if(L[r]+numcerosinic[r]-1<=RLR){
          se[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
    op=numeric(regionlen)
    for(k in 1:length(se[,1])){
      p=c(se[k,1:end],rep(0,(regionlen-end)))
      q=c(rep(0,start-1),se[k,end:RLR])
      op=rbind(op,p,q)
    }
    s=rbind(op,s)
  }

  #calculamos la cobertura de cada base
  u=colSums(s,dims=1L)

  #nos quedamos con la región
  #u=u[u>0]

  if(j==1){
    plot(u,type="l",ylim=c(0,200))
  }
  if(j>1){lines(u,col=2)
  }
}
if(alter==0){
  out.f="L_CTE.txt"}
```

```
  if(alter==1){
    out.f="L_CTE_delhomo.txt"}
  if(alter==2){
    out.f="L_CTE_delhetero.txt"}
  if(alter==3){
    out.f="L_CTE_duphomo.txt"}
  if(alter==4){
  out.f="L_CTE_duphetero.txt"}

  #directorio de salida
  outf="C:\\Users\\Jackie\\Downloads\\"
  setwd(outf)
  write.table(u,out.f,sep = "\t", row.names = F,col.names = F)
  return(u)
}

u=simulCTE(readlen=101,regionlen=312,coverage=130,repe=1000,
          alter=0,start=130,end=170)
```

Script *Simulador_2.R* (página 22):

```
################################
#                              #
#     Simulador R aleatorio    #
#                              #
################################

#####################################################################
#Simulacion alternativa reads(Fn)
#####################################################################

#alteraciones
#Normal---------------0
#DEL HOMOCIGOSIS------1
#DEL HETEROCIGOSIS----2
#DUP HOMOCIGOSIS------3
#DUP HETEROCIGOSIS----4

rlrsimulator=function(readlen=101,N=500,RLR=1000,repe=1,
                      alter=0,start=NULL,end=NULL){
  if(alter==4){
    N=N/2
  }

  for(j in 1:repe){
    readlen=readlen
    longreads= rep(readlen,N)

    m=matrix(0,nrow=N,ncol=max(longreads))
    for (i in 1:length(longreads)){
      m[i,]=m[i,]+c(rep(1,longreads[i]),rep(0,max(longreads)-longreads[i]))
    }
    head(m)

    s=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=round(runif(N,0,RLR))

    for (r in 1:N){
      if(length(m[r,])+numcerosinic[r]>=RLR)
        s[r,]=c(rep(0,RLR))
      else(
        s[r,]=c(rep(0,numcerosinic[r]),m[r,],rep(0,RLR-(length(m[r,])+numcerosinic[r]))))
    }
    #DEL HOMO
    if(alter==1){
      for(i in 1:length(s[,1])){

        if(sum(s[i,start:end])>0){
          p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

          if(sum(p[(RLR-(end-start)):RLR])>0){
            p=rep(0,RLR)}

          if(sum(p[(RLR-(end-start)):RLR])==0){
            p=p[1:RLR]}

          s[i,]=p}
      }}

    #DEL HETERO
    if(alter==2){
      for(i in 1:length(s[,1])){

        if(sum(s[i,start:end])>0){
          m=rbinom(1,1,0.5)

          if(m==1){p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

          if(sum(p[(RLR-(end-start)):RLR])>0){
            p=rep(0,RLR)}
          if(sum(p[(RLR-(end-start)):RLR])==0){
            p=p[1:RLR]}}


          if(m==0){p=s[i,]}
          s[i,]=p}
      }}

    #DUP HOMO
    if(alter==3){

      longreads= rep(readlen,N)
```

```
      #parámetros
      RLR2=RLR+(end-start)

      m=matrix(0,nrow=N,ncol=max(longreads))
      for (i in 1:length(longreads)){
        m[i,]=m[i,]+c(rep(1,longreads[i]),rep(0,max(longreads)-longreads[i]))
      }
      head(m)

      #colocamos los reads en su posición
      s=matrix(0,nrow=N,ncol=RLR2)
      numcerosinic=round(runif(N,0,RLR2))

      for (r in 1:N){
        if(length(m[r,])+numcerosinic[r]>=RLR2)
          s[r,]=c(rep(0,RLR2))
        else(
          s[r,]=c(rep(0,numcerosinic[r]),m[r,],rep(0,RLR2-(length(m[r,])+numcerosinic[r]))))
      }
      o=numeric(RLR)
      for(k in 1:length(s[,1])){
        p=c(s[k,1:end],rep(0,(RLR-end)))
        q=c(rep(0,start-1),s[k,end:RLR2])
        o=rbind(o,p,q)
      }
      s=o
    }

    #DUP HETERO
    if(alter==4){
      longreads= rep(readlen,N)

      #parámetros
      RLR2=RLR+(end-start)

      m=matrix(0,nrow=N,ncol=max(longreads))
      for (i in 1:length(longreads)){
        m[i,]=m[i,]+c(rep(1,longreads[i]),rep(0,max(longreads)-longreads[i]))
      }
      head(m)

      se=matrix(0,nrow=N,ncol=RLR2)
      numcerosinic=round(runif(N,0,RLR2))

      for (r in 1:N){
        if(length(m[r,])+numcerosinic[r]>=RLR2)
          se[r,]=c(rep(0,RLR2))
        else(
          se[r,]=c(rep(0,numcerosinic[r]),m[r,],rep(0,RLR2-(length(m[r,])+numcerosinic[r]))))
      }
      op=numeric(RLR)
      for(k in 1:length(se[,1])){
        p=c(se[k,1:end],rep(0,(RLR-end)))
        q=c(rep(0,start-1),se[k,end:RLR2])
        op=rbind(op,p,q)
      }
      s=rbind(op,s)
    }

    z=colSums(s,dims=1L)

    if(j==1){
      plot(z,type="l")
    }
    if(j>1){lines(z,col=2)
    }
  }
  #nombre del archivo de salida
  out.f="L_CTE_2.txt"
  #directorio de salida
  outf="C:\\Users\\Jackie\\Downloads\\"
  setwd(outf)
  write.table(z,out.f,sep = "\t", row.names = F,col.names = F)
}

z=rlrsimulator(readlen=101,N=500,RLR=312,repe=1,
               alter=4,start=130,end=170)
```

Script *Simulador R_aleatorio.R* (página 48):

```
#################################
#                               #
#     Simulador R aleatorio     #
#                               #
#################################

#######################################
#simulador con reads de longitud aleatoria (Fn)
#######################################
datosl=read.table("C:\\Users\\Jackie\\Downloads\\muestras\\Read_length.txt")
hist(datosl$V3,breaks=7)

#alteraciones
#Normal---------------0
#DEL HOMOCIGOSIS------1
#DEL HETEROCIGOSIS----2
#DUP HOMOCIGOSIS------3
#DUP HETEROCIGOSIS----4

simulFn=function(readlen=100,regionlen=1000,coverage=60,lambda="No",repe=10,
                 alter=0,start=NULL,end=NULL){
```

```
if(lambda!="No"){
 coverage=lambda
}
if(lambda=="No"){
readlen=median(datosl$V3)
coverage=(coverage/readlen)
}

if(alter==4){
  coverage=coverage/2
}

for(j in 1:repe){
  readlen=mean(datosl$V3)
  readsbase=rpois(regionlen,coverage)
  muestra<-c(datosl$V3)
  n=sum(readsbase)
  remuestra<-numeric(n)
  for (s in 1:n){
    o=runif(1)
    w=floor(o*n)+1
    remuestra[s]=muestra[w]}
  longreads=remuestra
  y=0
  for(t in 1:length(readsbase)){
    re=rep(t,readsbase[t])
    y=c(y,re)}
  y=y[-1]
  datos=data.frame(y,longreads)

  #parámetros
  RLR=regionlen
  L=datos[,2]
  #posición de inicio de cada read
  x=datos[,1]
  #número de reads
  N=length(x)

  #matriz con los reads
  m=matrix(0,nrow=N,ncol=max(L))
  for (i in 1:length(x)){
    m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
  head(m)

  #colocamos los reads en su posición
  s=matrix(0,nrow=N,ncol=RLR)
  numcerosinic=datos[,1]
  for (r in 1:N){
    if(numcerosinic[r]<=1){
      s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
    if(numcerosinic[r]>1){
      if(L[r]+numcerosinic[r]-1<=RLR){
        s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}

  #DEL HOMO
  if(alter==1){
    for(i in 1:length(s[,1])){

      if(sum(s[i,start:end])>0){
        p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

        if(sum(p[(RLR-(end-start)):RLR])>0){
          p=rep(0,RLR)}

        if(sum(p[(RLR-(end-start)):RLR])==0){
          p=p[1:RLR]}

        s[i,]=p}
    }}

  #DEL HETERO
  if(alter==2){
    for(i in 1:length(s[,1])){

      if(sum(s[i,start:end])>0){
        m=rbinom(1,1,0.5)

        if(m==1){p=c(s[i,1:start],rep(0,end-start),s[i,end:RLR])

        if(sum(p[(RLR-(end-start)):RLR])>0){
          p=rep(0,RLR)}
        if(sum(p[(RLR-(end-start)):RLR])==0){
          p=p[1:RLR]}}


        if(m==0){p=s[i,]}
        s[i,]=p}
    }}

  #DUP HOMO
  if(alter==3){
  readsbase=rpois(regionlen+(end-start),coverage)
  n=sum(readsbase)
  remuestra<-numeric(n)
  for (s in 1:n){
    o=runif(1)
    w=floor(o*n)+1
    remuestra[s]=muestra[w]}
  longreads=remuestra
  y=0
  for(t in 1:length(readsbase)){
    re=rep(t,readsbase[t])
    y=c(y,re)}
```

```
    y=y[-1]
    datos=data.frame(y,longreads)

    #parámetros
    RLR=regionlen+(end-start)
    L=datos[,2]
    #posición de inicio de cada read
    x=datos[,1]
    #número de reads
    N=length(x)

    #matriz con los reads
    m=matrix(0,nrow=N,ncol=max(L))
    for (i in 1:length(x)){
      m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
    head(m)

    #colocamos los reads en su posición
    s=matrix(0,nrow=N,ncol=RLR)
    numcerosinic=datos[,1]
    for (r in 1:N){
      if(numcerosinic[r]<=1){
        s[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
      if(numcerosinic[r]>1){
        if(L[r]+numcerosinic[r]-1<=RLR){
          s[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
    o=numeric(regionlen)
    for(k in 1:length(s[,1])){
      p=c(s[k,1:end],rep(0,(regionlen-end)))
      q=c(rep(0,start-1),s[k,end:RLR])
      o=rbind(o,p,q)
    }
    s=o
    }

    #DUP HETERO
    if(alter==4){
      readsbase=rpois(regionlen+(end-start),coverage)
      n=sum(readsbase)
      remuestra<-numeric(n)
      for (ss in 1:n){
        o=runif(1)
        w=floor(o*n)+1
        remuestra[ss]=muestra[w]}
      longreads=remuestra
      y=0
      for(t in 1:length(readsbase)){
        re=rep(t,readsbase[t])
        y=c(y,re)}
      y=y[-1]
      datos=data.frame(y,longreads)

      #parámetros
      RLR=regionlen+(end-start)
      L=datos[,2]
      #posición de inicio de cada read
      x=datos[,1]
      #número de reads
      N=length(x)

      #matriz con los reads
      m=matrix(0,nrow=N,ncol=max(L))
      for (i in 1:length(x)){
        m[i,]=m[i,]+c(rep(1,L[i]),rep(0,max(L)-L[i]))}
      head(m)

      #colocamos los reads en su posición
      se=matrix(0,nrow=N,ncol=RLR)
      numcerosinic=datos[,1]
      for (r in 1:N){
        if(numcerosinic[r]<=1){
          se[r,]=c(rep(1,L[r]),rep(0,RLR-L[r]))}
        if(numcerosinic[r]>1){
          if(L[r]+numcerosinic[r]-1<=RLR){
            se[r,]=c(rep(0,numcerosinic[r]-1),rep(1,L[r]),rep(0,RLR-L[r]-numcerosinic[r]+1))}}}
      op=numeric(regionlen)
      for(k in 1:length(se[,1])){
        p=c(se[k,1:end],rep(0,(regionlen-end)))
        q=c(rep(0,start-1),se[k,end:RLR])
        op=rbind(op,p,q)
      }
      s=rbind(op,s)
    }

    #calculamos la cobertura de cada base
    u=colSums(s,dims=1L)

    #nos quedamos con la region
    #u=u[u>0]

    if(j==1){
      plot(u,type="l")
    }
    if(j>1){lines(u,col=2)
    }
  }
  out.f="L_Fn_R_alea.txt"
  #directorio de salida
  outf="C:\\Users\\Jackie\\Downloads\\"
  setwd(outf)
  write.table(u,out.f,sep = "\t", row.names = F,col.names = F)
  return(u)
}
```

```
u=simulFn(readlen=101,regionlen=312,coverage=131,repe=1,
          alter=0,start=130,end=170)
```

Bibliografía

[1] Aird D., Ross MG., Chen W-S., et al. (2011) Analyzing and minimizing PCR amplification bias in Illumina sequencing libraries. Genome Biology. 12(2): R18.

[2] Benjamini Y., Hochberg Y. (1995). Controlling the false discovery rate: a practical and powerful approach to multiple testing. J. R. Stat. Soc. 57: 289-300.

[3] Benjamini Y., Yekutieli D. (2001). The control of the false discovery rate in multiple testing under dependency. Annals of Statistics 29 (4): 1165-1188.

[4] Benjamini Y., Speed T. (2011) Estimation and correction for GC-content bias in high throughput sequencing. Tech Rep 804 Department of Statistics, University of California, Berkeley; 2011.

[5] Boeva V., et al. (2011) Control-free calling of copy number alterations in deep-sequencing data using GC-content normalization. Bioinformatics 2011, 27(2): 268-269.

[6] M. Burrows, D. J. Wheeler (1994), A block sorting lossless data compression algorithm. Digital Equipment Corporation Technical Report. 124.

[7] Joaquin Cañizares, Jose Miguel Blanca. 2015. Grado de Biotecnologia. http://personales.upv.es/jcanizar/bioinformatica/mapeo.html

[8] Conrad TM, et al.(2009). Whole-genome resequencing of Escherichia coli K-12 MG1655 undergoing short-term laboratory evolution in lactate minimal media reveals flexible selection of adaptive mutations. Genome Biol 10.)

[9] Cuevas, A., Febrero, M., and Fraiman, R. (2007). Robust estimation and classification for functional data via projection-based depth notions. Computational Statistics, 22(3): 481-496

[10] Manuel Febrero Bande et al. (2016). Functional Data Analysis and Utilities for Statistical Computing. Package "fda.usc".

[11] Hayya, Jack; Armstrong, Donald; Gressis, Nicolas (1975). A Note on the Ratio of Two Normally Distributed Variables. Management Science 21(11): 1338-1341.

[12] Lander ES., Waterman MS. (1988) Genomic mapping by fingerprinting random clones: a mathematical analysis, Genomics 2(3): 231-239.

[13] Li H., Durbin R. (2009) Fast and accurate short read alignment with Burrows-Wheeler transform. Bioinformatics. 25(14):1754-60.

[14] Li H., Handsaker B., Wysoker A., et al. (2009). The Sequence alignment/map (SAM) format and SAMtools. Bioinformatics, 25(16), 2078-2079.

[15] Li J., Lupat R., Amarasinghe KC., et al. (2012) CONTRA: copy number analysis for targeted resequencing. Bioinformatics. 28(10): 1307-1313.

[16] Rebekah L. Rogers, Julie M. Cridland, Ling Shao, Tina T. Hu, Peter Andolfatto and Kevin R. (2014). Landscape of Standing Variation for Tandem Duplications in Drosophila yakuba and Drosophila simulans. Molecular Biology And Evolution.

[17] Jill M. Johnsen, Deborah A. Nickerson, Alex P. Reiner. Massively parallel sequencing: the new frontier of hematologic genomics Blood Nov 2013, 122 (19) 3268-3275.

[18] Olshen AB., Venkatraman ES., Lucito R., Wigler M. (2004). Circular binary segmentation for the analysis of array-based DNA copy number data. Biostatistics. 5(4): 557-72.

[19] Quinlan AR., Hall IM. (2010). BEDTools: a flexible suite of utilities for comparing genomic features. Bioinformatics. 26(6): 841-2.

[20] Robinson M.D., et al. (2010). edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics. 26: 139-140.

[21] Benjamin Rodriguez-Santiago y Lluis Armengol (2012). Tecnologias de secuenciacion de nueva generacion en diagnostico genetico pre y postnatal. Elsevier. 23(2): 56-66.

[22] Sathirapongsasuti JF., Lee H., Horst BAJ., et al. (2011). Exome sequencing-based copy-number variation and loss of heterozygosity detection: ExomeCNV. Bioinformatics. 27(19): 2648-2654.

[23] Yen-Chun Chen, Tsunglin Liu, Chun-Hui Yu, Tzen-Yuh Chiang, Chi-Chuan Hwang. (2013) Effects of GC Bias in Next-Generation-Sequencing Data on De Novo Genome Assembly, PLoS ONE. 8(4): e62856.

[24] Yoon, S., Xuan, Z., Makarov, V., Ye, K., and Sebat, J. (2009). Sensitive and accurate detection of copy number variants using read depth of coverage. Genome Research, 19, 1586-1592.