Universitat Oberta de Catalunya

uoc.edu

Comparación de diferentes métodos de “calling” de variantes somáticas: Mutect2, SomaticSniper, Varscan2 y Strelka2.

Sonia Mª Garrote Fernández
Máster en Bioinformática y Bioestadística
M0.178 - TFM-Bioinformática y Bioestadística Área 2

Nombre Consultor/a: Jaime Sastre Tomas
Nombre Profesor/a responsable de la asignatura: Marc Maceira Duch

Enero-2021

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Comparación de diferentes métodos de "calling" de variantes somáticas: Mutect2, SomaticSniper, Varscan2 y Strelka2*
Nombre del autor:	*Sonia Mª Garrote Fernández*
Nombre del consultor/a:	*Jaime Sastre Tomas*
Nombre del PRA:	*Marc Maceira Duch*
Fecha de entrega (mm/aaaa):	01/2021
Titulación:	*Máster en Bioinformática y Bioestadística*
Área del Trabajo Final:	*M0.178 - TFM-Bioinformática y Bioestadística Área 2*
Idioma del trabajo:	*Castellano*
Palabras clave	*Cáncer, genética, "Variant Calling".*

Resumen del Trabajo

Uno de los ámbitos de aplicación de la secuenciación de genomas es el relacionado con el estudio de los genomas del cáncer. El calling o variant calling es un proceso mediante el cual se pueden identificar variantes en las secuencias genéticas, lo que podemos aplicar a la detección de genes relacionados con el desarrollo y la progresión del cáncer. Existen métodos que permiten implementar este proceso, pero cada método cuenta con su algoritmo de búsqueda y en muchas ocasiones, el número de variantes encontradas es diferente. En este estudio nos centramos en el descubrimiento de variantes somáticas en secuencias tumorales, analizando los métodos y resultados que arrojan distintos programas: Mutect2, SomaticSniper, Varscan2 y Strelka2, utilizando el genoma de referencia hg38. Se efectúa un estudio individual de cada uno de los programas a comparar y de los métodos de calling que aplican, se prueban y se comparan con datos de secuencias de pares tumor-normal. Los resultados obtenidos fueron que Mutect2 no tuvo coincidencias con VarScan2, tuvo 17 coincidencias con SomaticSniper y 24 con Strelka2; VarScan2 tuvo solo 1

coincidencia con SomaticSniper en el Modo 1[1]; SomaticSniper tuvo 38 coincidencias con Strelka2 en el Modo 1 y 31 en el Modo 2. Como se ha podido observar el resultado de los diferentes programas de descubrimiento de variantes somáticas solo coincide en un subconjunto de variantes.

Abstract (in English, 250 words or less):

Genome sequencing has been extensively applied to the study of the cancer genomes. “Calling” or “Variant Calling” is a process used for the detection of the genetic variation, this could be applied to detection of genes responsible for the development and progression of cancer. There are some methods that implement this process. Each method has its own search algorithm, and it is usual that the results of one method differ from the other. In this study we focus on the discovery of somatic mutations in cancer genomes. The methods and results of some programs (Mutect2, SomaticSniper, Varscan2 y Strelka2) are analyzed. Hg38 reference genome is used. We made individual analysis of variant calling tools and the methods they apply. Doing Test and comparing those tools with tumor-normal samples. There were no coincidences between Mutect2 and VarScan2, there was 17 coincidences between Mutect2 and SomaticSniper, and 24 between Mutect2 and Strelka2; VarScan2 had only 1 coincidence with SomaticSniper in Mode 1; SomaticSniper had 38 coincidences with Strelka2 in Mode 1 and 31 in Mode 2. The results from the variant callers agree only on a subset of variants.

[1] Ver el punto 3.3.2 Modelo, para más información sobre los modos de ejecución de SomaticSniper

Índice

Lista de figuras

Lista de tablas

1 Introducción

1.1 Contexto y justificación del Trabajo

1.1.1 Descripción General

Los grandes avances en la tecnología utilizada para la secuenciación del genoma permiten una rápida, y relativamente barata, obtención de grandes conjuntos de secuencias disponibles para su análisis. Uno de los ámbitos de aplicación de la secuenciación de genomas es el relacionado con el estudio de los genomas del cáncer (1).

El *calling* o *variant calling* es un proceso mediante el cual se pueden identificar variantes en las secuencias genéticas analizadas. Las variantes en las secuencias genéticas se refieren a cambios en la secuencia de los nucleótidos que conforman los genes, pueden ser variaciones en nucleótidos individuales –*Single Nucleotid Variant* (SNV) / *Single Nucleotide Polimorfism*[2] (SNP)- (Figura 1), pueden ser inserciones o eliminaciones de unos pocos nucleótidos –*indel*- (Figura 2), grandes alteraciones de la secuencia de nucleótidos o reajustes estructurales de toda la secuencia – variaciones en más de 50 pares de bases que pueden ser eliminaciones, duplicados, inserciones, inversiones o combinaciones de estas- (Figura 3).

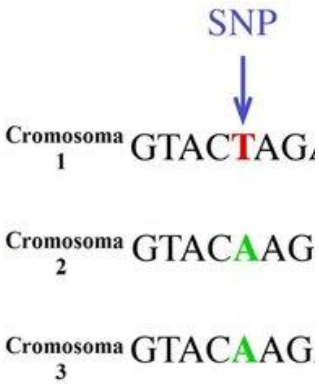


Figura 1. Polimorfismos de un solo nucleótido o SNPs -*single nucleotide polymorphisms*-

[2] Cuando la variación está presente en al menos el 1% de la población hablamos de SNP, sino hablamos de SNV.

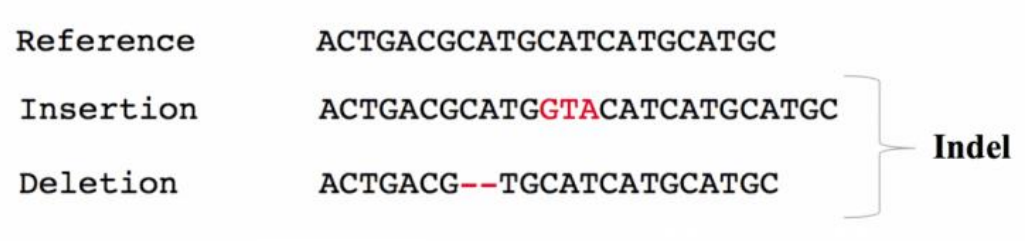


Figura 2. Indel. Inserción de la secuencia GTA y la eliminación de la secuencia CA

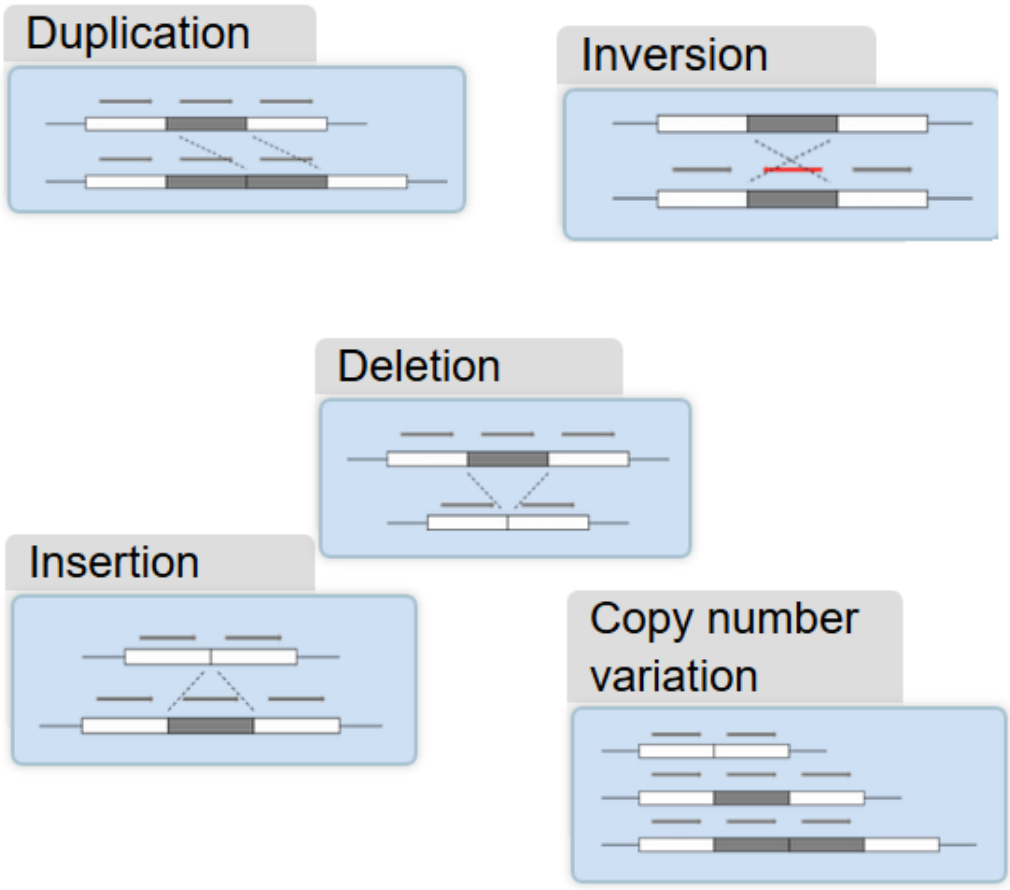


Figura 3. Los 5 tipos más comunes de variaciones estructurales en secuencias de genes

"Cuando los cambios genéticos se producen después del nacimiento se denominan mutaciones somáticas. Estas mutaciones o variantes somáticas se pueden presentar en cualquiera de las células del cuerpo, excepto las germinativas (esperma y óvulo). Los cambios de este tipo pueden causar cáncer u otras enfermedades, pero no siempre"[3], por eso, es importante diferenciar entre las variantes somáticas denominadas "passengers" y las denominadas "drivers". Las variantes de tipo

[3] https://www.cancer.gov/espanol/publicaciones/diccionario/def/mutacion-somatica

"passengers" no tienen efecto en el desarrollo del cáncer, mientras que las denominadas "drivers", pueden generar un tumor. Podemos aplicar el "calling" a la detección de genes relacionados con el desarrollo y la progresión del cáncer, lo que supone un avance importante en la diagnosis y el tratamiento de la enfermedad. Ya existen algunos métodos que permiten implementar este proceso y que se despliegan en sistemas computacionales que son capaces de manejar el gran volumen de datos de las secuencias de genes. El problema, es que cada método cuenta con su algoritmo de búsqueda y en muchas ocasiones, el número de variantes encontradas es diferente. (2)(3)

Por otro lado, la detección de variantes o mutaciones somáticas se puede ver influenciada, no sólo por los métodos que aplican los distintos programas de "calling" sino también por la profundidad media de la cobertura de secuenciación y la proporción de subclones patológicos mutados o frecuencia de las mutaciones. La profundidad media de la cobertura de secuenciación se puede definir teóricamente como LN / G, donde L es la longitud de lectura, N es el número de lecturas y G es la longitud del genoma haploide objeto de estudio (4). G, en el caso de estudios sobre el genoma humano, sería la longitud de los 23 cromosomas que conforman las células haploides del ser humano, que es aproximadamente 3.1GB. Los subclones patológicos mutados tienen que ver con la heterogeneidad de las células tumorales, en un determinado tumor no todas las células tumorales son idénticas de forma que a medida que las células se van dividiendo se producen variaciones o mutaciones en los nuevos clones(5).

En este estudio nos centramos en el descubrimiento de variantes somáticas en secuencias tumorales, en concreto variantes somáticas simples que se producen en nucleótidos individuales de tipo *SNVs* e *Indels*, analizando el funcionamiento y los resultados que arrojan distintos programas, que permiten efectuar el "calling" de las variantes en datos de secuencias de cáncer. En concreto se efectúa una comparativa de los programas: Mutect2 (6), SomaticSniper (7), Varscan2 (8) y Strelka2 (9),

utilizando el genoma de referencia en su versión de ensamblado GRChg/hg38[4] (GRChg: Genome Reference Consortium human genome), analizando sus diferencias.

Un ensamblado del genoma, sea la versión que sea, no es más que la secuenciación del genoma completo, una secuenciación con mayor o menor calidad, es decir que puede incluir huecos que no se han podido completar con la tecnología de secuenciación utilizada. No se trata de la secuenciación del genoma de un individuo, sino que cada segmento de un genoma de referencia se caracteriza por estar compuesto de la secuencia más comúnmente observada entre los genomas disponibles secuenciados. Los ensamblados del genoma de referencia actuales son de tipo haploide, es decir que representan solo una copia de cada cromosoma (*or contig*). El ensamblado más actual es el GRChg38, que se lanzó en diciembre de 2013, y utiliza *contigs* alternativos para representar variaciones comunes complejas o alternativas (*ALT contigs*). Estas variaciones, también están representadas en ensamblados anteriores como el hg37, pero en menor medida. La mayoría de las mejoras que incluye el ensamblado GRCh38 se deben a varios proyectos de secuenciación y análisis del genoma, incluido el 1000 Genomes Project (10). No solo el ensamblado GRCh38 incluye *contigs* alternativos, sino que corrige miles de pequeños problemas de secuenciación debidos a falsos SNPs e *indels* que había en ensamblados previos[5]. Por estas razones se selecciona el GRCh38 como genoma de referencia para este trabajo.

Se analiza la viabilidad de obtener los datos de secuencias de genes procedentes de repositorios como son los consorcios *The Cancer Genome Atlas* (TCGA)[6] y el *International Cancer Genome Consortium* (ICGC)[7] que disponen de la secuenciación de genomas completos de

[4] https://www.ncbi.nlm.nih.gov/grc/human
[5] https://gatk.broadinstitute.org/hc/en-us/articles/360035890951-Human-genome-reference-builds-GRCh38-or-hg38-b37-hg19
[6] https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga
[7] https://icgc.org/

pares de tumor y tejido normal. Finalmente, ante la imposibilidad de obtener los datos de estos repositorios, al ser de acceso restringido, se opta por la creación de muestras tumorales virtuales partiendo de secuencias procedentes del proyecto 1000 Genomes Project[8] (10).

1.1.2 Justificación del TFM

La razón por la cual se ha seleccionado este área y tema para el TFM tiene que ver con el área de conocimiento previo a la realización del Máster en Bioinformática y Bioestadística, de la autora del TFM, en este caso la informática. De este modo se consideró que se tenía más posibilidades de llevar a cabo un mejor trabajo de final de máster si se seleccionaba una temática relacionada con el uso o comparación de programas. Después de revisar las distintas áreas y subáreas de la oferta de TFMs, se seleccionó una de las propuestas del área 2, subárea 6, al ver que esta tenía como palabras clave asociadas: análisis de datos, integración de datos, programación y aplicaciones web.

1.2 Objetivos del Trabajo

I. Realizar un análisis de los métodos utilizados en varios programas (Mutect2, SomaticSniper, VarScan2 y Strelka2), capaces de realizar el calling de las variantes en datos de secuencias de cáncer utilizando el genoma de referencia hg38. Estudiar sus características y realizar pruebas de funcionamiento con muestras de ejemplo.

II. Realizar un estudio comparativo de las variantes que encuentra cada programa y ver las posibles diferencias entre ellas, si existen variantes coincidentes o no y reajustar los parámetros asociados a la ejecución de los análisis para ver cómo afectan a los resultados y la comparativa.

[8] https://www.internationalgenome.org/home

1.3 Enfoque y método seguido

Para poder comparar los distintos métodos de *calling* que aplican los programas a estudiar (Mutect2, SomaticSniper, Varscan2 y Strelka2) es necesario conocerlos individualmente y ponerlos a prueba analizando los resultados que arrojan. De esta forma se podrá efectuar una comparativa entre ellos. Es por ello por lo que para cumplir con los objetivos del trabajo se aplica la siguiente metodología:

- Estudio individual de cada uno de los programas a comparar: Mutect2, SomaticSniper, Varscan2 y Strelka2.
- Análisis de los métodos de *calling* que implementan.
- Prueba de los programas con datos de secuencias de cáncer utilizando el genoma de referencia hg38.
- Comparativa de los resultados obtenidos tras las pruebas.

1.4 Planificación del Trabajo

1.4.1 Planificación I

Antes de comenzar a trabajar en la PEC2 se llevó a cabo el primer plan de trabajo, el cual se muestra a continuación.

Recursos

Para llevar a cabo el trabajo se utilizarán las versiones de los programas evaluados Mutect2, SomaticSniper, Varscan2 y Strelka2, siendo todas de libre acceso. Fuentes posibles para la obtención de los conjuntos de datos de secuencias de genes: *The Cancer Genome Atlas Consortium* (TCGA), el *International Cancer Genome Consortium* (ICGC).

Tareas

PEC2

- Tarea 1. Evaluación individual de cada programa (Del 14/10/20 al 30/10/2020)

- Tarea 2. Selección del conjunto de datos a testear (Del 2/11/2020 al 4/11/2020)

- Tarea 3. Análisis individuales con cada programa sobre el mismo conjunto de datos de secuencias de genes que permitan la obtención de informes de resultados y gráficos (Del 5/11/2020 al 16/11/2020)

PEC3

- Tarea 4. Comparativa entre los distintos métodos que aplica cada programa y los resultados diferentes que arrojan (Del 17/11/2020 al 14/12/2020)

PEC4

- Tarea 5. Cierre de la memoria (Del 15/12/2020 al 5/01/2021)

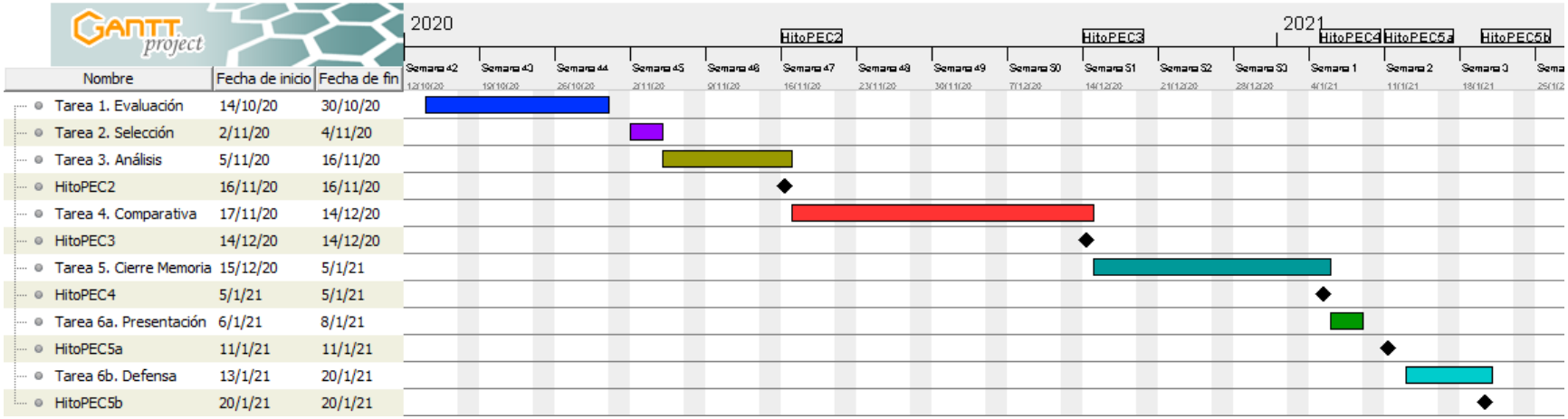
PEC5a

- Tarea 6. Elaboración de la presentación (Del 06/01/2021 al 8/01/2021)

PEC5b

- Tarea 7. Defensa (Del 13/01/2021 al 20/01/2021)

Calendario de tareas con hitos:



Nombre	Fecha de inicio	Fecha de fin
Tarea 1. Evaluación	14/10/20	30/10/20
Tarea 2. Selección	2/11/20	4/11/20
Tarea 3. Análisis	5/11/20	16/11/20
HitoPEC2	16/11/20	16/11/20
Tarea 4. Comparativa	17/11/20	14/12/20
HitoPEC3	14/12/20	14/12/20
Tarea 5. Cierre Memoria	15/12/20	5/1/21
HitoPEC4	5/1/21	5/1/21
Tarea 6a. Presentación	6/1/21	8/1/21
HitoPEC5a	11/1/21	11/1/21
Tarea 6b. Defensa	13/1/21	20/1/21
HitoPEC5b	20/1/21	20/1/21

Análisis de riesgos:

- Ajuste incorrecto en los tiempos asignados a la evaluación, selección de datos y análisis de los programas debido al desconocimiento previo de las dificultades que puede entrañar la comprensión de los métodos que aplican. (tareas 1, 2 y 3).

- Ajuste incorrecto en los tiempos asignados a la comparativa de métodos. (tarea 4).

1.4.2 Planificación II

Una vez comenzada la PEC2 y debido a varias causas que se detallan a continuación, la planificación inicial descrita en el apartado anterior (Planificación I) sufrió cambios.

Causas de las desviaciones en la planificación I.

- **Imposibilidad de acceder a los conjuntos de datos en las fuentes especificadas en los recursos de la planificación I:** *The Cancer Genome Atlas Consortium* (TCGA), el *International Cancer Genome Consortium* (ICGC). Los conjuntos de datos necesarios -ficheros BAM muestras normal y tumoral- son de acceso restringido.

- **Dificultades encontradas durante la instalación de herramientas asociadas al primer método analizado, mutect2.**

- **Dificultades encontradas en la comprensión asociada a los conceptos genéricos.**

Se indica a continuación la nueva planificación.

Recursos

Para llevar a cabo el trabajo se utilizarán las versiones de los programas evaluados Mutect2, SomaticSniper, Varscan2 y Strelka2, siendo todas de

libre acceso. Fuentes posibles para la obtención de los conjuntos de datos de secuencias de genes: Proyecto *1000 Genomes Project*.

Tareas

Se reajustan las tareas llevadas a cabo en la PEC2 y se modifican las de la PEC3 que pasan a absorber parte del trabajo que no ha sido posible realizar en la PEC2, por las causas antes citadas.

PEC2

- Tarea 1. Evaluación individual de mutect2 (Del 14/10/20 al 4/11/2020).
- Tarea 2. Análisis de fuentes posibles para la obtención del conjunto de datos, pares tumor-normal (Del 5/11/2020 al 16/11/2020).

PEC3

- Tarea 3. Generación de los pares tumor-normal de tipo virtual. (Del 16/11/2020 al 30/11/2020).
- Tarea 4. Análisis individuales con cada programa sobre el mismo conjunto de datos de secuencias de genes que permitan la obtención de informes de resultados y gráficos (Del 1/12/2020 al 7/12/2020).
- Tarea 5. Comparativa entre los distintos métodos que aplica cada programa y los resultados diferentes que arrojan (Del 7/12/2020 al 14/12/2020).

PEC4

- Tarea 6. Cierre de la memoria (Del 15/12/2020 al 5/01/2021).

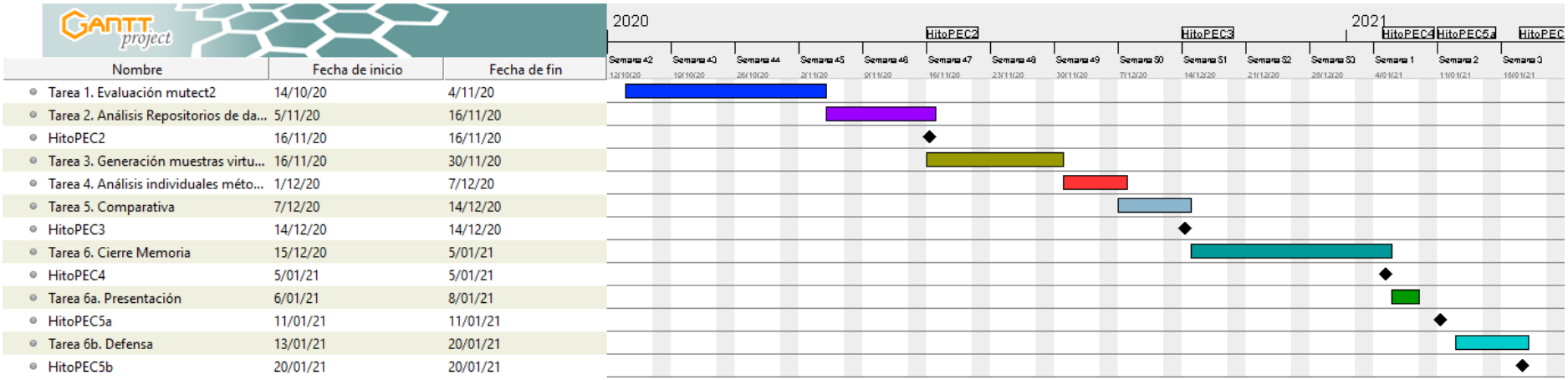
PEC5a

- Tarea 7a. Elaboración de la presentación (Del 06/01/2021 al 11/01/2021).

PEC5b

- Tarea 7b. Defensa (Del 13/01/2021 al 20/01/2021).

Calendario de tareas con hitos:



Nombre	Fecha de inicio	Fecha de fin
Tarea 1. Evaluación mutect2	14/10/20	4/11/20
Tarea 2. Análisis Repositorios de da...	5/11/20	16/11/20
HitoPEC2	16/11/20	16/11/20
Tarea 3. Generación muestras virtu...	16/11/20	30/11/20
Tarea 4. Análisis individuales méto...	1/12/20	7/12/20
Tarea 5. Comparativa	7/12/20	14/12/20
HitoPEC3	14/12/20	14/12/20
Tarea 6. Cierre Memoria	15/12/20	5/01/21
HitoPEC4	5/01/21	5/01/21
Tarea 6a. Presentación	6/01/21	8/01/21
HitoPEC5a	11/01/21	11/01/21
Tarea 6b. Defensa	13/01/21	20/01/21
HitoPEC5b	20/01/21	20/01/21

Análisis de riesgos:

- Ajuste incorrecto en los tiempos asignados a las tareas 3, 4 y 5 debido al desconocimiento previo de las dificultades que puede entrañar la generación de las muestras virtuales, la aplicación de las diferentes herramientas de *calling* y la posterior comparativa.

1.4.3 Planificación III

Una vez comenzada la PEC3 y después de las problemáticas encontradas con la obtención de las muestras virtuales tumor-normal para la realización de los análisis, y tras efectuar el estudio y comparativa de los programas VarScan2 y Mutect2 se tuvo que hacer un reajuste en la redacción de los objetivos y también en la Planificación II. La nueva planificación se detalla a continuación.

Recursos

Para llevar a cabo el trabajo se utilizan las versiones de los programas evaluados Mutect2, SomaticSniper, Varscan2 y Strelka2, siendo todas de libre acceso. La fuente utilizada para la obtención de las muestras de pares tumor-normal y el ensamblado de referencia hg38 son unos repositorios de ejemplo del instituto Broad[9].

Tareas

Se reajustan las tareas llevadas a cabo en la PEC3 y se modifican las de la PEC4, que pasan a absorber parte del trabajo que no ha sido posible realizar en la PEC3, por las causas antes citadas.

PEC2

[9] Muestras tumor-normal de ejemplo: https://drive.google.com/drive/folders/13RJCtKEhAXKCnLr-aQilFL1QId57bpHx. Ensamblado de referencia hg38: https://console.cloud.google.com/storage/browser/genomics-public-data/resources/broad/hg38/v0;tab=objects?prefix=&forceOnObjectsSortingFiltering=false

- Tarea 1. Evaluación individual de mutect2 (Del 14/10/20 al 4/11/2020).
- Tarea 2. Análisis de fuentes posibles para la obtención del conjunto de datos, pares tumor-normal (Del 5/11/2020 al 16/11/2020).

PEC3

- Tarea 3. Obtención de los pares tumor-normal. (Del 16/11/2020 al 30/11/2020).
- Tarea 4. Análisis individuales con Mutect2 y VarScan2 sobre el mismo conjunto de datos de secuencias de genes que permiten la obtención de informes de resultados (Del 1/12/2020 al 7/12/2020).
- Tarea 5. Comparativa entre los resultados obtenidos con Mutect2 y VarScan2 (Del 7/12/2020 al 14/12/2020).

PEC4

- Tarea 6. Análisis individual de SomaticSniper y Strelka2. (Del 15/12/2020 al 24/12/2020)
- Tarea 7. Comparativa entre los diferentes métodos. (Del 25/12/2020 al 31/12/2020)
- Tarea 8. Cierre de la memoria (Del 1/01/2020 al 5/01/2021).

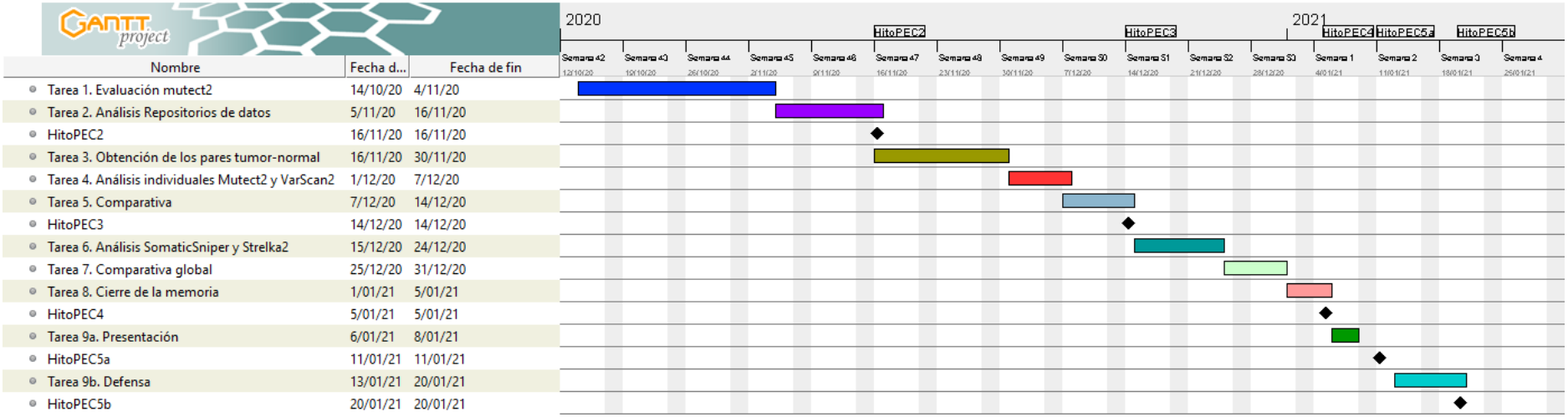
PEC5a

- Tarea 9a. Elaboración de la presentación (Del 06/01/2021 al 11/01/2021).

PEC5b

- Tarea 9b. Defensa (Del 13/01/2021 al 20/01/2021).

Calendario de tareas con hitos:



Nombre	Fecha d...	Fecha de fin
Tarea 1. Evaluación mutect2	14/10/20	4/11/20
Tarea 2. Análisis Repositorios de datos	5/11/20	16/11/20
HitoPEC2	16/11/20	16/11/20
Tarea 3. Obtención de los pares tumor-normal	16/11/20	30/11/20
Tarea 4. Análisis individuales Mutect2 y VarScan2	1/12/20	7/12/20
Tarea 5. Comparativa	7/12/20	14/12/20
HitoPEC3	14/12/20	14/12/20
Tarea 6. Análisis SomaticSniper y Strelka2	15/12/20	24/12/20
Tarea 7. Comparativa global	25/12/20	31/12/20
Tarea 8. Cierre de la memoria	1/01/21	5/01/21
HitoPEC4	5/01/21	5/01/21
Tarea 9a. Presentación	6/01/21	8/01/21
HitoPEC5a	11/01/21	11/01/21
Tarea 9b. Defensa	13/01/21	20/01/21
HitoPEC5b	20/01/21	20/01/21

1.5 Resultados esperados

Plan de trabajo e informes de seguimiento: en el plan de trabajo se definen las líneas generales del proyecto y su enfoque, los objetivos, la temporización y se definen y valoran los riesgos y posibles acciones de mitigación de estos. Este plan de trabajo se va reajustando y se documentan los cambios en los informes de seguimiento.

Memoria: la memoria incluye todos los capítulos que detallan todo el trabajo llevado a cabo.

Presentación virtual: es el documento utilizado para la exposición y defensa final del trabajo.

1.6 Breve descripción de los otros capítulos de la memoria

Capítulo 2. Selección de muestras para el testeo de los métodos de calling. En este capítulo se analizan distintas fuentes desde las que se pueden obtener muestras de secuencias de genes que posteriormente puedan ser introducidas como *input* en los programas de calling.

Capítulo 3. Estudio y análisis de programas y métodos de "calling". En este capítulo se abordan los siguientes puntos:

- Estudio individual de cada uno de los programas a comparar: Mutect2, SomaticSniper, Varscan2 y Strelka2.
- Análisis de los métodos de "calling" que implementan.
- Prueba de los programas con datos de secuencias de cáncer utilizando el genoma de referencia hg38.

  Los contenidos de este capítulo reflejan el trabajo desarrollado para cumplir con el primer objetivo (ver apartado 1.2 Objetivos del Trabajo)

Capítulo 4. Comparativa de los resultados obtenidos tras las pruebas. En este capítulo se muestra una comparativa de los resultados

obtenidos tras las pruebas efectuadas con los programas analizados en el capítulo 2. Los contenidos de este capítulo dan respuesta al objetivo número dos (ver apartado 1.2 Objetivos del Trabajo).

Capítulo 5. Conclusiones: en este capítulo se muestran las conclusiones del trabajo, se hace una reflexión crítica sobre la consecución de los objetivos, la planificación y la metodología y se plantean líneas de trabajo futuro.

Capítulo 6. Glosario: definición de términos y acrónimos relevantes utilizados en la memoria.

Capítulo 7. Bibliografía: listado de referencias bibliográficas utilizadas dentro de la memoria.

Capítulo 8. Anexos*:* se presentan varios anexos relacionados con procedimientos de instalación de herramientas de calling, funcionamiento de repositorios de conjuntos de datos de secuencias de genes, obtención de pares de muestras tumor-normal para efectuar la comparativa y ejecución de pruebas de los métodos de calling.

2 Selección de muestras para el testeo de los métodos de calling.

Primero necesitamos las muestras tumor-normal. Hacemos un estudio de posibles fuentes desde las que descargar dichas muestras. Probamos inicialmente con el repositorio disponible en *Genomic Data Commons* (GDC)[10]. Este repositorio de datos asociados a estudios genómicos en el cáncer es el repositorio que proporciona *The Cancer Genome Atlas (TCGA)[11].*

Tras llevar a cabo los pasos para acceder a las muestras, comprobamos que el acceso a las mismas está restringido y no es factible obtener la aprobación. En el Anexo II pueden verse los pasos que se llevaron a cabo

[10] https://gdc.cancer.gov/
[11] https://www.cancer.gov/about-nci/organization/ccg/research/structural-genomics/tcga

para acceder a los ficheros hasta el momento en el que se descubre que los ficheros de interés tienen acceso restringido.

Como segunda opción se decide consultar otro repositorio, en este caso el del *International Cancer Genome Consortium* -ICGC-[12]. Volvemos a encontrarnos con los mismos problemas de acceso restringido. En el Anexo III pueden verse los pasos ejecutados.

Como tercera alternativa nos planteamos generar nuestros propios pares tumor-normal virtuales utilizando como base secuencias extraídas del Proyecto 1000 Genomes Project[13]. Partiendo de una muestra normal simulamos mutaciones somáticas en determinados puntos (fracciones de alelos), sustituyendo determinadas lecturas por otras procedentes de otra muestra de forma que obtenemos la muestra de tipo tumor. Luego medimos la capacidad de detectar las mutaciones que tiene el método de *calling* que estemos aplicando. Esta alternativa nos causa varios inconvenientes asociados al alineamiento con el genoma de referencia.

Finalmente optamos por trabajar con unos conjuntos de datos de ejemplo disponibles en el instituto Broad[14].

En el Anexo IV se muestran los pasos llevados a cabo para la obtención de las muestras, tanto el intento inicial de generar pares virtuales, como la opción final de descarga de conjuntos de muestra.

3 Estudio y análisis de programas y métodos de "calling".

En este capítulo se mostrará un análisis individual de cuatro herramientas utilizadas para efectuar *calling* en mutaciones somáticas. Estas herramientas son: *Mutect2, VarScan2, Strelka2 y SomaticSniper.*

[12] https://dcc.icgc.org/
[13] https://www.internationalgenome.org/home
[14] https://www.broadinstitute.org/

3.1 Mutect2

3.1.1 Introducción

Mutect2[15] se utiliza para el descubrimiento de variantes somáticas simples que se producen en nucleótidos individuales de tipo *SNVs* e *Indels* y forma parte de las herramientas contenidas en el kit de herramientas de análisis genómico GATK[16] del instituto *Broad*[17].

3.1.2 Instalación

GATK se puede desplegar en Linux y otras plataformas compatibles, también en MacOS X, pero no está disponible para sistemas Windows.

Para probar el funcionamiento de Mutect2 incluido en el GATK, instalamos GATK en Ubuntu[18] que es una distribución del sistema operativo Linux. Como no disponemos de un ordenador con Ubuntu como sistema operativo anfitrión, creamos una máquina virtual con Virtual Vox[19], software de virtualización. Una vez creada la máquina virtual con Ubuntu, en versión 16.04 LTS, la arrancamos y procedemos a la instalación de varias herramientas, incluido el GATK, basándonos en la documentación oficial del *Broad Institute*[20].

En el Anexo I se muestra paso a paso la instalación de estas herramientas. Aunque no todas las herramientas instaladas pueden ser necesarias para la aplicación de mutect2 se ha decidido seguir las indicaciones de *Broad Institute* en su tutorial sobre "*Install all software packages required to follow the GATK Best Practices"*, siendo Mutect2 una de las partes del *Genome Analysis Toolkit* (GATK).

[15] Mutect2: https://gatk.broadinstitute.org/hc/en-us/articles/360051306691-Mutect2
[16] GATK -Genome Analysis Toolkit-: https://gatk.broadinstitute.org/hc/en-us#info-tab
[17] Broad Institute: https://www.broadinstitute.org/about-us
[18] https://ubuntu.com/
[19] https://www.virtualbox.org/
[20]https://gatk.broadinstitute.org/hc/en-us/articles/360041320571--How-to-Install-all-software-packages-required-to-follow-the-GATK-Best-Practices

3.1.3 Modelo

Una vez instaladas las herramientas vamos a explicar el funcionamiento y modelo que utiliza Mutect2. Mutect2 está clasificado como una herramienta para efectuar el descubrimiento de mutaciones somáticas cortas en ensamblados de tipo haplotipo (haplotypes). "Un haplotipo es, en su sentido más general, un conjunto de variaciones de ADN a lo largo de un cromosoma que tienden a ser heredados juntos porque están muy próximos"[21]. Estas mutaciones cortas incluyen alteraciones de nucleótidos simples e inserciones y eliminaciones (indel). Mutect2 utiliza un modelo Bayesiano de genotipado somático y utiliza la maquinaria de ensamblado basada en la herramienta HaplotypeCaller[22]. HaplotypeCaller es una herramienta utilizada para el "calling" de variantes en el genoma en células germinales (óvulo y esperma) -germline variants-.(11,12)

Mutect2 toma como entrada dos muestras de DNA en formato BAM, una tumoral y una normal. Se llevan a cabo cuatro pasos, resumidos en la Figura 4[23]:

1. Eliminación de datos de baja calidad en la secuencia.
2. Detección de variantes en la muestra tumoral usando un clasificador Bayesiano.
3. Filtrado para eliminar falsos positivos.
4. Determinación de variantes somáticas mediante un segundo clasificador Bayesiano

La detección de variantes en la muestra tumoral, paso 2, se lleva a cabo utilizando dos modelos:

[21] https://www.genome.gov/es/genetics-glossary/Haplotipo
[22] https://gatk.broadinstitute.org/hc/en-us/articles/360050814612-HaplotypeCaller
[23] Aunque esta figura se corresponde con el proceso de detección de Mutect, versión anterior a Mutect2, sirve para describir el modo de trabajo de Mutect2, ya que la diferencia entre Mutect2 y Mutect es que Mutect2 aplica un modelo de genotipado somático bayesiano diferente al que aplicaba Mutect (https://gatk.broadinstitute.org/hc/en-us/articles/360037593851-Mutect2)

a. Modelo de referencia (M0), este modelo presupone que no hay variación en un sitio y que cualquier base observada que no se corresponda con la de referencia se debe a un error aleatorio en la secuenciación.

b. Modelo de variación (M_f^m), este modelo presupone que un sitio contiene un alelo modificado (m) en una determinada fracción (f) además de posibles errores de secuenciación. Se desconoce cuál es la fracción f y se estima como aquella fracción de las lecturas del tumor que soportan m. Se declara m como una variante candidata si la prueba de razón de verosimilitud de los datos bajo el modelo de variación y el de referencia sobrepasa un determinado umbral previamente definido, este umbral depende de la frecuencia de mutaciones esperada y de la ratio de falsos positivos deseada. La elección del umbral se puede utilizar para controlar la relación entre especificidad y sensibilidad. Se utilizará un umbral de 6.3, que se corresponde con una relación $10^{6.3}$:1 a favor del modelo de referencia, esto se considera dentro de lo razonable ya que la frecuencia de las mutaciones en muchos tumores es de 1-10 por Mb.

El filtrado, paso 3, elimina falsos positivos provocados por colocación inadecuada de las lecturas y por errores de secuenciación no independientes. Se aplican 6 filtros y un panel de normales (panel of normal) para eliminar eventos no deseados. El método se puede aplicar sin usar los filtros, usando los filtros y usando los filtros + el panel de normales.

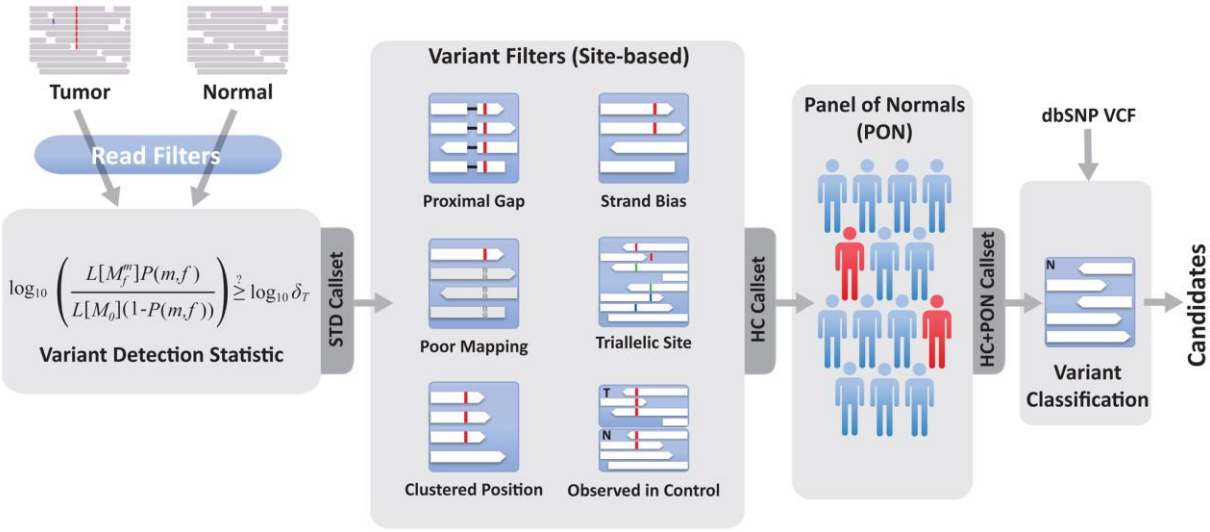


Figura 4. Proceso de detección de mutaciones somáticas usando Mutect2. -*Fuente* (11)-

Por último, en el paso 4, cada variante que se ha detectado en el paso 2 se clasifica como:

- somática, sino está presente en la muestra normal
- germinal, si está presente en la muestra normal
- variante, está presente en el tumor, pero no hay suficiente información para saber si está o no presente en la muestra normal

3.1.4 Test

Para probar el funcionamiento de Mutect2 usaremos el escenario con dos muestras tumor-normal, en este escenario tenemos una muestra tumoral que se corresponde con una muestra normal. Otros posibles escenarios serían analizar una muestra tumoral únicamente, o un análisis mitocondrial[24].

Ver el capítulo 2 donde se explica la selección de muestras.

Ver el anexo 5 donde se indican los pasos para la realización del test.

[24] Usage examples en https://gatk.broadinstitute.org/hc/en-us/articles/360051306691-Mutect2

3.2 VarScan2

3.2.1 Introducción

VarScan2 es una versión mejorada de VarSan (2009) desarrollada en el Instituto del Genoma de la Universidad de Washington. VarScan2 es un algoritmo para la detección de variantes en *next-generation sequencing data*. Mediante VarScan2 se pueden identificar variantes germinales, mutaciones somáticas, además de pérdidas de heterocigosidad (LOH *loss of heterozygosity*[25]) y eventos de tipo CNA (*copy number alterations*, también denominadas CNV o *copy number variations*[26]) (8).

3.2.2 Instalación

La herramienta está disponible para su descarga en http://varscan.sourceforge.net.

La herramienta está escrita en lenguaje Java y por tanto puede ser ejecutada en múltiples sistemas operativos (Linux, Mac, Windows).

Se pueden ver todos los pasos para la instalación en el Anexo VI.

3.2.3 Modelo

VarScan2, al contrario de lo que sucedía en Mutect2, no utiliza un modelo Bayesiano, sino que emplea una aproximación heurística/estadística para la detección de variantes. Teniendo en cuenta unos umbrales relacionados con la profundidad de las lecturas, la calidad, la frecuencia de variación de los alelos y la significación estadística.

Cuando se utiliza VarScan2 para la detección de variantes somáticas, el programa lee los ficheros "normal" y "tumor" de forma simultánea y

[25] LOH: se da cuando una célula somática de tipo heterocigótica se convierte en una de tipo homocigótica debido a que uno de los dos alelos se pierde. Este problema se ha reconocido como uno de los principales causantes de la tumorogénesis (https://www.sciencedirect.com/topics/biochemistry-genetics-and-molecular-biology/loss-of-heterozygosity)

[26] CNA/CNV: se trata de una variación en la estructura de las secuencias de DNA, de forma que para una determinada región de un cromosoma se aprecian variaciones en el número de copias de esa región; en algunas ocasiones estas copias incluyen genes completos, de forma que una persona puede tener 4 copias de un gen en vez de las 2 que son habituales, o puede tener 3 u otras variantes. (https://www.genome.gov/genetics-glossary/Copy-Number-Variation?id=40)

compara aquellas posiciones que están presentes en ambos y que tienen la suficiente cobertura (por defecto 8). Los ficheros tienen que haber sido previamente ordenados por posición.

Primero se realiza un *call* independiente para cada fichero utilizando la funcionalidad *germline consensus*[27], que consiste en la detección de variantes germinales (SNPs y Indels); luego se comparan los genotipos mediante el siguiente algoritmo:

If tumor matches normal:
If tumor and normal match the reference
==> Call Reference
Else tumor and normal do not match the reference
==> Call Germline

Else tumor does not match normal:
Calculate significance of allele frequency difference by Fisher's Exact Test
If difference is significant (p-value < threshold):
If normal matches reference
==> Call Somatic
Else If normal is heterozygous
==> Call LOH
Else normal and tumor are variant, but different
==> Call IndelFilter or Unknown
Else difference is not significant:
Combined tumor and normal read counts for each allele. Recalculate p-value.
==> Call Germline

Tabla 1. Algoritmo utilizado por VarScan2 para la detección de variantes somáticas (Fuente: http://dkoboldt.github.io/varscan/somatic-calling.html)

Una vez ejecutado el análisis se obtienen dos ficheros, uno contiene los SNVs (.snp) y el otro los indels (.indel). Se puede utilizar la opción -validation para obtener un tercer fichero que contendrá todas las posiciones que han sido analizadas (called).

[27] http://dkoboldt.github.io/varscan/germline-calling.html

El formato de estos ficheros, los campos que contienen, y cómo se corresponden estos campos con los de un fichero VCF, puede consultarse en la tabla que aparece en el apartado **Output** en este enlace: http://dkoboldt.github.io/varscan/somatic-calling.html.

En nuestro caso, y con el objeto de poder compararlo con otros métodos de calling, nos interesa la columna *somatic_status*, que se corresponde con la columna **INFO o columna 8** en un fichero VCF, y hace referencia al tipo de variante: germline, somatic, LOH o desconocida. Para nuestros propósitos nos interesan las somáticas.

3.2.4 Test

Para la prueba usaremos las mismas muestras tumor-normal que usamos con Mutect2. En el caso de Mutect2 usábamos ficheros .bam, pero VarScan requiere ficheros en formato *SAMtools pileup* obtenidos a partir de secuencias alineadas en formato .bam. Para poder crear estos ficheros en el formato pileup necesitamos:

- Que los ficheros .bam estén ordenados por posición mediante el comando sort de SAMtools
- El fichero del genoma de referencia con el que se alinearon los datos en formato .fasta.

Hay que comprobar la versión que tenemos de SAMtools, pues en función de esta el comando para obtener los ficheros pileup es diferente. En nuestro caso la versión es la 1.11:

```
sonia@sonia-VirtualBox:~/tfm$ sam/samtools --version
samtools 1.11
Using htslib 1.11
Copyright (C) 2020 Genome Research Ltd.
```

Usaremos por tanto esta sintaxis:

```
samtools mpileup -f [reference sequence] [BAM file] >myData.pileup
```

Una vez tenemos los ficheros preparados, usamos esta sintaxis para hacer el análisis:

```
java -jar VarScan.jar somatic normal.pileup tumor.pileup output.basename
```

En el Anexo VII se pueden ver los pasos llevados a cabo para la realización del análisis con VarScan2.

3.3 SomaticSniper

3.3.1 Introducción

SomaticSniper es un software que utiliza un modelo Bayesiano para realizar una comparativa entre probabilidades de ocurrencia de genotipos en muestras normales y sus correspondientes muestras tumorales. Se basa en el algoritmo incluido en el paquete MAQ, tal y como está implementado en Samtools (13). El algoritmo MAQ es un algoritmo que permite realizar el alineamiento de lecturas o secuencias cortas con el genoma de referencia, además permite descubrir variantes de tipo SNPs e indels entre el genoma de referencia y la muestra. MAQ tiene dos fases en su ejecución. En la primera fase se produce el alineamiento. Si durante el alineamiento una secuencia puede ser igualmente alineada con varias posiciones del genoma de referencia, MAQ selecciona una de ellas al azar y la alinea con esa, asignando una calidad de alineamiento de cero a esa secuencia. Estas secuencias con una calidad de alineamiento de cero no van a contribuir al descubrimiento de variantes, pues se pueden alinear igualmente bien con varias posiciones, no obstante, dan información del número de secuencias repetitivas y pueden ser filtradas si se desea. En la segunda fase se produce el descubrimiento de SNPs e indels, cuyo funcionamiento se explica en el punto 3.3.3 Modelo (7).

SomaticSniper compara dos ficheros .bam (tumor-normal) y su salida es un fichero con un formato muy similar al estándar definido en Samtools.

3.3.2 Modelo

Para el descubrimiento de variantes somáticas SomaticSniper se basa en el modelo probabilístico de genotipado de MAQ, calculando la probabilidad de que los genotipos de un par tumor-normal sean diferentes. Esta probabilidad se muestra como una "puntuación somática". Esta

puntuación suele estar entre 0 y 255 y se trata de una probabilidad *Phred-scaled probability*[28]. Si la puntuación es alta la probabilidad de que haya una variante somática en una determinada posición es alta y al contrario.

El algoritmo para detectar diferencias entre el tumor y su par normal utiliza una puntuación que se calcula de la siguiente manera (7):

Modo 1: suponiendo independencia entre los datos de las muestras tumoral y normal. Se calcula la probabilidad de que una determinada variante no sea somática. Tomando el dato de la muestra tumoral (T), la normal (N) y la de referencia (G); se calcula una puntuación S usando la siguiente fórmula:

$$S = -10\log_{10}\left(\sum_{G_i=0}^{9}\frac{P(T|G_i)P(G_i)P(N|G_i)P(G_i)}{\sum_{G_j=0}^{9}P(T|G_j)P(G_j)\sum_{G_k=0}^{9}P(N|G_k)P(G_k)}\right)$$

Como se aprecia se trata de probabilidades condicionadas del tipo P(D|Gi), siendo D el dato de la muestra tumoral o de la normal en cada caso, y con relación a los 10 posibles tipos de combinaciones de los genotipos diploides: AA, AC, AG, AT, CC, CG, CT, GG, TT; por eso la i, j o k varían entre 0 y 9.

La probabilidad $P(G_i)$ se calcula así:

$$P(G_1) = \begin{cases} \theta & \text{Case 1} \\ \frac{\theta}{2} & \text{Case 2} \\ \theta^2 & \text{Case 3} \\ 1-\sum_{k=0}^{9}P(G_k)P(G_k \neq G_R) & \text{Case 4} \end{cases}$$

θ *tasa de mutación heterocigóta en la población de interés*

[28] https://gatk.broadinstitute.org/hc/en-us/articles/360035531872-Phred-scaled-quality-scores

G_R *referencia base en la posición de interés*

Caso 1: el genotipo es heterocigoto, pero comparte un alelo con la referencia. Ejemplo: la referencia es A y Gi=AG.

Caso 2: el genotipo es una variante homocigota.

Caso 3: el genotipo es heterocigoto y no comparte alelos con la base de referencia. Ejemplo: la referencia es A y Gi=CG.

Caso 4: el genotipo es homocigoto en la base de referencia.

Modo 2: debido a que la muestra tumoral y normal proceden del mismo individuo, no se pueden considerar independientes y se plantea esta otra opción:

$$S = -10\log_{10}\left(\frac{\sum_{i=0}^{9} P(T|H_i)P(N|G_i)P(H_i|G_i)P(G_i)}{\sum_{j=0}^{9}\sum_{k=0}^{9} P(N|G_k)P(T|H_j)P(H_j|G_k)P(G_k)}\right)$$

En este caso, la G representa el genotipo en la muestra normal y H es el genotipo en la muestra tumoral. La probabilidad P(Hm|G) se define como:

$$P(H_m|G) = \begin{cases} \mu & \text{when } H_m \text{ shares an allele with } G \\ \mu^2 & \text{when } H_m \text{ shares no alleles with } G \\ 1 - \sum_{n=0}^{n=9} P(H_n|G)P(H_m \neq G) & \text{when } H_m \text{ equals } G \end{cases}$$

μ: probabilidad previa de una mutación somática para un genotipo normal G

Un valor de μ de 0.01, valor mucho mayor que las tasas de mutaciones somáticas observadas en tumores, proporciona resultados parecidos a la primera ecuación planteada. Un valor de 0.000001 se presenta como un valor preciso, pero puede causar pérdida de sensibilidad cuando las lecturas son de baja cobertura;

Se puede aumentar este valor para conseguir mayor sensibilidad, pero también se corre el riesgo de que aumente el número de falsos positivos[29].

Además, se aplican varios filtros para eliminar los posibles errores no encontrados por el modelo MAQ. Las variantes somáticas que pasan estos filtros se interseccionan con la distribución 130 de dbSNP[30], de forma que los sitios que empaten en ambas posiciones y alelo del dbSNP se eliminan. También se eliminan aquellos sitios donde el genotipo en la muestra normal es heterocigoto y en la muestra tumoral es homocigoto. Por otro lado, se clasifican las mutaciones en las altamente confiables y las que no. Las altamente confiables son aquellas en las que la calidad de cobertura de mapeo es >=40, BWA – Algoritmo Burrows-Wheller-, y la puntuación somática (S) es >= 40.

3.3.3 Instalación

SomaticSniper se puede descargar de esta página http://gmt.genome.wustl.edu/packages/somatic-sniper/.

Se puede conseguir el código fuente del repositorio **github** al que da acceso la página anterior o se puede descargar como un paquete para diferentes distribuciones de Linux.

En el Anexo VIII se pueden ver los pasos de la instalación.

3.3.4 Test

Nuevamente, usaremos las mismas muestras tumor-normal que usamos con Mutect2 y VarScan2. SomaticSniper tiene como entrada dos ficheros .bam, el de la muestra tumoral y el de la muestra normal.

[29] http://gmt.genome.wustl.edu/packages/somatic-sniper/documentation.html
[30] dbSNP es la mayor base de datos de variaciones de nucleótidos, forma parte del Centro Nacional de Información Biotecnológica (NCBI). Es una base de datos relacional implementada en un servidor SQL. https://www.ncbi.nlm.nih.gov/books/NBK44469/.
https://www.ncbi.nlm.nih.gov/projects/SNP/snp_summary.cgi?build_id=130

SomaticSniper puede ser utilizado en dos modos mencionados en el apartado 3.3.3, el modo por defecto es el Modo 1, si queremos usar el Modo 2 debemos utilizar la opción -J.

La sintaxis general de la orden es:

```
bam-somaticsniper [options] -f <ref.fasta> <tumor.bam> <normal.bam> <snv_output_file>
```

Se puede consultar el listado completo de opciones en http://gmt.genome.wustl.edu/packages/somatic-sniper/documentation.html.

En el Anexo IX se pueden ver los pasos llevados a cabo para la realización del análisis con SomaticSniper.

3.4 Strelka2

3.4.1 Introducción

Strelka2 (9) es un método de descubrimiento de variantes tanto germinales como somáticas, que se basa en la anterior versión Strelka (14).

3.4.2 Modelo

Los pasos que se llevan a cabo para el descubrimiento de las variantes somáticas se muestran en la siguiente figura (Figura 5).

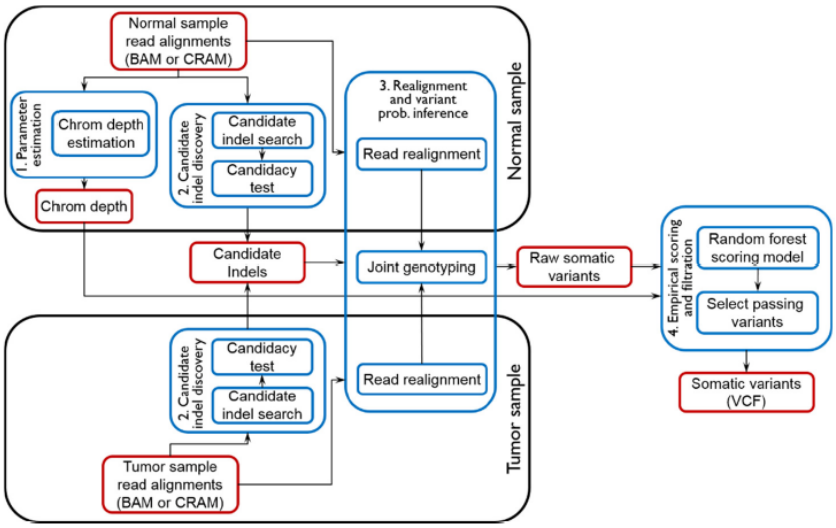


Figura 5. Flujo de trabajo aplicado por Strelka2 para la detección de variantes somáticas (Fuente: Supplementary Figures (9))

Como se aprecia, primero hay un trabajo previo con la muestra normal y tumoral por separado para la detección de indels candidatas, las lecturas realineadas se combinan, y teniendo en cuenta las indels candidatas, se obtiene un conjunto en crudo de variantes somáticas que, tras pasar por la fase de asignación de una puntuación y el filtrado, da finalmente las variantes somáticas detectadas en formato vcf.

El modelo de descubrimiento de variantes somáticas asume que las muestras son de tipo diploide. Tanto para los SNVs como para los indels, los posibles estados de los genotipos en la muestra normal son:

$$G_n \in \{g_{\text{ref}}, g_{\text{het}}, g_{\text{hom}}\}$$ [31]

Refiriéndose bien a no-variante o variante heterocigota o homocigota en la muestra normal.

Los posibles estados de los genotipos tumorales son:

$$G_t \in \{g_{\text{nonsom}}, g_{\text{som}}\}$$

Refiriéndose a la ausencia o presencia de una variante somática en la muestra tumoral, respectivamente.

El método calcula una probabilidad cuando combina los genotipos de la muestra normal y tumoral:

$$P(G_t, G_n \mid D) \propto P(G_t, G_n) P(D \mid G_t, G_n)$$

D: datos de la secuencia de ambas muestras

[31] Esta ecuación y las siguientes relativas al funcionamiento del modelo Strelka2 están extraídas del artículo de referencia que detalla el funcionamiento de este modelo (9)

$$P(D \mid G_t, G_n) = \int_{F_t, F_n} P(D \mid F_t, F_n) P(F_t, F_n \mid G_t, G_n)$$

F_t y F_n son las frecuencias de los alelos en la muestra tumoral y normal, respectivamente.

La probabilidad de la frecuencia del alelo: $P(D|F_t,F_n)$ se descompone para cada muestra en $P(D_t|F_t)P(D_n|F_n)$, donde D_t y D_n se refieren al dato de la muestra tumoral o de la normal, respectivamente. Se puede ver el desarrollo completo de todo el modelo en el artículo de referencia (9).

Una vez obtenidas las variantes somáticas en crudo se someten al cálculo de una puntuación y un filtrado. Esto permite aumentar la precisión del descubrimiento de variantes, descartando aquellas que no alcanzan la puntuación o no pasan los filtros. Se utiliza el denominado modelo *Empirial Variant Scoring (EVS), un clasificador supervisado de tipo bosque aleatorio* (Figura 6). Este clasificador es entrenado con datos etiquetados de secuencias obtenidas bajo varias condiciones (diferentes secuenciadores, diferente preparación de la muestra, distintas coberturas). El modelo EVS proporciona una puntuación de calidad que se agrega a cada variante. Dependiendo de si Strelka2 se usa para la detección de variantes germinales o de variantes somáticas, los conjuntos y modelos de EVS son diferentes. El modelo EVS es pre-entrenado lo que permite mejores costes en tiempo de ejecución con relación a otros sistemas como el usado en GATK VQSR.

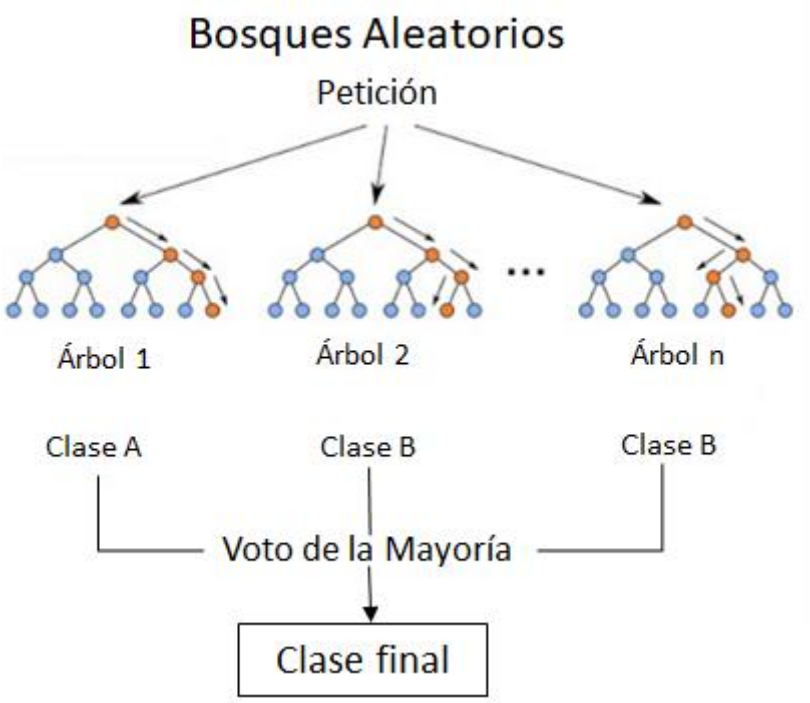


Figura 6. Bosque aleatorio
(Fuente: https://medium.com/@hpumah/bosques-aleatorios-482163ace92e)

3.4.3 Instalación

Para la descarga e instalación de Strelka2 debemos dirigirnos a la siguiente página:

https://github.com/Illumina/strelka/blob/v2.9.x/docs/userGuide/quickStart.md

En ella se nos indican los comandos que debemos ejecutar en la consola de la distribución Linux para que se realice la instalación y se efectúe una prueba que verifique el éxito de esta.

En el Anexo X pueden verse los pasos llevados a cabo.

3.4.4 Test

Para llevar a cabo el test con Strelka2 necesitamos como en ocasiones anteriores los ficheros .bam tumoral y normal, el fichero de referencia hg38, así como especificar el directorio donde irán a parar los resultados del análisis.

La ejecución del análisis tiene dos pasos. Uno de configuración, usando esta sintaxis:

```
# configuration
${STRELKA_INSTALL_PATH}/bin/configureStrelkaSomaticWorkflow.py \
    --normalBam normal.bam \
    --tumorBam tumor.bam \
    --referenceFasta hg38.fa \
    --runDir demo_somatic
```

Y otro de ejecución del flujo de trabajo, para lo cual es esta sintaxis:

```
# execution on a single local machine with 20 parallel jobs
demo_somatic/runWorkflow.py -m local -j 20
```

En el Anexo XI se puede ver paso a paso el análisis realizado con Strelka2 ajustando las sintaxis anteriores a las secuencias de ejemplo.

4 Comparativa de resultados

Para los datos de ejemplo analizados Mutect2, VarScan2, SomaticSniper y Strelka2 arrojaron resultados diferentes en cuanto al número de variantes somáticas detectadas cuando se usaron los parámetros de configuración por defecto y/o recomendados. Mientras que Mutect2 arrojó 28 variantes detectadas, VarScan nos dio solo 2, SomaticSniper con los parámetros recomendados por los autores dio 136 en el Modo 1 y 59 en el Modo 2 y Strelka2 detectó 68 que pasaran todos los filtros (1 de tipo indel y 67 de tipo SNV). Además, Mutect2 detectó las variantes solo en posiciones del chr17, VarScan lo hizo en posiciones del chr6, SomaticSniper en posiciones del chr6, del chr11 y del chr17 y Strelka2 en posiciones del chr17.

Vimos en el Anexo V, que el resultado del análisis de Mutect2 se almacenó en el fichero somatic_oncefiltered.vcf.gz y que las variantes somáticas detectadas, que se consideraban pasaban los filtros, tenían el texto PASS en el campo FILTER. Si contamos cuántas de las líneas de este fichero contienen el texto chr17 vemos como son las mismas que contienen el texto PASS.

Primero vamos a quedarnos solo con las líneas del fichero que contienen el texto PASS, para quitar las cabeceras.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ zcat
somatic_oncefiltered.vcf.gz | grep -v "#" | grep "PASS" >
somatic_m2.vcf
```

Contamos:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat somatic_m2.vcf |
grep "chr17" | wc -l
28
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat somatic_m2.vcf |
grep "PASS" | wc -l
28
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat somatic_m2.vcf |
wc -l
28
```

En el Anexo VII vimos que los ficheros obtenidos en el análisis con VarScan2 se denominaban somatic_varscan.snp y somatic_varscan.indel, y almacenaban las variantes de tipo SNP e indel detectadas. Nos interesan las clasificadas como Somatic. Esto podemos verlo en el campo 13 (somatic_status). Si nos fijamos en qué cromosoma se detectan las somáticas (las cuales solo estaban en el fichero .snp), vemos que es el chr6:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat
somatic_varscan.snp | grep "Somatic"
chr6  29944193    G     A     41    8     16,33%      G     34
21    38,18%      R     Somatic     1.0   0.011137449944548962
30    4     17    4     20    21    7     1
chr6  29944538    A     C     17    3     15%   A     0     9
100%  C     Somatic     1.0   2.1967038458792515E-5   0     0
0     9     12    5     0     3
```

En general todas las variantes, que son 48 (una de tipo indel y 47 de tipo snp), también se corresponden con posiciones del chr6:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat
somatic_varscan.snp | grep  "chr6" | wc -l
47
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat
somatic_varscan.indel | grep  "chr6" | wc -l
1
```

En cuanto a SomaticSniper si queremos saber en qué cromosomas se han detectado las variantes podemos mostrar el contenido del fichero obtenido en la ejecución en el Modo 2 (ver Anexo IX fichero

snv_somaticsniperM2.vcf). Usamos el comando **cat** y **more** para verlo paso a paso, ya que no son demasiadas líneas. El vistazo rápido nos indica que los cromosomas son el chr6, el 17 y el 11. Para comprobarlo contamos:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep "chr6" | wc -l
6
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep "chr17" | wc -l
50
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep "chr11" | wc -l
3
```

Sumando 6+50+3=59 total líneas de variantes detectadas.

Veamos si las 2 variantes detectadas con VarScan2 en el chr6 están dentro de las 6 detectadas con SomaticSniper:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep "chr6"
chr6    29945148        .       A       G       .       .       .       GT:IGT:DP:DP4:BCOUNT:GQ:JGQ:VAQ:BQ:MQ:AMQ:SS
:SSC    0/0:0/0:17:8,9,0,0:17,0,0,0:78:51:0:29:52:52:0:.       0/1:0/1:5:2,0,1,2:2,0,3,0:8:51:44:27,30:54:45,60:2:5
1
chr6_GL000252v2_alt     1200681 .       G       C       .       .       .       GT:IGT:DP:DP4:BCOUNT:GQ:JGQ:VAQ:BQ:M
Q:AMQ:SS:SSC    0/0:0/0:41:15,25,1,0:0:1,40,0:120:121:0:32:58:58:0:.    0/1:0/1:10:1,2,3,4:0:7,3,0:39:121:87:30,33:3
9:30,60:2:103
chr6_GL000253v2_alt     1199671 .       G       C       .       .       .       GT:IGT:DP:DP4:BCOUNT:GQ:JGQ:VAQ:BQ:M
Q:AMQ:SS:SSC    0/0:0/0:36:12,23,1,0:0:1,35,0:105:123:0:32:57:58:0:.    0/1:0/1:9:0,2,3,4:0:7,2,0:7:123:55:30,33:37:
30,60:2:88
chr6_GL000254v2_alt     1289208 .       G       C       .       .       .       GT:IGT:DP:DP4:BCOUNT:GQ:JGQ:VAQ:BQ:M
Q:AMQ:SS:SSC    0/0:0/0:41:15,25,1,0:0:1,40,0:120:121:0:32:58:58:0:.    0/1:0/1:10:1,2,3,4:0:7,3,0:39:121:87:30,33:3
9:30,60:2:103
chr6_GL000255v2_alt     1200087 .       G       C       .       .       .       GT:IGT:DP:DP4:BCOUNT:GQ:JGQ:VAQ:BQ:M
Q:AMQ:SS:SSC    0/0:0/0:36:12,23,1,0:0:1,35,0:105:123:0:32:57:58:0:.    0/1:0/1:9:0,2,3,4:0:7,2,0:7:123:55:30,33:37:
30,60:2:88
chr6_GL000256v2_alt     1242943 .       G       C       .       .       .       GT:IGT:DP:DP4:BCOUNT:GQ:JGQ:VAQ:BQ:M
Q:AMQ:SS:SSC    0/0:0/0:36:12,23,1,0:0:1,35,0:105:123:0:32:57:58:0:.    0/1:0/1:9:0,2,3,4:0:7,2,0:7:123:55:30,33:37:
30,60:2:88
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat somatic_varscan.snp | grep "Somatic"
chr6    29944193        G       A       41      8       16,33%  G       34      21      38,18%  R       Somatic 1.00
.011137449944548962     30      4       17      4       20      21      7       1
chr6    29944538        A       C       17      3       15%     A       0       9       100%    C       Somatic 1.02
.1967038458792515E-5    0       0       0       9       12      5       0       3
```

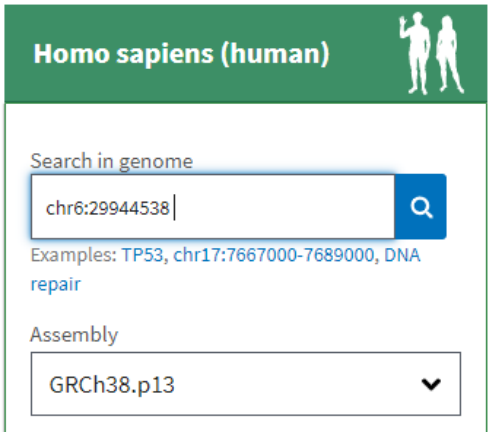
Como se aprecia, las posiciones en las que se detectan las variantes con VarScan2 (29944193 y 29944538) no coinciden con las detectadas por SomaticSniper en el Modo 2.

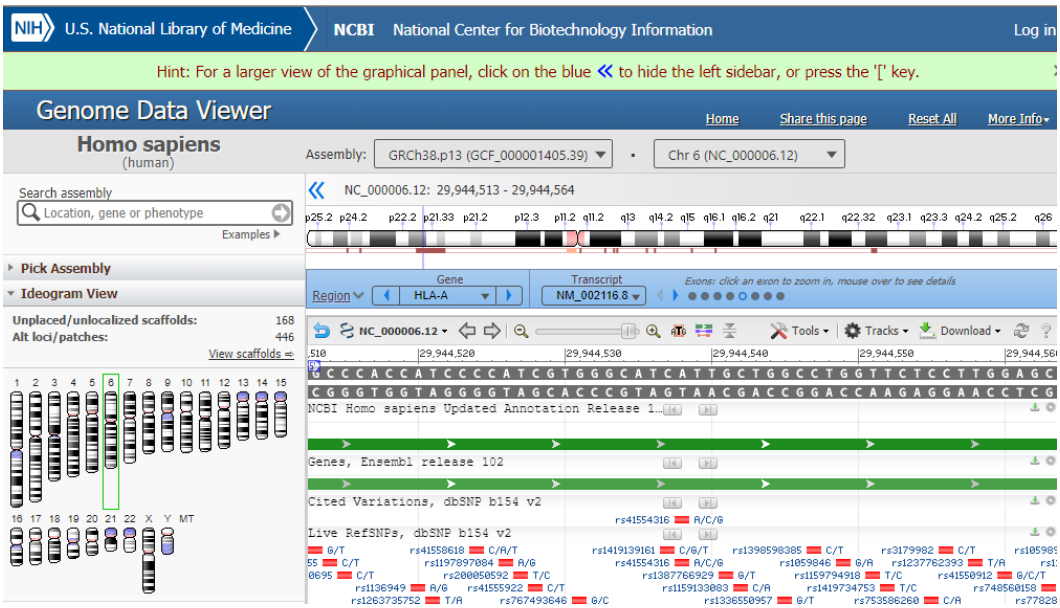
Probemos si en el Modo 1 están (fichero snv_somaticsniper.vcf obtenido en el Anexo IX):

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniper.vcf | grep "chr6"
chr6    29944538        .       A       C       .       .       .       GT:IGT:DP:DP4:BCOUN
:SSC    0/0:0/0:33:20,7,0,6:27,4,1,0:53:.:0:31:57:59:0:.       1/1:1/1:10:0,1,0,9:1,9,0,0:
chr6    29945148        .       A       G       .       .       .       GT:IGT:DP:DP4:BCOUN
:SSC    0/0:0/0:17:8,9,0,0:17,0,0,0:78:.:0:29:52:52:0:. 0/1:0/1:5:2,0,1,2:2,0,3,0:8:.:44:27
chr6_GL000250v2_alt     1201941 .       G       A       .       .       .       GT:IGT:DP:D
Q:AMQ:SS:SSC    0/0:0/0:58:16,31,3,8:11,0,47,0:55:.:0:33:22:24:0:.      1/1:1/1:74:22,18,9,
10:19:2:60
chr6_GL000251v2_alt     1423616 .       G       A       .       .       .       GT:IGT:DP:D
```

Vemos que la detectada en la posición 29944538 en VarScan2 también ha sido detectada por SomaticSniper en el Modo1.

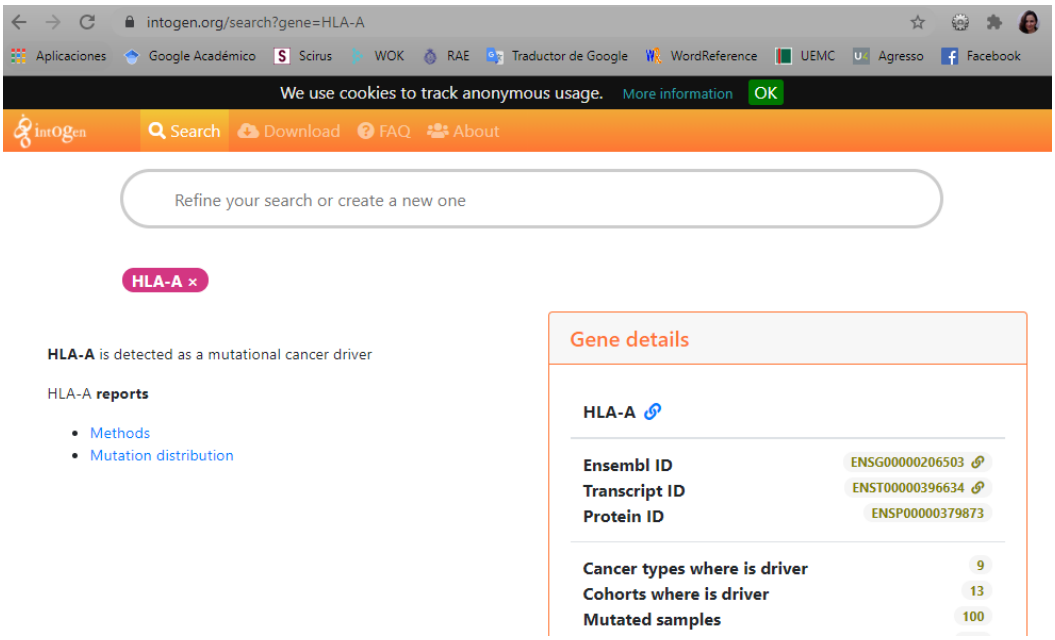
Si buscamos el gen en el que se encuentra la posición chr6: 29944538 en *Genome Data Viewer*[32]*:*





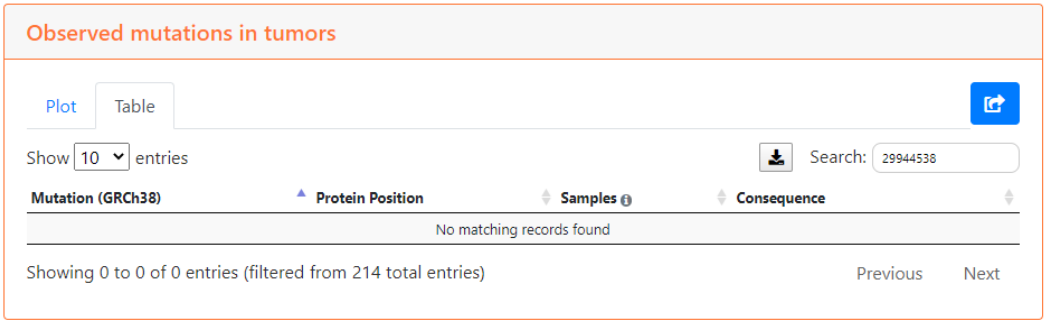
Vemos que se trata del gen HLA-A. Luego podemos buscar en IntOgen[33] para ver si este gen está asociado al cáncer, si es un *cancer driver*:

[32] https://www.ncbi.nlm.nih.gov/genome/gdv/
[33] https://www.intogen.org/search



Vemos que efectivamente este gen se clasifica dentro de los *mutational cancer driver*.

Si buscamos exactamente la posición 29944538 en la tabla de mutaciones observadas en tumores (en esta misma página, un poco más abajo), vemos que no se encuentra:



Por lo que según IntOgen se trataría de un falso positivo en lo relacionado con mutaciones asociadas a cáncer.

Veamos si las 28 detectadas por Mutect2 en el chr17 están entre las 50 detectadas con SomaticSniper en el Modo2. Primero obtenemos las posiciones ordenadas de los ficheros somatic_m2.vcf y snv_somaticsniperM2.cvf que contengan respectivamente las variantes detectadas en posiciones del chr17. Luego comparamos los ficheros

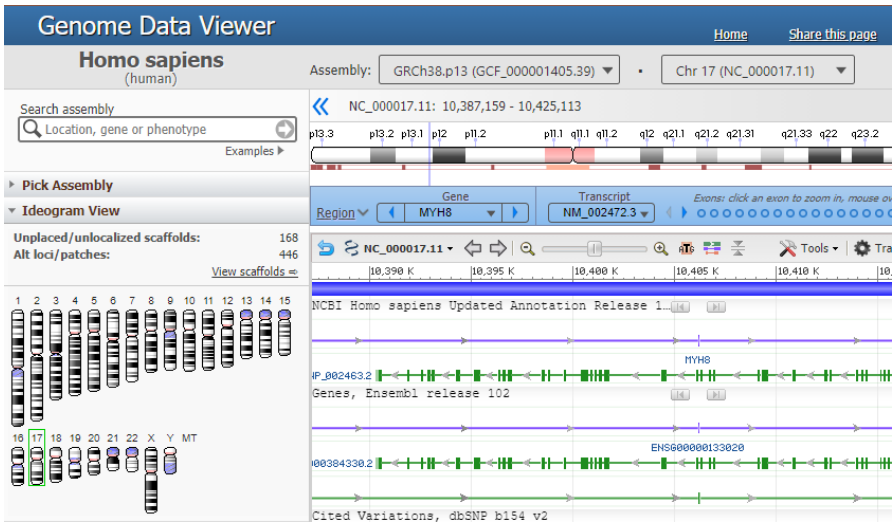
resultantes con el comando **comm**, este comando nos muestra tres columnas: la primera contiene todas las filas que solo aparecen en el primer archivo, la segunda todas las filas que solo aparecen en el segundo archivo y la tercera las filas que aparecen en ambos archivos. Si usamos las opciones -1 y -2 el comando solo nos mostrará las filas que aparecen en ambos, pues elimina las que solo están en el primero (-1) y las que solo están en el segundo (-2).

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat somatic_m2.vcf | cut -f 2 | sort > somatic_m2_posS.vcf
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep "chr17" | cut -f 2 | sort
 > snv_somaticsniperM2_posS.vcf
```

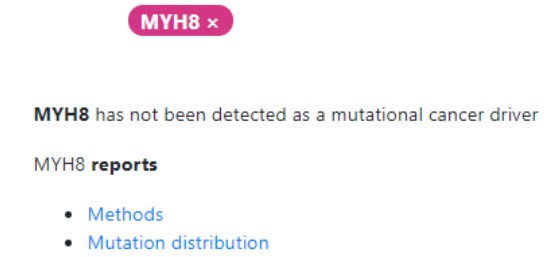
```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 somatic_m2_posS.vcf snv_somaticsniperM2_posS.vcf
10410710
19556155
2394409
29609163
39988669
41349384
4632718
50461355
5541887
6707176
6779878
75225975
75239434
76626447
7674220
81671752
82374762
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 somatic_m2_posS.vcf snv_somaticsniperM2_posS.vcf
| wc -l
17
```

Vemos que 17 de las variantes detectadas por Mutect2 y SomaticSniper en el chr17 coinciden.

Podemos buscar alguna de estas posiciones en *Genome Data Viewer*, por ejemplo, la primera 10410710:

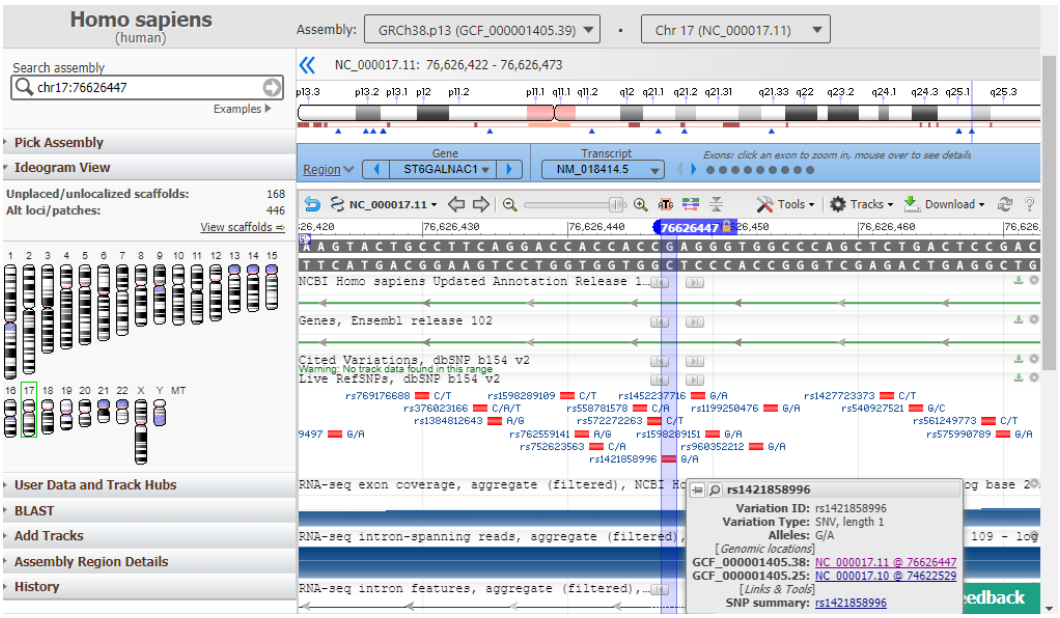


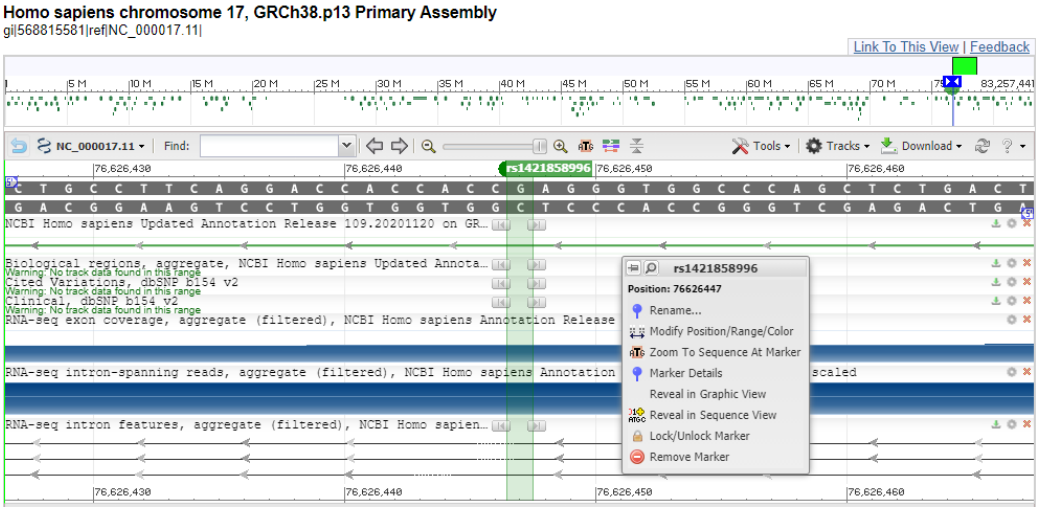
Vemos que el gen correspondiente es el MYH8. Lo buscamos en IntOgen:



En este caso, este gen no está clasificado como un gen que propicie el avance y desarrollo del cáncer.

En el propio *Genome Data Viewer* podemos ver información sobre una variante. Por ejemplo, si tomamos la chr17: 76626447 y nos colocamos encima, nos muestra información del tipo de variante (*Variation Type: SNV, length 1)*:





Sigamos con la comparación de variantes encontradas. Hagamos lo mismo, pero para el Modo 1: ¿Cuántas de las 28 detectadas por Mutect2 en el chr17 están entre las 116 detectadas con SomaticSniper en el Modo1?

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniper.vcf | grep "chr17" | wc -l
116
```

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniper.vcf | grep "chr17" | cut -f 2 |
 sort > snv_somaticsniper_posS.vcf
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 somatic_m2_posS.vcf snv_somaticsniper_posS
.vcf | wc -l
17
```

Vemos que también son 17. Muchas de las variantes detectadas por SomaticSniper en el chr17 en el Modo 2 están también incluidas entre las detectadas en el Modo 1.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 snv_somaticsniper_posS.vcf snv_somaticsnip
erM2_posS.vcf | wc -l
46
```

De las 50 variantes detectadas por SomaticSniper en el chr17 en el Modo 2 son detectadas en el Modo 1 46 de esas 50.

En cuanto a Strelka2, todas las variantes habían sido detectadas en el chr17. Veamos las que coinciden con Mutect2 y con SomaticSniper.

Primero trasladamos los ficheros con las variantes que pasan los filtros a la carpeta tutorial_11136, que es donde tenemos el resto de ficheros de los análisis realizados.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ zcat somatic.indels.vcf.gz
 | grep "PASS" > ../../../strelka2_somatic_indel.vcf
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ zcat somatic.snvs.vcf.gz |
 grep "PASS" > ../../../strelka2_somatic_snvs.vcf
```

Ahora nos quedamos con las posiciones, ordenamos y comparamos.

En el caso de las de tipo indel, como solo había 1 no es necesario ordenar.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat strelka2_somatic_indel.vcf | cut -f 2 > strelka2_somatic
_indel_posS.vcf
```

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat strelka2_somatic_snvs.vcf | cut -f 2 | sort > strelka2_s
omatic_snvs_posS.vcf
```

Comparamos:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 snv_somaticsniperM2_posS.vcf strelka2_somatic_indel_posS.vcf | wc -l
0
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 somatic_m2_posS.vcf strelka2_somatic_indel_posS.vcf | wc -l
0
```

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 snv_somaticsniperM2_posS.vcf strelka2_somatic_snvs_posS.vcf | wc -l
31
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 somatic_m2_posS.vcf strelka2_somatic_snvs_posS.vcf | wc -l
24
```

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 snv_somaticsniper_posS.vcf strelka2_somatic_snvs_posS.vcf | wc -l
38
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ comm -1 -2 snv_somaticsniper_posS.vcf strelka2_somatic_indel_posS.vcf | wc -l
0
```

Tabla 2. Resultados de la comparativa entre los programas de calling (Mutect2, VarScan2, SomaticSniper y Strelka2)

Programa	Nº de variantes	Cromosoma	Coincidencia Mutect2	Coincidencia VarScan2	Coincidencia SomaticSniper	Coincidencia Strelka2
Mutect2	28	chr17		0	17 Modos 1 y 2	24
VarScan2	2	chr6	0		1 Modo 1 0 Modo 2	0
SomaticSniper	136 Modo 1 59 Modo 2	chr6, chr11, chr17	17 Modos 1 y 2	1 Modo 1 0 Modo 2		38 Modo 1 31 Modo 2
Strelka2	68	Chr17	24	0	38 Modo 1 31 Modo 2	

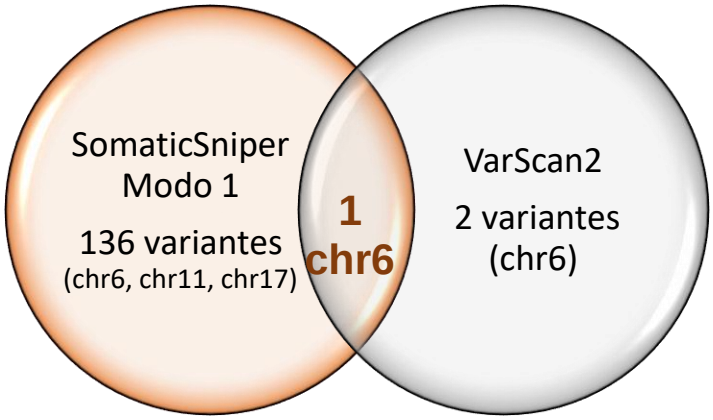


Figura 7. Una variante coincidente en el mismo cromosoma -chr6- y posición entre SomaticSniper -Modo 1- y VarScan2

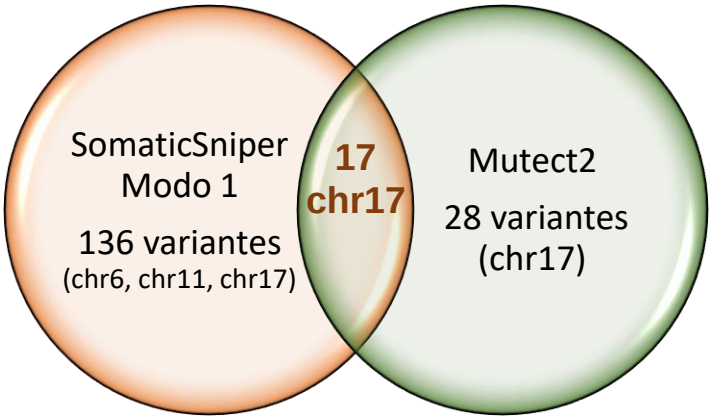


Figura 8. Diecisiete variantes en el mismo cromosoma -chr17- y mismas posiciones entre SomaticSiniper -Modo 1- y Mutect2

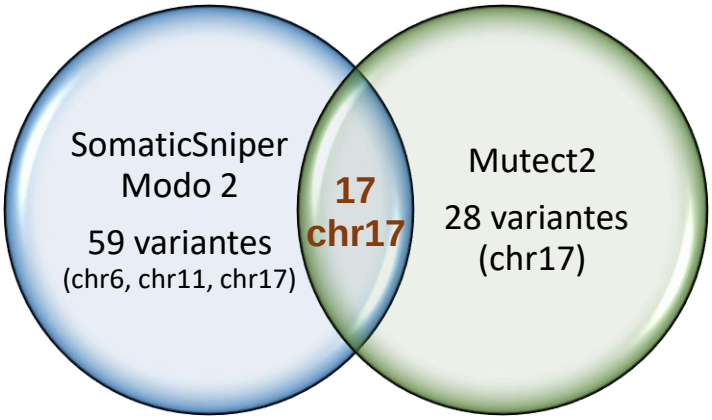


Figura 9. Diecisiete variantes en el mismo cromosoma -chr17- y mismas posiciones entre SomaticSiniper -Modo 2- y Mutect2.

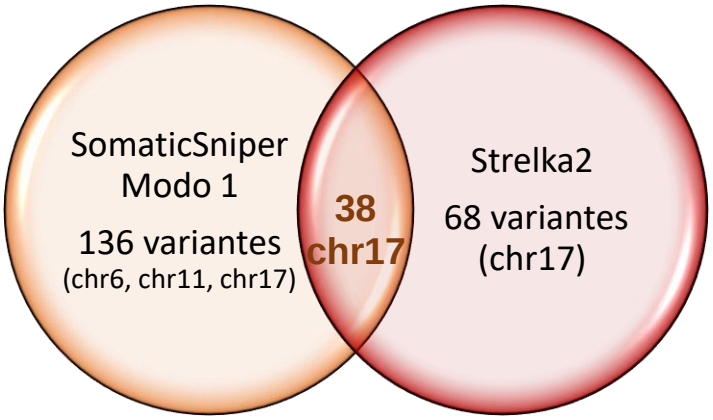


Figura 10. Treinta y ocho variantes en el mismo cromosoma -chr17- y mismas posiciones entre SomaticSiniper -Modo 1- y Strelka2

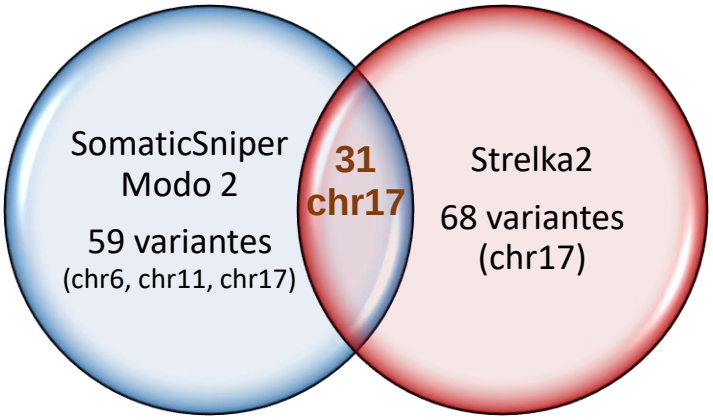


Figura 11. Treinta y una variantes en el mismo cromosoma -chr17- y mismas posiciones entre SomaticSiniper -Modo 2- y Strelka2.

Ya hemos visto que un ajuste en los parámetros de configuración en SomaticSniper (ver Anexo IX) afectaba al resultado obtenido; previsiblemente lo mismo sucederá con el resto de los métodos de calling. Se podrían realizar pruebas con diferentes configuraciones de parámetros en cada método buscando un mayor acercamiento entre la detección de variantes somáticas entre los métodos.

Como se ha podido comprobar el resultado de los diferentes programas de descubrimiento de variantes somáticas solo coincide en un subconjunto de variantes. Esto ya se ha visto en otros estudios publicados, con algunos de estos mismos programas analizados aquí y otros diferentes (GRISP, SNVer, EBCall, JointSNVMix, Streika). También se ha reportado que los mejores son VarScan2 y Mutect2; aunque no

parece que esté cercano un consenso para la aplicación de un estándar (3).

5 Conclusiones

5.1 Lecciones aprendidas

Uno de los principales problemas encontrados durante el desarrollo del trabajo se ha debido a la obtención de las muestras de ejemplo. La lectura inicial de artículos científicos sobre los métodos de calling nos dirigió hacia fuentes de datos (Genomic Data Commons y The International Cancer Genome Consortium) que finalmente se comprobaron como inviables al necesitar permisos especiales para la descarga de las muestras que requería nuestro estudio. La alternativa a esta piedra en el camino fue la generación de nuestras propias muestras virtuales partiendo de datos del 1000 Genome Project, que no concluyó con éxito al encontrar dificultades en la alineación con el genoma de referencia. Además, existía la limitación de la máquina, ordenador, donde se iban a efectuar las pruebas; la cual no soportaba el trabajo con muestras de gran tamaño, por lo que habían de conseguirse ficheros de nuestras tumor-normal del orden de Mb. Finalmente se consiguió trabajar con unas muestras de ejemplo obtenidas del repositorio del Instituto Broad. Esto ha limitado el estudio, ya que el no poder disponer de múltiples muestras para el análisis, y que estas sean completas, empobrece la comparativa final.

5.2 Logro de objetivos

Los objetivos finalmente definidos, que consistían básicamente en un estudio y comparativa de diferentes métodos de detección de variantes somáticas, se han conseguido completar con éxito con ciertas limitaciones y además hubo que adaptar la versión inicial de los mismos.

Cuando se planteó la primera redacción de los objetivos, en la elaboración inicial del primer plan de trabajo, apenas se conocía el mundo del análisis de variantes somáticas, los conceptos y nomenclatura asociados al mismo y no se había puesto en práctica el trabajo con los programas de detección

de variantes, solo se habían leído unos pocos artículos y se plantearon los objetivos en base a los estudios que en estos artículos se reflejaban. Una vez avanzado en el conocimiento, y tras las pruebas efectuadas con las dos primeras herramientas, Mutect2 y VarScan2, se comprobó que no se disponía de los recursos necesarios para abordar un trabajo similar al que se plasmaba en los estudios de los artículos. Hubo que adaptar la redacción de los objetivos, de forma que se eliminó lo relacionado con la obtención de gráficas y la comparativa de la sensibilidad y especificidad, debido a las limitaciones derivadas de la no posibilidad de obtener múltiples muestras de ejemplo tumor-normal completas y las limitaciones de cálculo del ordenador de trabajo. Por tanto, las aspiraciones iniciales se rebajaron de forma que el estudio fuera abordable.

5.3 Análisis de la planificación y la metodología

La planificación inicial sufrió cambios a lo largo de la realización de las entregas 2 y 3 del informe de seguimiento y resultados. Se ha detallado en el apartado 1.4 Planificación del trabajo, cómo ha ido evolucionando la planificación. La causa principal del retraso en la planificación fue debida a la obtención de las muestras tumor-normal.

La metodología planteada inicialmente se ha mantenido, consistiendo en el estudio individual de cada uno de los programas a comparar: Mutect2, SomaticSniper, Varscan2 y Strelka2; el análisis de los métodos de *calling* que implementan; la prueba de los programas con datos de secuencias de cáncer utilizando el genoma de referencia hg38 y la comparativa de los resultados obtenidos tras las pruebas. Si bien es cierto que la comparativa no se realizó al final del análisis de los cuatro métodos; sino que se realizó una primera comparativa en cuanto se tuvieron resultados de los dos primeros métodos que se probaron, Mutect2 y VarScan y luego se completó con los resultados obtenidos con SomaticSniper y Strelka2.

5.4 Líneas de trabajo futuro

Como principal línea de trabajo futuro se plantea la generación de múltiples muestras virtuales tumor-normal, que permitan hacer un análisis

más completo y una comparativa de los métodos a través de gráficas relacionadas con la especificidad y sensibilidad, en la línea de lo que se muestra en los estudios científicos detallados en los artículos que se han consultado.

También sería interesante profundizar en los parámetros de las diferentes herramientas de detección de variantes somáticas y cómo los ajustes en estos parámetros afectan a la detección de variantes y por tanto a la comparativa entre los métodos que aplican estas herramientas.

6 Glosario

Alelo (*Allele*): un alelo es cada una de las dos o más versiones de un gen. Un individuo hereda dos alelos para cada gen, uno del padre y el otro de la madre. Los alelos se encuentran en la misma posición dentro de los cromosomas homólogos. Si los dos alelos son idénticos, el individuo es homocigoto para este gen.

Alelo alternativo (*Alternative allele*): se refiere a cualquier base, diferente de la de referencia, que se encuentra en un determinado lugar.

Alelo de referencia (*Reference allele*): se refiere a la base que se encuentra en el genoma de referencia.

Bases nitrogenadas: compuesto químico nitrogenado que constituye los ácidos nucleicos que conforman el ADN. Son: Adenina-Timina, Guanina-Citosina.

Cobertura (*Coverage*): este término relacionado con la secuenciación genética tiene que ver con la calidad de los datos obtenidos cuando se secuencian los genes. Cuanta más cobertura, más calidad, ya que la cobertura se refiere al número de veces que la máquina secuenciadora secuencia el genoma. Cuantas más pasadas haga la máquina menos errores potenciales habrá. De esta forma cuantas más veces se secuencia un genoma, más precisos son los datos obtenidos. Cuando se obtienen datos de secuencias genómicas estos suelen indicar la cobertura, la

cobertura se indica con un número seguido de una 'x'. Por ejemplo: WGS 30x, indicaría que se trata de un conjunto de datos del genoma al completo con una cobertura de 30x, es decir se ha secuenciado el genoma completo de media 30 veces.

Conductor (*Driver*): mutación de tipo Driver (*Driver mutation*): mutaciones en el ADN responsables del desarrollo y progresión del cáncer.

Diploide (*diploid)*: células diploides, células que contienen dos copias de cada cromosoma, normalmente una de la madre y otra del padre. Las células somáticas también se las denomina *diploid cells*. Las células de tipo *diploid* en los seres humanos tienen 46 cromosomas 2n, 23*2.

Fenotipo: es el rasgo que podemos observar, es el reflejo de nuestro genotipo. Si dos individuos tienen distinto fenotipo también tendrán distinto genotipo.

Genotipo: código genético de las células de un organismo. Información genética heredada, se expresa en forma de fenotipo.

Haploide (*haploid*): células haploides, son aquellas que contienen un conjunto simple de cromosomas. Por ejemplo: óvulo y espermatozoide, a estas células también se las denomina gametos. Las células de tipo *haploid* en el ser humano (óvulo y esperma) tienen 23 cromosomas.

Haplotipo (*haplotype*): "Un haplotipo es, en su sentido más general, un conjunto de variaciones de ADN a lo largo de un cromosoma que tienden a ser heredados juntos porque están muy próximos".

Homocigoto/Heterocigoto: Cada alelo de un gen, en un organismo diploide, se hereda de cada progenitor. Si ambos alelos para ese gen son iguales, entonces el organismo es homocigoto, si son distintos entonces es heterocigoto.

Pasajero (*Passenger*): mutación de tipo Passenger (*Passenger mutation*): mutaciones en el ADN que no son responsables del desarrollo y progresión del cáncer.

SNP: polimorfismo de nucleótido simple (single nucleotid Polimorfism).

Variant Calling: descubrimiento de variaciones/mutaciones**,** método para el descubrimiento de variantes o mutaciones en los genes.

Variante somática o mutación somática (*Somatic vatiation/mutation*): se trata de alteraciones en el ADN que se producen después del nacimiento. Se pueden dar en cualquier célula, excepto en el óvulo y el esperma.

***WES (Whole exome sequencing)*:** secuenciación del exoma completo, es una técnica genómica para secuenciar todas las regiones de genes que codifican proteínas en un genoma (exones). Primero hay que seleccionar solamente un subconjunto del ADN que codifique proteínas, los denominados exones, luego hay que secuenciar estos exones utilizando alguna tecnología de secuenciación. Los seres humanos tienen aproximadamente 180,000 exones, que representa aproximadamente el 1% de todo el genoma humano, estos 180.000 exones tienen aproximadamente 30 millones de bases de pares. Uno de los objetivos de llevar a cabo este tipo de secuenciación es detectar variantes genéticas que alteran la secuenciación de proteínas y hacerlo a un coste inferior que si usáramos la secuenciación completa del genoma WGS (Whole Genome Sequencing).

7 Bibliografía

1. Dees ND, Zhang Q, Kandoth C, Wendl MC, Schierding W, Koboldt DC, et al. MuSiC: Identifying mutational significance in cancer genomes. Genome Res [Internet]. 2012 Aug 1 [cited 2020 Sep 25];22(8):1589–98. Available from: http://www.genome.org/cgi/doi/10.1101/gr.134635.111.
2. Tian R, Basu MK, Capriotti E. ContrastRank: A new method for ranking putative cancer driver genes and classification of tumor samples. Bioinformatics [Internet]. 2014 Sep 1 [cited 2020 Sep 25];30(17):572–8. Available from: https://academic.oup.com/bioinformatics/article/30/17/i572/201025
3. Tian R, Basu MK, Capriotti E. Computational methods and resources for the interpretation of genomic variants in cancer. BMC Genomics [Internet]. 2015 Jun 18 [cited 2020 Sep 25];16(8):1–19. Available from: https://link.springer.com/articles/10.1186/1471-2164-16-S8-S7
4. Sims D, Sudbery I, Ilott NE, Heger A, Ponting CP. Sequencing depth and coverage: Key considerations in genomic analyses [Internet]. Vol. 15, Nature Reviews Genetics. Nature Publishing Group; 2014 [cited 2020 Oct 25]. p. 121–32. Available from: https://www.nature.com/articles/nrg3642
5. Chen Z, Yuan Y, Chen X, Chen J, Lin S, Li X, et al. Systematic comparison of somatic variant calling performance among different sequencing depth and mutation frequency. Sci Rep [Internet]. 2020 Dec 1 [cited 2020 Oct 13];10(1):1–9. Available from: www.nature.com/scientificreports
6. Benjamin D, Sato T, Cibulskis K, Getz G, Stewart C, Lichtenstein L. Calling Somatic SNVs and Indels with Mutect2. 2019; Available from: https://doi.org/10.1101/861054
7. Larson DE, Harris CC, Chen K, Koboldt DC, Abbott TE, Dooling DJ, et al. Somaticsniper: Identification of somatic point mutations in whole genome sequencing data. Bioinformatics [Internet]. 2012 Feb 1 [cited 2020 Oct 21];28(3):311–7. Available from: http://gmt.genome.wustl.edu/somatic-sniper/current/,
8. Koboldt DC, Zhang Q, Larson DE, Shen D, McLellan MD, Lin L, et al. VarScan 2: Somatic mutation and copy number alteration discovery in cancer by exome sequencing. Genome Res [Internet]. 2012 Mar 1 [cited 2020 Oct 21];22(3):568–76. Available from: http://www.genome.org/cgi/doi/10.1101/gr.129684.111.
9. Kim S, Scheffler K, Halpern AL, Bekritsky MA, Noh E, Källberg M, et al. Strelka2: Fast and accurate variant calling for clinical sequencing applications. bioRxiv [Internet]. 2017 [cited 2020 Dec 27]; Available from: https://doi.org/10.1101/192872
10. Auton A, Abecasis GR, Altshuler DM, Durbin RM, Bentley DR, Chakravarti A, et al. A global reference for human genetic variation [Internet]. Vol. 526, Nature. Nature Publishing Group; 2015 [cited 2020 Nov 15]. p. 68–74. Available from: https://www.nature.com/articles/nature15393
11. Cibulskis K, Lawrence MS, Carter SL, Sivachenko A, Jaffe D, Sougnez C, et al. Sensitive detection of somatic point mutations in impure and heterogeneous cancer samples. Nat Biotechnol. 2013;31(3):213–9.
12. Benjamin D, Sato T, Cibulskis K, Getz G, Stewart C, Lichtenstein L. Calling Somatic SNVs and Indels with Mutect2. bioRxiv [Internet]. 2019 Dec 2

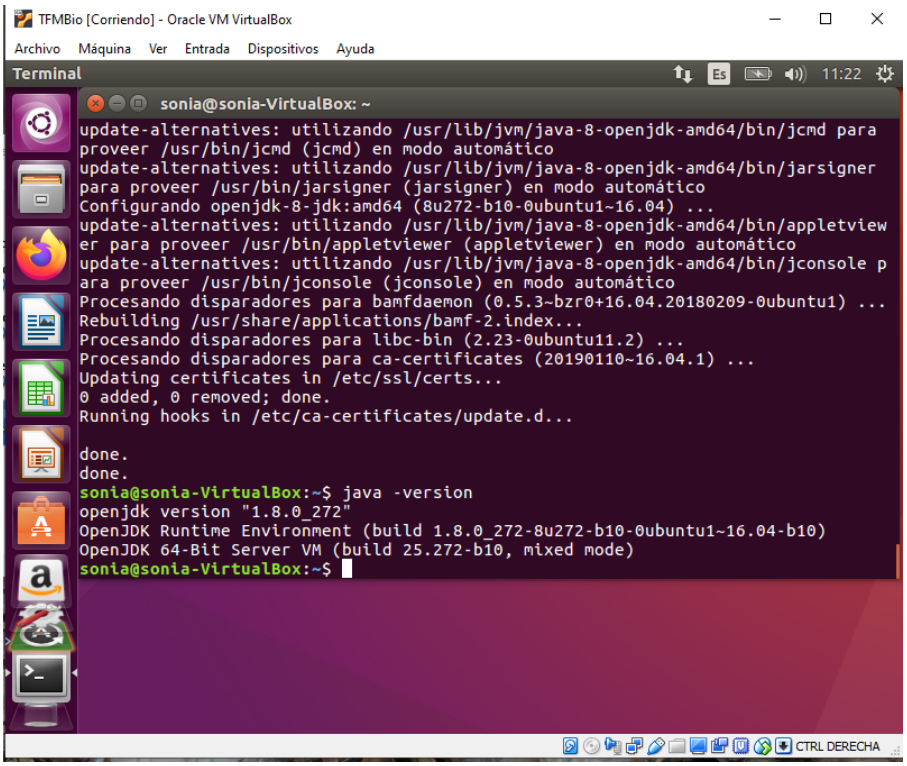
[cited 2020 Nov 2];861054. Available from: https://doi.org/10.1101/861054

13. Li H, Ruan J, Durbin R. Mapping short DNA sequencing reads and calling variants using mapping quality scores. Genome Res. 2008 Nov 1;18(11):1851–8.
14. Saunders CT, Wong WSW, Swamy S, Becq J, Murray LJ, Cheetham RK. Strelka: Accurate somatic small-variant calling from sequenced tumor-normal sample pairs. Bioinformatics. 2012 Jul;28(14):1811–7.

8 Anexos

8.1 Anexo I. Instalación de paquetes requeridos para seguir las mejores prácticas del GATK[34].

- Instalar el Java Runtime Environment (JRE) en su versión 1.8.



- Instalar: BWA, SAMtools, Picard, GATK, IGV, RStudio/R y las librerías para gráficos de R ggplot2 y gsalib.

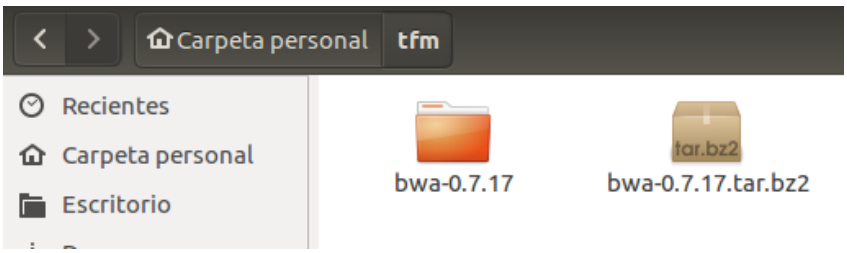
a. BWA

BWA contiene la implementación del algoritmo de alineamiento de Burrows-Wheller. Este algoritmo permite el alineamiento de secuencias con baja divergencia enfrentadas a largas secuencias del genoma, como por ejemplo el genoma humano. Contiene 3 algoritmos: *BWA-backtrack, BWA-SW y BWA-MEM.* El primero está diseñado para secuencias Illumina[35] con lecturas de hasta 100bp. Los otros dos son para lecturas más grandes entre 70bp y

[34]https://gatk.broadinstitute.org/hc/en-us/articles/360041320571--How-to-Install-all-software-packages-required-to-follow-the-GATK-Best-Practices
[35] www.illumina.com

1Mbp. Tanto BWA-SW como BWA-MEM son similares, siendo el último más rápido y preciso, incluso da mejores resultados con respecto a BWA-backtrack en secuencias Illumina con lecturas entre 70-100bp.

Para efectuar la instalación primero descargamos el fichero comprimido que contiene los ficheros binarios[36] y luego lo descomprimimos, nos crea la carpeta **bwa-0.7.17**



Compilar BWA:

```
sonia@sonia-VirtualBox:~/tfm/bwa-0.7.17$ make
```

Nos da este error, por no tener zlip:

```
fatal error: zlib.h: No existe el archivo o el directorio
```

Instalamos zlip con la sentencia **sudo apt-get install libz-dev** y repetimos el **make**.

Una vez compilado efectuamos una prueba:

```
sonia@sonia-VirtualBox:~/tfm/bwa-0.7.17$ ./bwa

Program: bwa (alignment via Burrows-Wheeler transformation)
Version: 0.7.17-r1188
Contact: Heng Li <lh3@sanger.ac.uk>

Usage:   bwa <command> [options]
```

[36] https://sourceforge.net/projects/bio-bwa/files/

Para poder ejecutar **bwa** desde cualquier ruta debemos añadirlo a la variable de entorno PATH. Podemos hacer que esto sea permanente incluyendo el comando:

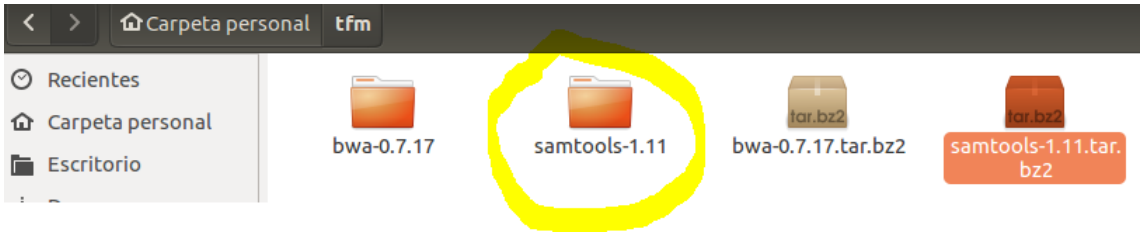
export PATH=$PATH:/home/sonia/tfm/bwa-0.7.17

al fichero /etc/profile

b. SAMtools

SAMTools es un conjunto de herramientas utilizadas para manipular los ficheros SAM, realizando operaciones como ordenamiento, unión, indexación, etc. El formato SAM (Sequence Aligment/Map) es un formato genérico que permite almacenar grandes alineaciones de secuencias de nucleótidos[37].

Para efectuar la instalación primero descargamos el fichero comprimido que contiene los binarios y luego lo descomprimimos, nos crea la carpeta **samtools-1.11**



Compilamos:

```
sonia@sonia-VirtualBox:~/tfm$ cd samtools-1.11
sonia@sonia-VirtualBox:~/tfm/samtools-1.11$ make
```

Durante la compilación nos da este error:

```
bam_tview_curses.c:41:20: fatal error: curses.h: No existe el archivo o el direc
torio
```

Es necesario instalar **ncurses** que es una librería utilizada para la programación de aplicaciones basadas en terminar. En Ubuntu es

[37] http://samtools.sourceforge.net/

necesario instalar los paquetes **libncurses5-dev** y **libncursesw5-dev.**

```
sonia@sonia-VirtualBox:~/tfm/samtools-1.11$ sudo apt-get install libncurses5-dev libncursesw5-dev
```

Repetimos el **make**, y nos da otro error:

```
cram/cram_io.c:53:19: fatal error: bzlib.h: No existe el archivo o el direc
compilation terminated.
```

Debemos instalar el paquete libbz2-dev:

```
sonia@sonia-VirtualBox:~/tfm/samtools-1.11$ sudo apt-get install libbz2-dev
```

Repetimos el **make** y otro error:

```
cram/cram_io.c:57:18: fatal error: lzma.h: No existe el archivo o el directorio
```

Debemos instalar el paquete **liblzma-dev**

```
sonia@sonia-VirtualBox:~/tfm/samtools-1.11$ sudo apt-get install liblzma-dev
```

Un nuevo error en el siguiente intento de compilación:

```
hfile_libcurl.c:47:23: fatal error: curl/curl.h: No existe el archivo o el directorio
```

Debemos instalar el paquete libcurl4-openssl-dev

```
sonia@sonia-VirtualBox:~/tfm/samtools-1.11$ sudo apt-get install libcurl4-openssl-dev
```

Repetimos el **make** y en esta ocasión no da ningún error.

Una vez compilado hacemos una prueba:

```
sonia@sonia-VirtualBox:~/tfm/samtools-1.11$ ./samtools

Program: samtools (Tools for alignments in the SAM format)
Version: 1.11 (using htslib 1.11)

Usage:   samtools <command> [options]
```

Añadimos **samtools** a la variable PATH, repitiendo el mismo procedimiento que seguimos con **bwa**.

c. Picard

Picard[38] es un conjunto de comandos que permiten manipular ficheros en formato SAM/BAM/CRAM y VCF[39]. Está programado en java, por lo que necesitamos JRE que ya instalamos en su momento.

Descargamos picard.jar de la ruta http://software.broadinstitute.org/software/igv/FileFormats.

Par poder ejecutar los distintos comandos incluidos en Picard debemos usar la sintaxis:

java -jar picard.jar <ToolName>

Podemos ver los comandos/programas disponibles en Picard ejecutando la siguiente orden:

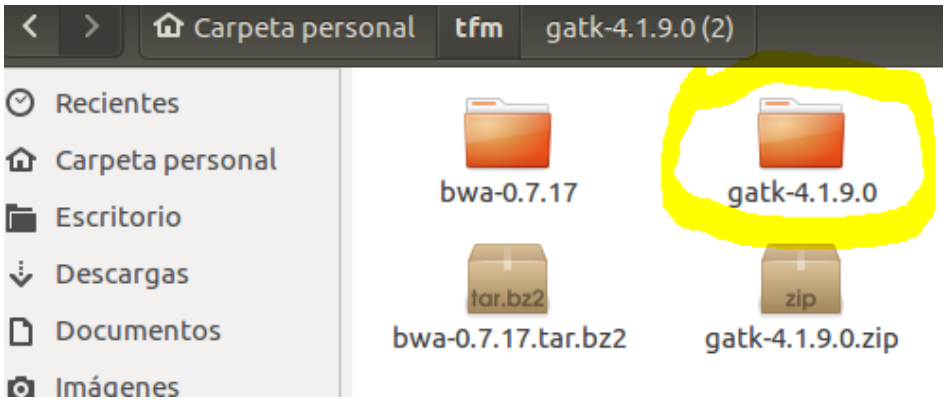
```
sonia@sonia-VirtualBox:~/tfm$ java -jar picard.jar -h
```

d. GATK

GATK son las siglas de Genome Analysis Toolkit. Este conjunto de herramientas de análisis del genoma se centra principalmente en el descubrimiento de variantes y en el genotipado[40].

Si se va a utilizar GATK para uso comercial hay que pagar una licencia. Esto no aplica en nuestro caso. Descargamos GATK[41], descomprimimos el archivo y dentro de la carpeta que nos crea

[38] broadinstitute.github.io/picard/
[39] http://software.broadinstitute.org/software/igv/FileFormats
[40] https://gatk.broadinstitute.org/hc/en-us
[41] https://github.com/broadinstitute/gatk/releases



encontramos la utilidad **gatk** que nos permitirá ejecutar los distintos programas disponibles dentro del Toolkit (uno de ellos es Mutect2). Para ver un listado de todos los programas podemos ejecutar la orden:

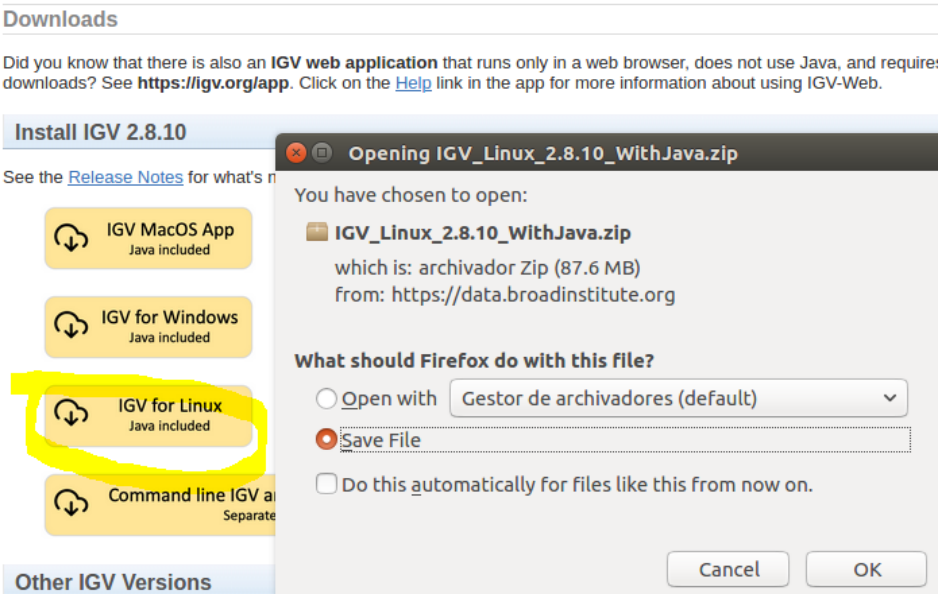
```
sonia@sonia-VirtualBox:~/tfm/gatk-4.1.9.0$ ./gatk --list
Using GATK jar /home/sonia/tfm/gatk-4.1.9.0/gatk-package-4.1.9.0-local.jar
Running:
    java -Dsamjdk.use_async_io_read_samtools=false -Dsamjdk.use_async_io_write_samtools=true -Dsamjd
k.use_async_io_write_tribble=false -Dsamjdk.compression_level=2 -jar /home/sonia/tfm/gatk-4.1.9.0/ga
tk-package-4.1.9.0-local.jar --help
USAGE:  <program name> [-h]

Available Programs:
--------------------------------------------------------------------------------------
Base Calling:                                Tools that process sequencing machine data, e.g. Il
lumina base calls, and detect sequencing level attributes, e.g. adapters
    CheckIlluminaDirectory (Picard)          Asserts the validity for specified Illumina basecal
ling data.
```

e. IGV

IGV son las siglas de Integrated Genomics Viewer. Se trata de un navegador que permite visualizar ficheros BAM, VCF y otra información genómica. Dispone de una interfaz gráfica fácil de utilizar y se puede descargar de forma gratuita[42].

Nos descargamos IGV for Linux (Java included):

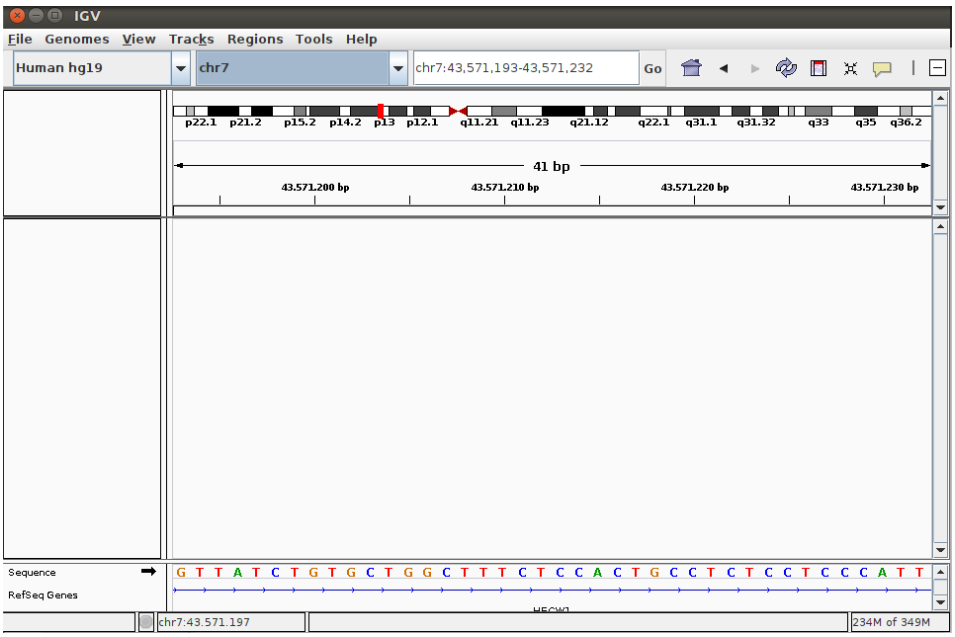
[42] https://software.broadinstitute.org/software/igv/



Una vez descargado y descomprimido, podemos ejecutarlo con el script igv.sh:

```
sonia@sonia-VirtualBox:~/tfm/IGV_Linux_2.8.10$ ./igv.sh
```

Esto nos abrirá el programa:



f. RStudio IDE and R libraries ggplot2 and gsalib

RStudio[43] es una interfaz gráfica que nos permite acceder a las librerías de R. R es un lenguaje de programación y también un conjunto de librerías específicas para la realización de cálculos estadísticos y gráficos. Antes de instalar RStudio debemos instalar R[44].

Comenzamos añadiendo el repositorio desde el que descargar R dentro de Ubuntu. Este repositorio es mantenido por CRAN[45].

Editar el fichero **/etc/apt/sources.list** y añadir esta línea al final:

deb https://cloud.r-project.org/bin/linux/ubuntu xenial-cran35/

Luego actualizar los paquetes con:

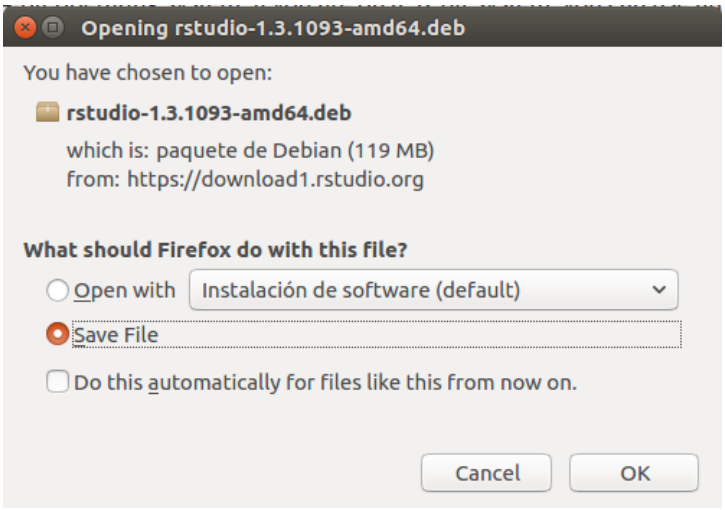
sudo apt-get update

Y por último instalar r:

sudo apt-get install r-base

A continuación, descargamos el paquete .deb de RStudio Desktop en su versión gratuita de la página https://rstudio.com/products/rstudio/download/#download.

[43] https://rstudio.com/products/rstudio/
[44] https://cran.r-project.org/
[45] CRAN (The Comprehensive R Archive Network), https://cran.r-project.org/



Una vez descargado lo instalamos con el comando gdebi, si este comando no está disponible en la distribución Ubuntu que estamos usando lo instalamos:

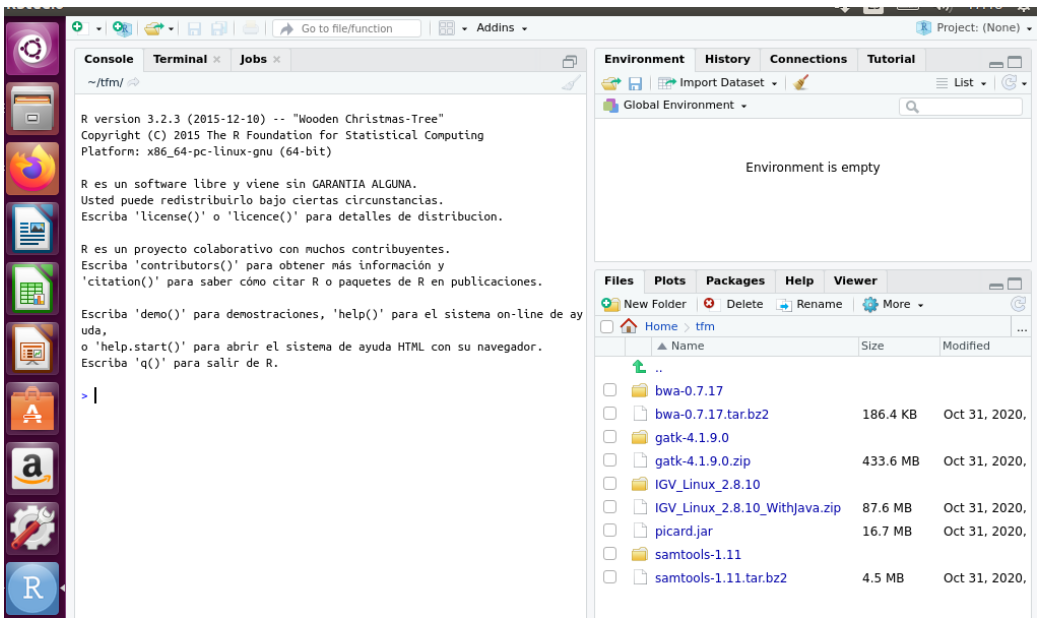
```
sonia@sonia-VirtualBox:~/Descargas$ sudo apt install gdebi-core
```

Luego instalamos el paquete de RStudio con gdebi:

```
sonia@sonia-VirtualBox:~/Descargas$ sudo gdebi rstudio-1.3.1093-amd64.deb
```

Ya podemos arrancar RStudio:

```
sonia@sonia-VirtualBox:~/tfm$ rstudio
```



Para incorporar la librería **ggplot2**[46]**,** una librería para la realización de gráficos, realizamos lo siguiente dentro de la ventana de comandos de RStudio:

```
> install.packages("ggplot2")
```

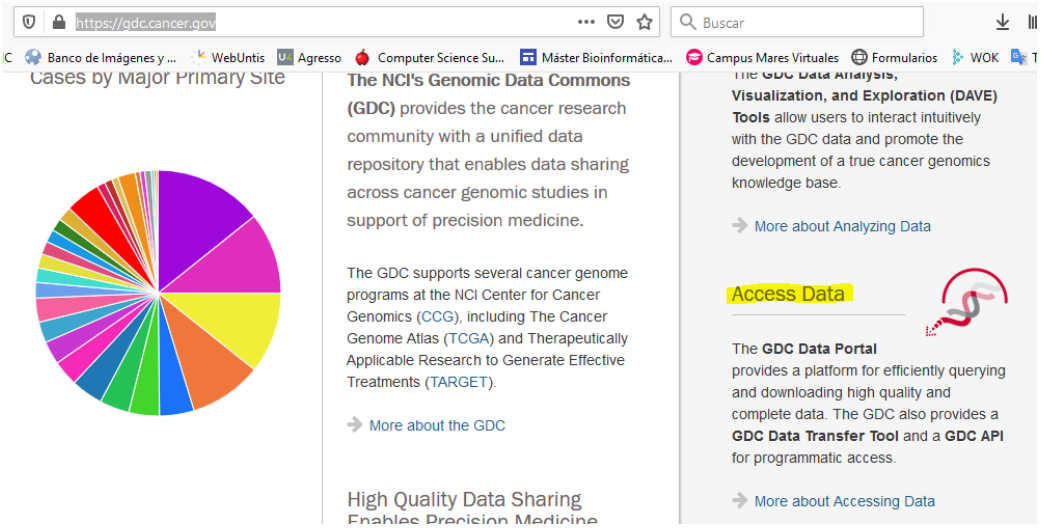
Ahora instalamos el paquete **gsalib**. Este paquete contiene funcione que son usadas por GATK para realizar tablas y gráficos. Para incorporar la librería **gsalib**[47]:

```
> install.packages("gsalib")
```

[46] https://cran.r-project.org/web/packages/ggplot2/index.html
[47] https://cran.r-project.org/web/packages/gsalib/index.html

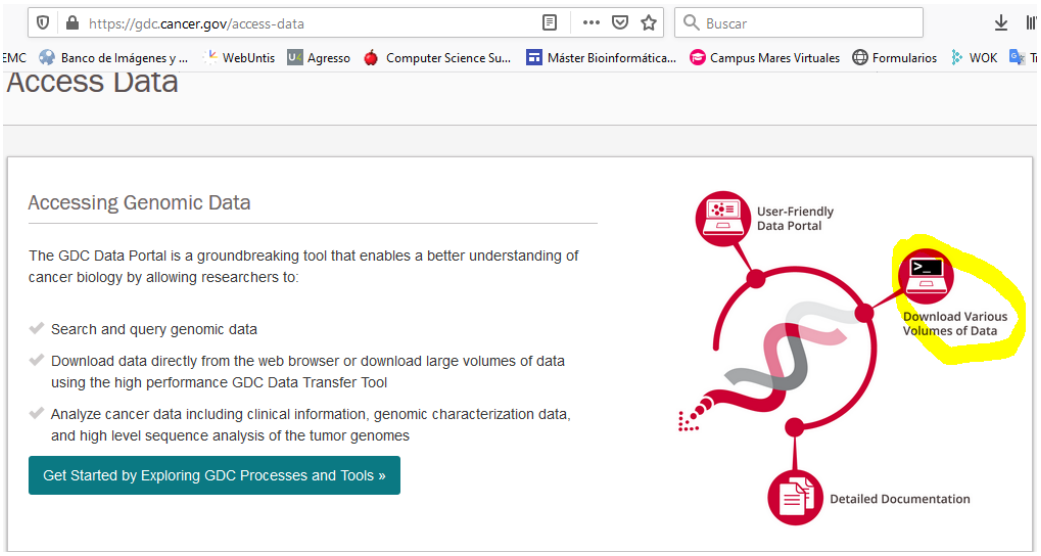
8.2 Anexo II. Pasos para el acceso a datos en el *Genomic Data Commons*.

Dentro de la página https://gdc.cancer.gov/, nos dirigimos a la zona ***Access Data***:

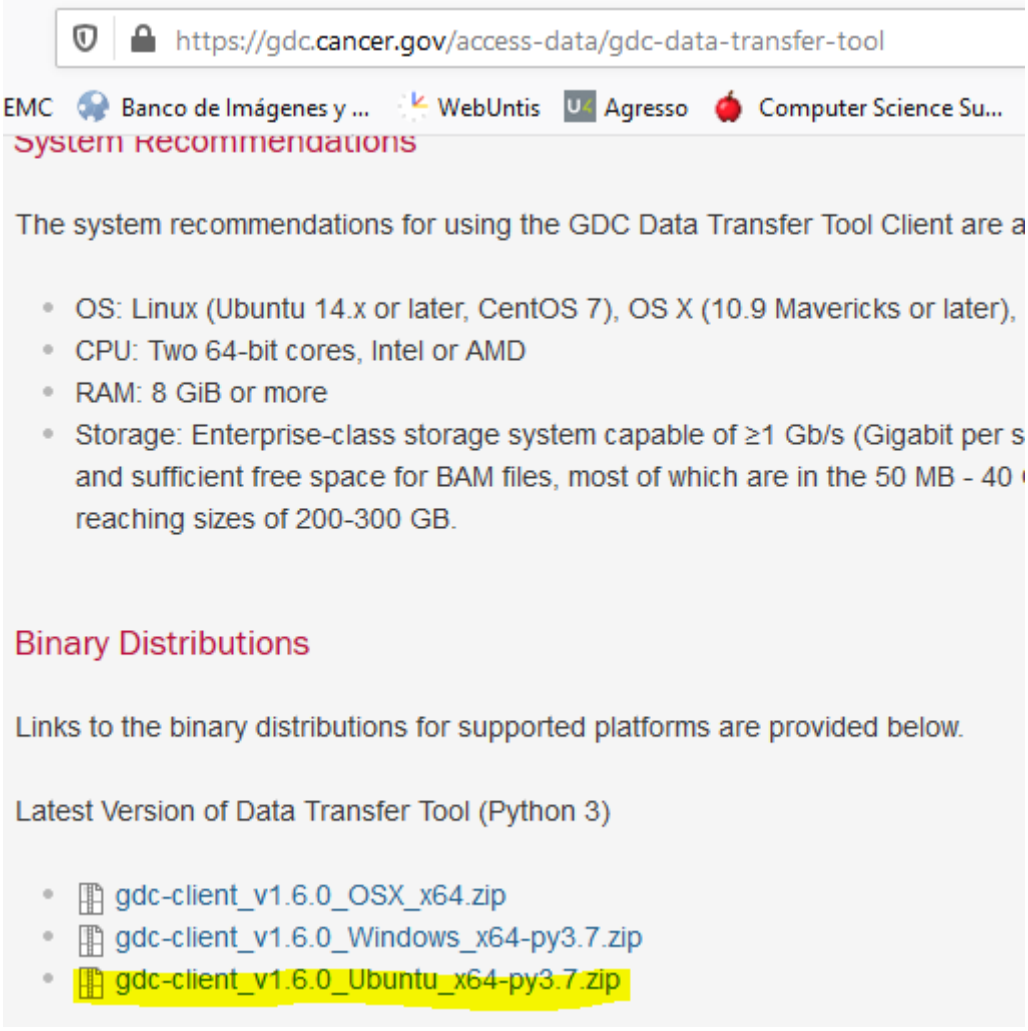


Y seleccionamos *More about Accessing Data*.

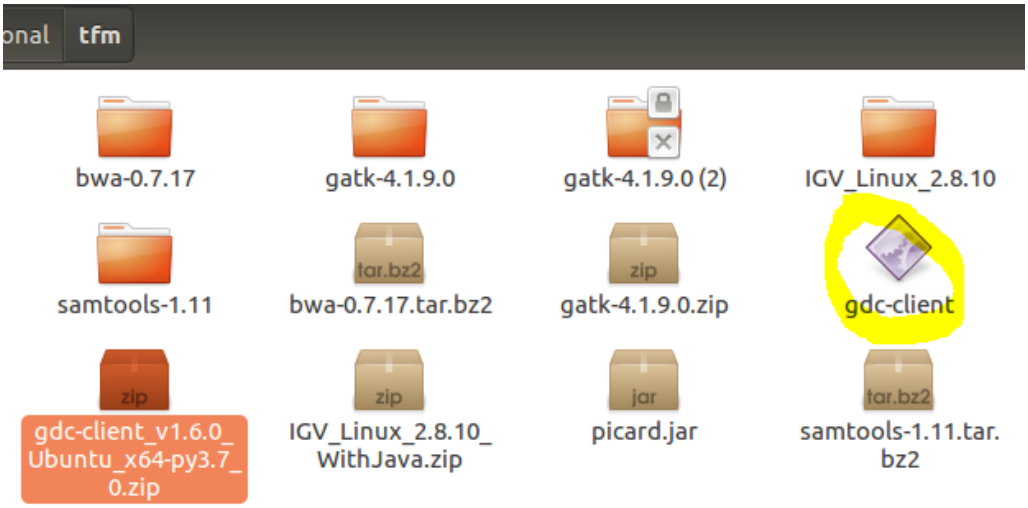
En la página que aparece, seleccionamos *Download Various Volumnes of Data*.



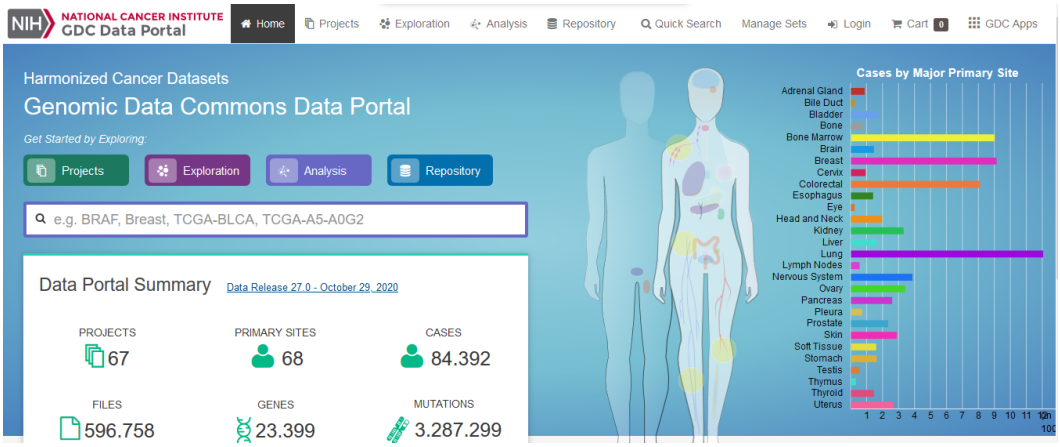
En la página que aparece, descargamos la herramienta de transferencia de datos (*GDC Data Transfer Tool) usando la distribución binaria de Ubuntu.*



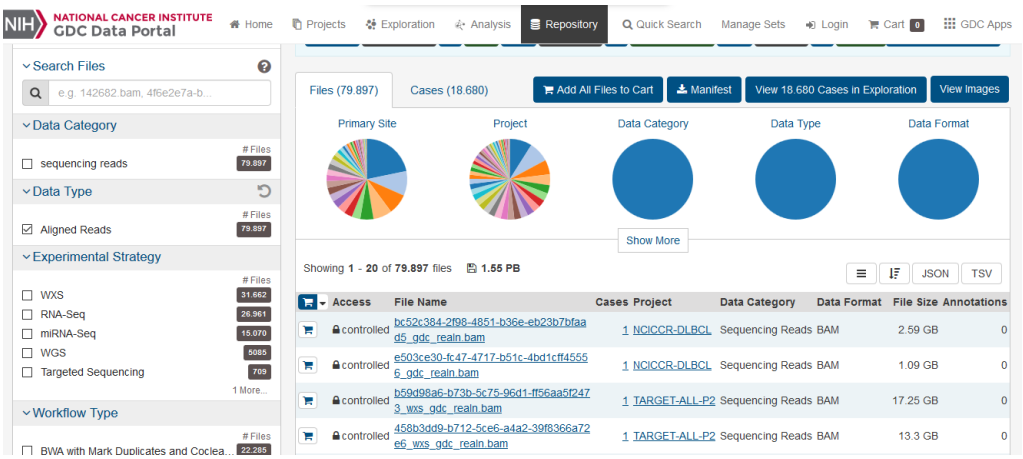
Una vez descargada la distribución la descomprimimos y ya tenemos a nuestra disposición el cliente *GDC Data Transfer:*



Seguimos las indicaciones de la guía de usuario de la *GDC Data Transfer Tool*[48]*, donde se indica en el capítulo Preparing for Data Download and Upload*, que lo primero que debemos hacer es descargar el fichero que contiene el manifiesto. En el manifiesto figurará la lista de ficheros que vamos a descargar o identificadores de los mismos. Este manifiesto será utilizado por la herramienta para hacer la descarga. Para obtener el manifiesto debemos dirigirnos a *Genomic Data Commons Data Portal*[49].



Una vez aquí, pulsamos en el botón *Repository.*

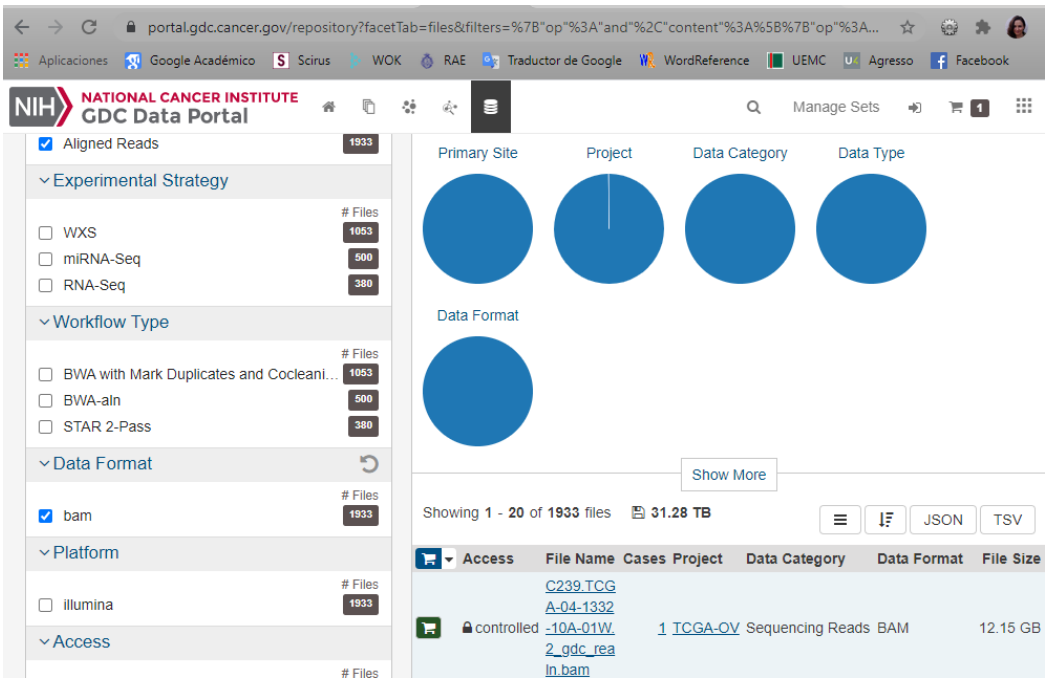


Una vez en el repositorio debemos filtrar con las opciones del panel izquierdo para quedarnos con los ficheros .bam que deseemos añadir a la lista de ficheros a descargar. Para elegir los ficheros se va pulsando en el

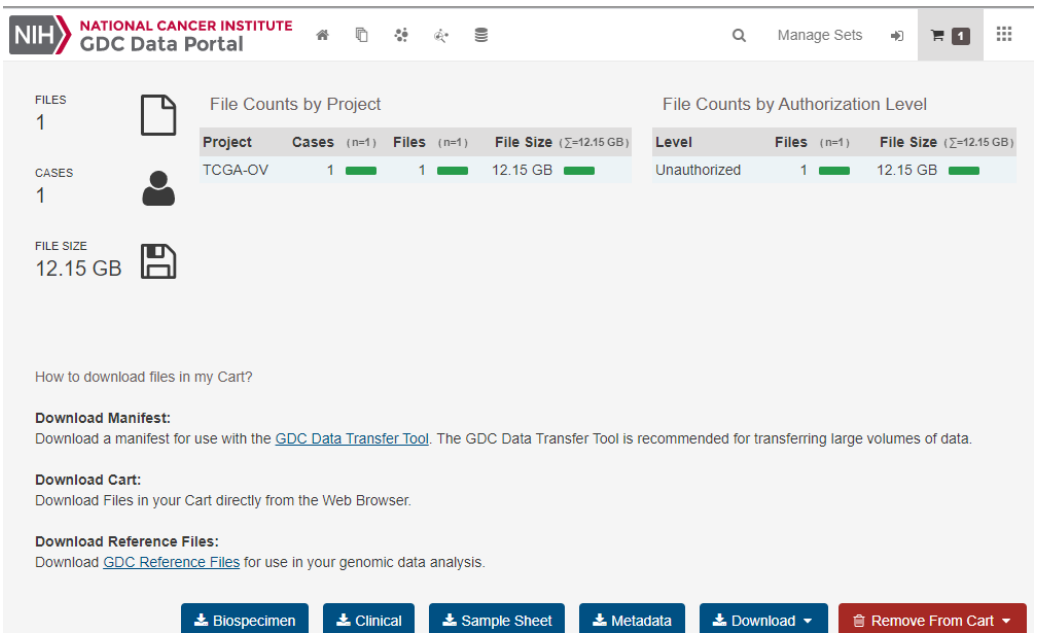
[48] https://docs.gdc.cancer.gov/Data_Transfer_Tool/Users_Guide/Getting_Started/
[49] https://portal.gdc.cancer.gov/

botón con forma de carrito de la compra que aparece a la izquierda del nombre del fichero.

Para hacer el test vamos a seleccionar el primer fichero .bam que aparece cuando filtramos por "Aligned Reads" y ".bam":



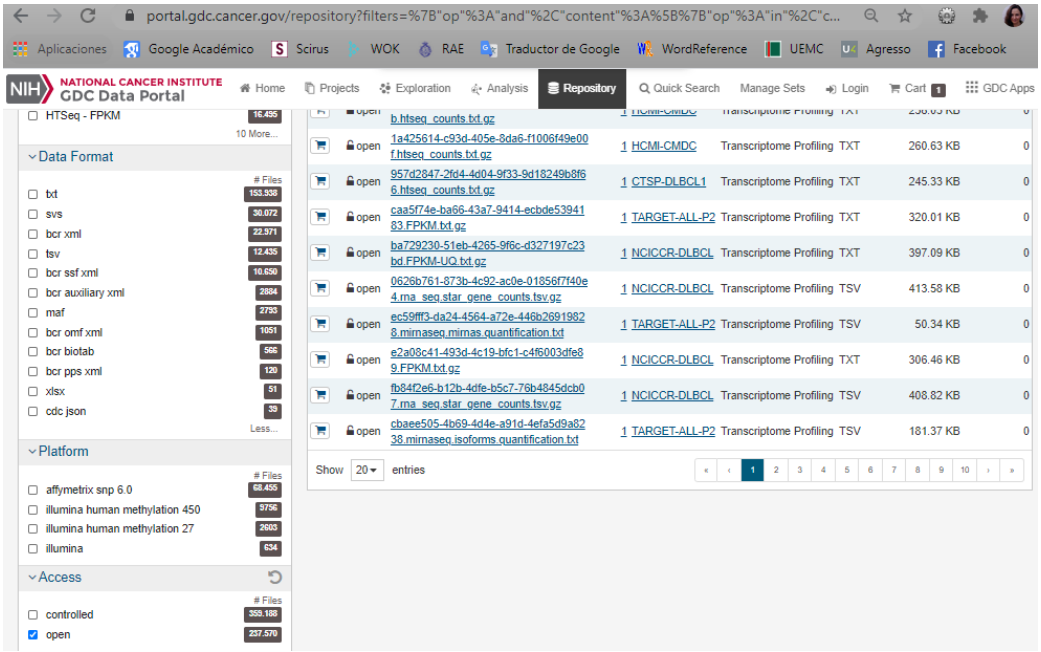
Después pulsamos en el botón  que aparece en la esquina superior derecha. Esto nos da acceso a la siguiente ventana:



En ella se ven los ficheros seleccionados. Para descargar el manifiesto pulsamos el botón *Download*, y seleccionamos *manifest*.

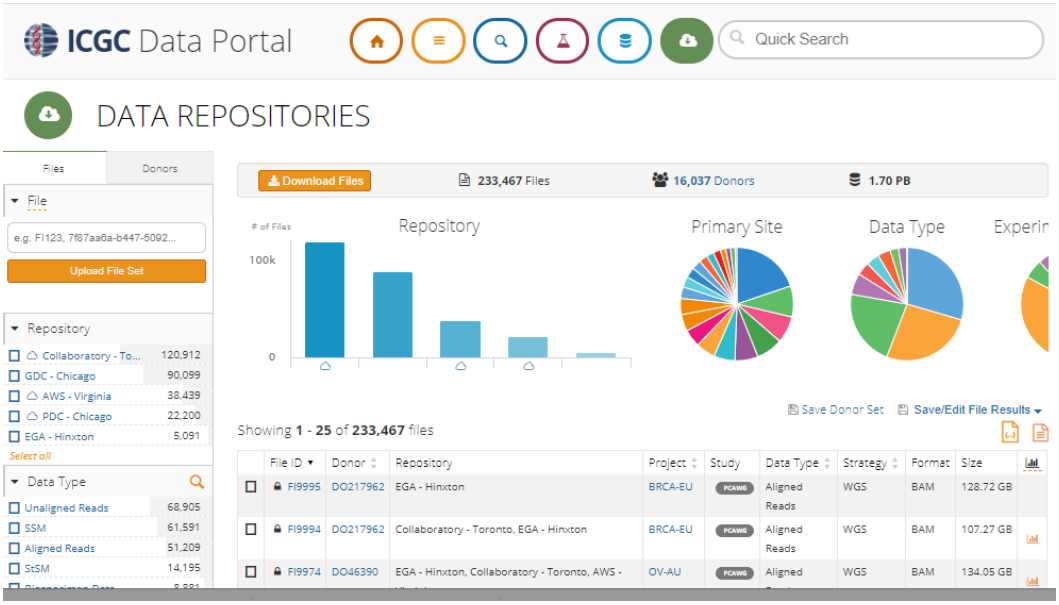
Lo siguiente que debemos hacer es obtener el *token* de autenticación. Primero debemos logearnos. Pero para conseguir un usuario con acceso debemos pertenecer a una organización que esté registrada en eRA (https://grants.nih.gov/grants/guide/notice-files/NOT-OD-07-003.html) y una vez registrados aquí se necesita otra autorización para acceder a los datos (https://gdc.cancer.gov/access-data/obtaining-access-controlled-data), algo que se sale de nuestro alcance.

Aunque GDC proporciona datos en abierto, ninguno de los ficheros es .bam. Al aplicar el filtro *open*, vemos el resultado de los tipos de ficheros que se pueden descargar.

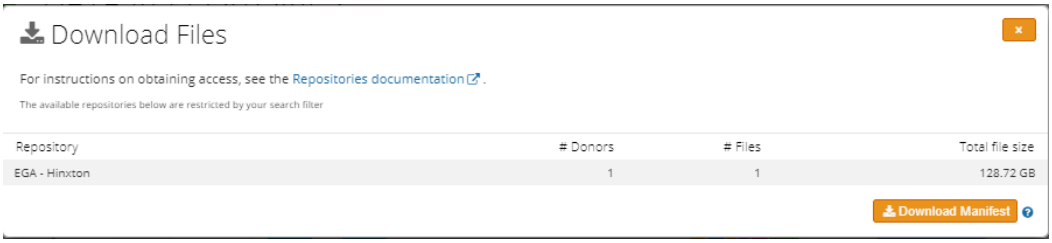


8.3 Anexo III. Pasos para el acceso a datos en el *International Cancer Genome Consortium.*

Intentamos obtener los datos desde el repositorio[50] del portal de ICGC.

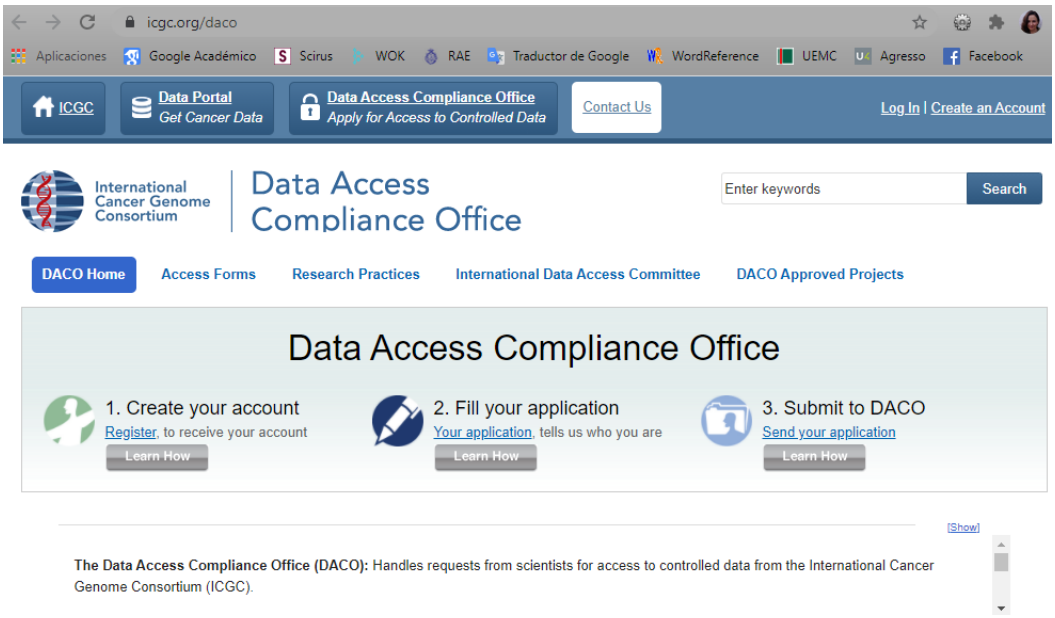


Si seleccionamos algún fichero BAM, que son los que nos interesan, y pulsamos en **Download Files**, nos muestra una ventana informando de que el acceso es restringido.



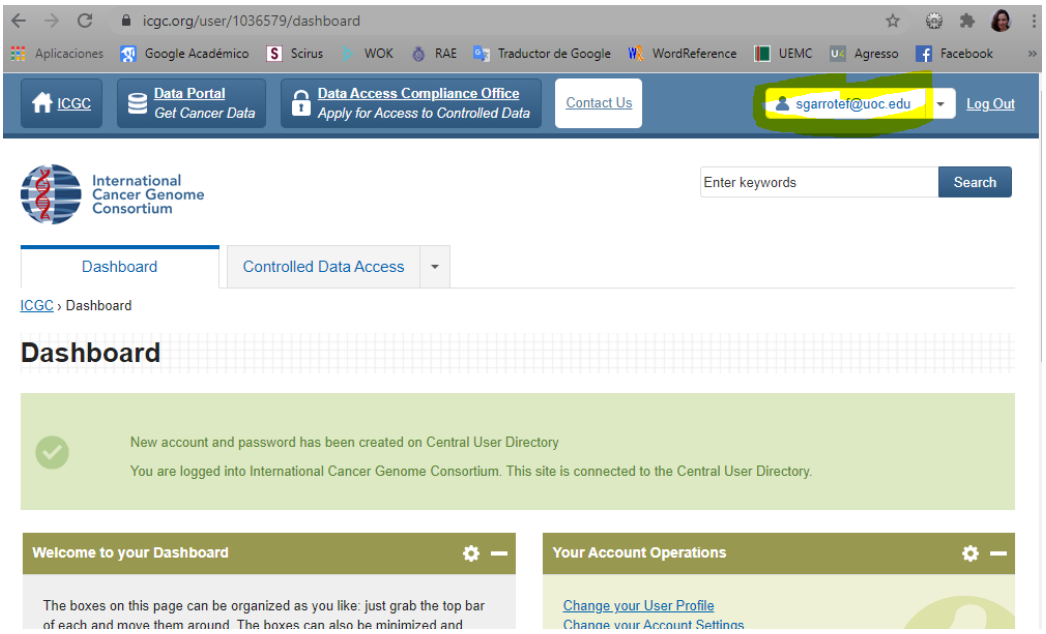
Por lo tanto se requiere autorización para el acceso a ficheros de tipo .bam. Los pasos a seguir para obtener la autorización se indican aquí:

[50] https://dcc.icgc.org/repositories

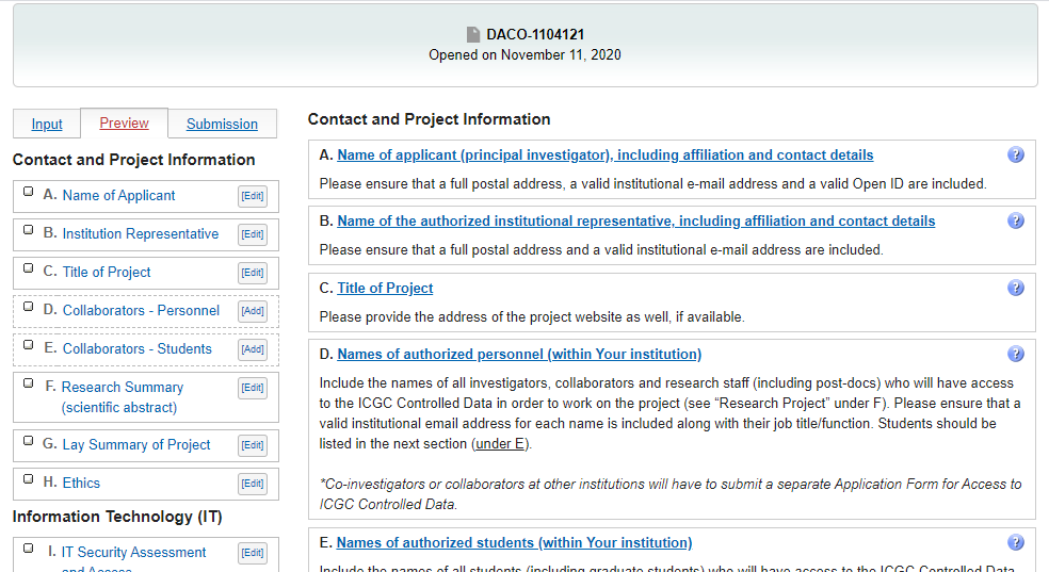


Primero creamos una cuenta en la que únicamente se solicita nuestra dirección de correo electrónico. Se envían por correo la contraseña y el resto de las instrucciones.

Una vez registrados podemos acceder:



Realizamos el siguiente paso *Controlled Data Access/Data Access Compliance Office – Access Forms.* Y volvemos a encontrarnos con el mismo problema que con GDC, necesitamos estar adscritos a una institución y aun proyecto concreto, pues se nos piden todos estos datos:



DACO-1104121
Opened on November 11, 2020

Input | Preview | Submission

Contact and Project Information

- A. Name of Applicant [Edit]
- B. Institution Representative [Edit]
- C. Title of Project [Edit]
- D. Collaborators - Personnel [Add]
- E. Collaborators - Students [Add]
- F. Research Summary (scientific abstract) [Edit]
- G. Lay Summary of Project [Edit]
- H. Ethics [Edit]

Information Technology (IT)

- I. IT Security Assessment [Edit]

Contact and Project Information

A. Name of applicant (principal investigator), including affiliation and contact details

Please ensure that a full postal address, a valid institutional e-mail address and a valid Open ID are included.

B. Name of the authorized institutional representative, including affiliation and contact details

Please ensure that a full postal address and a valid institutional e-mail address are included.

C. Title of Project

Please provide the address of the project website as well, if available.

D. Names of authorized personnel (within Your institution)

Include the names of all investigators, collaborators and research staff (including post-docs) who will have access to the ICGC Controlled Data in order to work on the project (see "Research Project" under F). Please ensure that a valid institutional email address for each name is included along with their job title/function. Students should be listed in the next section (under E).

**Co-investigators or collaborators at other institutions will have to submit a separate Application Form for Access to ICGC Controlled Data.*

E. Names of authorized students (within Your institution)

8.4 Anexo IV. Pasos para la obtención de las muestras virtuales tumor-normal.

Debido a la capacidad de cálculo del ordenador portátil utilizado durante el desarrollo de este proyecto no fue viable trabajar con muestras grandes. Se trata de un equipo con sistema operativo Windows, con un microprocesador Intel i7, con 8Gb de RAM; pero se operaba en realidad con una máquina virtual con Ubuntu, configurada con 4Gb de RAM, para poder ejecutar las distintas herramientas como samtools, mutect, etc..

Inicialmente se optó por buscar ficheros .bam de un tamaño inferior a los 10Gb para ver si podían ser manejados por la máquina en tiempos prudenciales. Se descargó un fichero .bam de 8Gb del cromosoma 21 -chr21- perteneciente al 1000 Genomes Project. Se comprobó enseguida, con la ejecución de la orden:

```
./samtools sort ../NA12878.chr21.bam -o ../sorted.bam
```

que permite ordenar los alineamientos en el .bam, que la máquina iba a tardar demasiado tiempo en realizar cualquier tratamiento con este fichero. Después de 20 minutos, se canceló manualmente el comando pues aún no había concluido.

Se tomó la decisión de bajar el tamaño del fichero base para la confección de las muestras virtuales y optar por tamaños del orden de Mb. Se encontró un fichero .bam del chr11 de 368Mb[51] procedente de 1000 Genomes Project.

La ordenación de este fichero tardó 3 minutos, por lo que se decidió tomarlo como base.

Lo siguiente fue dividir el fichero .bam en varias partes, de forma que consiguiéramos muestras más pequeñas. Usamos la herramienta **picard**. El fichero tiene un total de 3861579 líneas en la zona de alineamiento y 104 líneas en la zona de cabecera:

[51] https://www.internationalgenome.org/data-portal/sample/HG01781

```
sonia@sonia-VirtualBox:~/tfm$ sam/samtools view chr11.bam | wc -l
3861579

sonia@sonia-VirtualBox:~/tfm$ sam/samtools view -H chr11.bam | wc -l
104
```

Decidimos crear 3 divisiones:

```
sonia@sonia-VirtualBox:~/tfm$ java -jar picard.jar
SplitSamByNumberOfReads I=chr11.bam OUTPUT=chr11Split
TOTAL_READS_IN_INPUT=3861579 SPLIT_TO_N_READS=1287193
```

Comprobamos que todas las divisiones tienen el mismo número de líneas:

```
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view -H
shard_0001.bam | wc -l
104
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view -H
shard_0002.bam | wc -l
104
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view -H
shard_0003.bam | wc -l
104
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view
shard_0001.bam | wc -l
1287193
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view
shard_0002.bam | wc -l
1287193
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view
shard_0003.bam | wc -l
1287193
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view -h
shard_0001.bam | wc -l
1287297
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view -h
shard_0002.bam | wc -l
1287297
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools view -h
shard_0003.bam | wc -l
1287297
```

Ahora haremos una prueba con mutect2 simulando que la muestra tumoral y normal son las mismas. Para poder ejecutar mutect2

necesitamos el genoma de referencia, en este caso usamos GRCh38[52]. Cuando ejecutamos mutect2:

```
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../gatk/gatk Mutect2 -R
GRCh38.d1.vd1.fa -I shard_0001.bam -I shard_001.bam -O
somatic.vcf.gz
```

Nos muestra un error al final:

```
**********************************************************************
A USER ERROR has occurred: Fasta index file
file:///home/sonia/tfm/chr11Split/GRCh38.d1.vd1.fa.fai for reference
file:///home/sonia/tfm/chr11Split/GRCh38.d1.vd1.fa does not exist. Please see
http://gatkforums.broadinstitute.org/discussion/1601/how-can-i-prepare-a-fasta-file-to-use-
as-reference for help creating it.

**********************************************************************
```

Indicando que se necesita el fichero de índice para el genoma de referencia. Creamos el fichero de índice, extensión .bai, con la siguiente orden:

```
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../sam/samtools faidx
GRCh38.d1.vd1.fa
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ls -l
total 3322196
-rw-rw-r-- 1 sonia sonia 3010334720 nov 29 13:22 GRCh38.d1.vd1.fa
-rw-rw-r-- 1 sonia sonia        730 nov 29 14:24 GRCh38.d1.vd1.fa.fai
-rw-rw-r-- 1 sonia sonia  130777763 nov 29 12:28 shard_0001.bam
-rw-rw-r-- 1 sonia sonia  129850348 nov 29 12:29 shard_0002.bam
-rw-rw-r-- 1 sonia sonia  130951212 nov 29 12:29 shard_0003.bam
```

Repetimos la orden de mutect2:

```
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../gatk/gatk Mutect2 -R
GRCh38.d1.vd1.fa -I shard_0001.bam -I shard_0001.bam -O
somatic.vcf.gz
```

Y nos da otro error, esta vez relacionado con la necesidad de tener un archivo diccionario para el genoma de referencia.

[52] Descargamos el GRCh38 de esta página https://gdc.cancer.gov/about-data/gdc-data-processing/gdc-reference-files

```
**********************************************************************
A USER ERROR has occurred: Fasta dict file
file:///home/sonia/tfm/chr11Split/GRCh38.d1.vd1.dict for reference
file:///home/sonia/tfm/chr11Split/GRCh38.d1.vd1.fa does not exist. Please see
http://gatkforums.broadinstitute.org/discussion/1601/how-can-i-prepare-a-fasta-file-to-use-
as-reference for help creating it.
**********************************************************************
```

Procedemos a generar el diccionario con la siguiente orden[53]:

```
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../gatk/gatk
CreateSequenceDictionary -R GRCh38.d1.vd1.fa
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ls -l
total 3322200
-rw-rw-r-- 1 sonia sonia       2615 nov 29 14:36 GRCh38.d1.vd1.dict
-rw-rw-r-- 1 sonia sonia 3010334720 nov 29 13:22 GRCh38.d1.vd1.fa
-rw-rw-r-- 1 sonia sonia        730 nov 29 14:24
GRCh38.d1.vd1.fa.fai
-rw-rw-r-- 1 sonia sonia  130777763 nov 29 12:28 shard_0001.bam
-rw-rw-r-- 1 sonia sonia  129850348 nov 29 12:29 shard_0002.bam
-rw-rw-r-- 1 sonia sonia  130951212 nov 29 12:29 shard_0003.bam
```

Repetimos mutect2:

```
sonia@sonia-VirtualBox:~/tfm/chr11Split$ ../gatk/gatk Mutect2 -R
GRCh38.d1.vd1.fa -I shard_0001.bam -I shard_0001.bam -O
somatic.vcf.gz
```

Y nos devuelve otro error, esta vez relacionado con una incompatibilidad de *contigs* entre el fichero de referencia, GRCh38.d1.vd1.fa, y las lecturas, shard_0001.bam:

[53] Indicaciones sobre la preparación del fichero fasta con el genoma de referencia para uso en comandos del gatk: https://gatk.broadinstitute.org/hc/en-us/articles/360035531652-FASTA-Reference-genome-format

```
**********************************************************************
A USER ERROR has occurred: Input files reference and reads have incompatible contigs: No overlapping contigs found.

  reference contigs = [chr1, chr2, chr3, chr4, chr5, chr6, chr7, chr8, chr9, chr10, chr11, chr12, chr13, chr14, chr15, chr16, chr17, chr18, chr19, chr20, chr21, chr22, chrX]

  reads contigs = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, X, Y, MT, GL000207.1, GL000226.1, GL000229.1, GL000231.1, GL000210.1, GL000239.1, GL000235.1, GL000201.1, GL000247.1, GL000245.1, GL000197.1, GL000203.1, GL000246.1, GL000249.1, GL000196.1, GL000248.1, GL000244.1, GL000238.1, GL000202.1, GL000234.1, GL000232.1, GL000206.1, GL000240.1, GL000236.1, GL000241.1, GL000243.1, GL000242.1, GL000230.1, GL000237.1, GL000233.1, GL000204.1, GL000198.1, GL000208.1, GL000191.1, GL000227.1, GL000228.1, GL000214.1, GL000221.1, GL000209.1, GL000218.1, GL000220.1, GL000213.1, GL000211.1, GL000199.1, GL000217.1, GL000216.1, GL000215.1, GL000205.1, GL000219.1, GL000224.1, GL000223.1, GL000195.1, GL000212.1, GL000222.1, GL000200.1, GL000193.1, GL000194.1, GL000225.1, GL000192.1, NC_007605, hs37d5]
**********************************************************************
```

Buscamos información[54] relacionada con este error y se nos indica que una solución es partir de una muestra sin alinear y alinearla con nuestro genoma de referencia.

Podemos usar la herramienta picard con su opción RevertSam[55] para revertir el alineamiento y conseguir, a partir de nuestro fichero .bam alineado, el mismo fichero sin alinear. ls

```
sonia@sonia-VirtualBox:~/tfm$ java -jar picard.jar RevertSam I=chr11.bam O=uchr11.bam
```

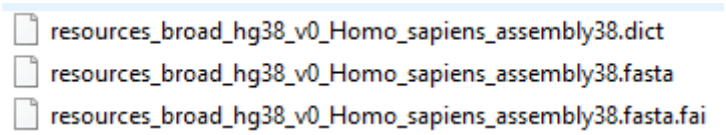
Luego podríamos alinearlo con el genoma de referencia que hemos estado usando con Mutect2 en los comandos anteriores.

[54] https://gatk.broadinstitute.org/hc/en-us/articles/360035891131
[55] https://gatk.broadinstitute.org/hc/en-us/articles/360037057532-RevertSam-Picard-

No obstante, decidimos buscar una alternativa, consistente en encontrar algún conjunto de datos ya preparado, teniendo en cuenta los problemas encontrados con la generación propia. Encontramos conjuntos de datos proporcionados por el instituto Broad para el análisis de secuencias genómicas. Partiendo de esta página:

https://console.cloud.google.com/storage/browser/genomics-public-data/resources/broad/hg38/v0;tab=objects?prefix=&forceOnObjectsSortingFiltering=false

Nos descargamos los ficheros:



Luego les cambiamos el nombre para acortarlos eliminando el texto anterior a Homo.

Luego nos descargamos los ficheros de ejemplo que se describen en el tutorial 11136 de cómo usar Mutect2[56].

Finalmente tenemos los ficheros necesarios para acometer el análisis:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls -l
total 1356384
-rwxrwx--- 1 sonia sonia    607714 ene 18  2018 4_NA19771.vcf.gz
-rwxrwx--- 1 sonia sonia     28618 ene 18  2018 4_NA19771.vcf.gz.tbi
-rwxrwx--- 1 sonia sonia    754824 ene 18  2018 5_HG02759.vcf.gz
-rwxrwx--- 1 sonia sonia     25282 ene 18  2018 5_HG02759.vcf.gz.tbi
-rwxrwx--- 1 sonia sonia    757396 ene 20  2018 chr17plus.interval_list
-rwxrwx--- 1 sonia sonia   3744912 ene 18  2018 HG00190.bai
-rwxrwx--- 1 sonia sonia 373599558 ene 18  2018 HG00190.bam
drwxrwxr-x 2 sonia sonia      4096 dic  5 20:40 hg38
-rwxrwx--- 1 sonia sonia   2126864 ene 18  2018 normal.bai
-rwxrwx--- 1 sonia sonia 437801420 ene 18  2018 normal.bam
drwxrwx--- 2 sonia sonia      4096 ene 20  2018 precomputed
drwxrwx--- 2 sonia sonia      4096 ene 18  2018 resources
-rwxrwx--- 1 sonia sonia    255301 ene 18  2018 somatic_m2.vcf.gz
-rwxrwx--- 1 sonia sonia     34573 ene 18  2018 somatic_m2.vcf.gz.tbi
-rwxrwx--- 1 sonia sonia     15312 ene 18  2018 tumor_artifact.pre_adapter_detail_metrics.txt
-rwxrwx--- 1 sonia sonia   2096768 ene 18  2018 tumor.bai
-rwxrwx--- 1 sonia sonia 567033677 ene 18  2018 tumor.bam
-rwxrwx--- 1 sonia sonia        78 ene 18  2018 tumor_calculatecontamination.table
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls hg38 -l
total 3174480
-rwxrwx--- 1 sonia sonia     581712 nov 29 17:42 Homo_sapiens_assembly38.dict
-rwxrwx--- 1 sonia sonia 3249912778 nov 29 17:48 Homo_sapiens_assembly38.fasta
-rwxrwx--- 1 sonia sonia     160928 nov 29 17:42 Homo_sapiens_assembly38.fasta.fai
```

[56] Tutorial: https://gatk.broadinstitute.org/hc/en-us/articles/360035889791?id=11136#ref4; Ficheros: https://drive.google.com/drive/folders/13RJCtKEhAXKCnLr-aQilFL1QId57bpHx . También podemos descargarnos los ejemplos de otros tutoriales de ftp://gsapubftp-anonymous@ftp.broadinstitute.org/tutorials/datasets.

8.5 Anexo V. Pasos para la obtención de variantes somáticas usando Mutect2

Usamos la siguiente orden para hacer el calling con Mutect2, de las muestras extraídas en los pasos detallados en el Anexo IV:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../gatk/gatk --java-
options "-Xmx2g" Mutect2 \
-R hg38/Homo_sapiens_assembly38.fasta \
-I tumor.bam \
-I normal.bam \
-tumor HCC1143_tumor \
-normal HCC1143_normal \
-pon resources/chr17_m2pon.vcf.gz \
--germline-resource resources/chr17_af-only-gnomad_grch38.vcf.gz \
--af-of-alleles-not-in-resource 0.0000025 \
--disable-read-filter MateOnSameContigOrNoMappedMateReadFilter \
-L chr17plus.interval_list \
-O 1_somatic_m2.vcf.gz \
-bamout 2_tumor_normal_m2.bam
```

En esta orden se indica la muestra tumoral y la normal usando la opción -I y a continuación el fichero .bam con las secuencias. Luego se incluyen las opciones -tumor y -normal y se indica el nombre del grupo de la muestra. Este nombre, se puede obtener visualizando el campo SM del fichero .bam:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../sam/samtools view -H  normal.bam | grep "@RG" | head -n 2
@RG     ID:C0ENM.3      LB:Pond-139372  PL:illumina     SM:HCC1143_normal       PU:C0ENMACXX120224.3.CATGCTTA   CN:B
I       DT:2012-02-24T05:00:00+0000
@RG     ID:C0ENM.4      LB:Pond-139372  PL:illumina     SM:HCC1143_normal       PU:C0ENMACXX120224.4.CATGCTTA   CN:B
I       DT:2012-02-24T05:00:00+0000
```

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../sam/samtools view -H  tumor.bam | grep "@RG" | head -n 2
@RG     ID:C0HV4.4      LB:Pond-147580  PL:illumina     SM:HCC1143_tumor        PU:C0HV4ACXX120402.4.GACCTAAC   CN:B
I       DT:2012-04-02T04:00:00+0000
@RG     ID:C0HV4.6      LB:Pond-147580  PL:illumina     SM:HCC1143_tumor        PU:C0HV4ACXX120402.6.GACCTAAC   CN:B
I       DT:2012-04-02T04:00:00+0000
```

Después tenemos la opción -pon. Esta opción se puede incluir o no, se recomienda incluirla. La opción -pon permite especificar un fichero de tipo "panel de normales" (*panel of normals*). Un fichero de este tipo almacena variaciones o mutaciones en una muestra normal respecto al genoma de referencia, de tal manera, que al usarlo en la detección de variantes entre una muestra normal y una tumoral, se puedan descartar como variantes aquellas que están contempladas en el fichero pon, pues si están en el fichero pon es que se trata de variantes germinales o derivadas del propio proceso técnico (artefactos técnicos) de obtención de las muestras y no

de mutaciones somáticas. El pon utilizado en el ejemplo lo proporciona el instituto Broad, los pon siempre se generan a partir de muestras normales de tejidos sanos que supuestamente no tienen alteraciones somáticas y su propósito es capturar esas variaciones germinales y artefactos técnicos que luego permitirán mejorar los resultados del análisis. Las muestras normales de las que se obtiene el pon deben haber sido obtenidas a nivel técnico con las mismas propiedades con las que se obtienen las muestras tumorales, es decir usando los mismos métodos para prepararlas, la misma tecnología de secuenciación, etc. Además de ser muestras procedentes de gente joven y sana, de forma que se minimice el riesgo de usar una muestra normal de alguien con un tumor no diagnosticado. Las muestras normales se suelen obtener de muestras de sangre. En el instituto Broad los pon para la detección de variantes cortas (*SNVs e Indels*) se obtienen usando la herramienta Mutect2[57].

Además del pon, podemos descartar otras variantes germinales proporcionando datos de bases de datos contrastadas usando la opción `--germline-resource`. En este caso hay que especificar que variants se descartan usando un % que indica que se descarten las variantes que ocurran al menos con ese % de frecuencia poblacional en los ancestros. En el comando empleado el % se ha marcado:

```
--af-of-alleles-not-in-resource 0.0000025
```

Se ha utilizado la opción -L para especificar que se lleve a cabo el análisis no en el conjunto de datos al completo, sino en un subconjunto el cual viene indicado en el fichero `chr17plus.interval_list`. De esta forma se reduce el tiempo de análisis que dadas las características del equipo donde se está ejecutando podría no ser viable si se hiciera el análisis con todo el conjunto al completo. Con un subconjunto ya se han tardado 22.2 minutos.

[57] Ver punto 2 en https://gatk.broadinstitute.org/hc/en-us/articles/360035889791?id=11136#ref4

Con la opción -O indicamos cómo queremos que se llame el fichero .vcf resultante del análisis.

Por último, la opción – `bamout,` nos permite obtener un fichero que luego podemos visualizar para hacer una revisión manual de las variantes detectadas.

Al terminar, la orden muestra que la ejecución ha sido exitosa después de 22.2 minutos:

```
0 read(s) filtered by: MappingQualityAvailableReadFilter
0 read(s) filtered by: MappingQualityNotZeroReadFilter
0 read(s) filtered by: MappedReadFilter
0 read(s) filtered by: NotSecondaryAlignmentReadFilter
1402003 read(s) filtered by: NotDuplicateReadFilter
0 read(s) filtered by: PassesVendorQualityCheckReadFilter
0 read(s) filtered by: NonChimericOriginalAlignmentReadFilter
0 read(s) filtered by: NonZeroReferenceLengthAlignmentReadFilter
0 read(s) filtered by: ReadLengthReadFilter
0 read(s) filtered by: GoodCigarReadFilter
0 read(s) filtered by: WellformedReadFilter
2495430 total reads filtered
21:12:07.627 INFO  ProgressMeter -  HLA-A*24:03:01:2323            22.2                286818          12928.9
21:12:07.633 INFO  ProgressMeter - Traversal complete. Processed 286818 total regions in 22.2 minutes.
21:12:08.006 INFO  VectorLoglessPairHMM - Time spent in setup for JNI call : 0.556942756
21:12:08.009 INFO  PairHMM - Total compute time in PairHMM computeLogLikelihoods() : 33.224269993
21:12:08.009 INFO  SmithWatermanAligner - Total compute time in java Smith-Waterman : 58.52 sec
21:12:14.252 INFO  Mutect2 - Shutting down engine
[5 de diciembre de 2020 21:12:14 CET] org.broadinstitute.hellbender.tools.walkers.mutect.Mutect2 done. Elapsed time:
 22.57 minutes.
Runtime.totalMemory()=213045248
Tool returned:
SUCCESS
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$
```

Como resultado del análisis obtenemos estos ficheros:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls -l
total 1359116
-rw-rw-r-- 1 sonia sonia     48001 dic  5 21:12 1_somatic_m2.vcf.gz
-rw-rw-r-- 1 sonia sonia        35 dic  5 21:12 1_somatic_m2.vcf.gz.stats
-rw-rw-r-- 1 sonia sonia      1790 dic  5 21:12 1_somatic_m2.vcf.gz.tbi
-rw-rw-r-- 1 sonia sonia     71008 dic  5 21:12 2_tumor_normal_m2.bai
-rw-rw-r-- 1 sonia sonia   2664654 dic  5 21:12 2_tumor_normal_m2.bam
```

En el fichero 1_somatic_m2.vcf.gz está el fichero .vcf con las variantes detectadas. Debemos aplicar a este fichero un conjunto de filtros para detectar los verdaderos positivos y los falsos positivos.

Se utiliza FilterMutectCalls para llevar esto a término:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../gatk/gatk FilterMutectCalls \
-V 1_somatic_m2.vcf.gz \
-R hg38/Homo_sapiens_assembly38.fasta \
-O somatic_oncefiltered.vcf.gz
```

Concluido el comando nos muestra:

```
20:02:35.516 INFO  ProgressMeter -            unmapped              0.1                  656          5678.0
20:02:35.520 INFO  ProgressMeter - Traversal complete. Processed 656 total variants in 0.1 minutes.
20:02:35.714 INFO  FilterMutectCalls - Shutting down engine
[6 de diciembre de 2020 20:02:35 CET] org.broadinstitute.hellbender.tools.walkers.mutect.filtering.FilterMutectCalls
 done. Elapsed time: 0.22 minutes.
Runtime.totalMemory()=62980096
```

Y se ha obtenido varios ficheros cuyo prefijo es el especificado en el parámetro -O:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls -l somatic_oncefiltered*.*
-rw-rw-r-- 1 sonia sonia 50864 dic  6 20:02 somatic_oncefiltered.vcf.gz
-rw-rw-r-- 1 sonia sonia  1817 dic  6 20:02 somatic_oncefiltered.vcf.gz.filteringStats.tsv
-rw-rw-r-- 1 sonia sonia  1791 dic  6 20:02 somatic_oncefiltered.vcf.gz.tbi
```

Podemos ver los filtros que aplica la orden FilterMutectCalls usando este comando:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ zcat somatic_oncefiltered.vcf.gz | grep "##FILTER"
##FILTER=<ID=FAIL,Description="Fail the site if all alleles fail but for different reasons.">
##FILTER=<ID=PASS,Description="Site contains at least one allele that passes filters">
##FILTER=<ID=base_qual,Description="alt median base quality">
##FILTER=<ID=clustered_events,Description="Clustered events observed in the tumor">
##FILTER=<ID=contamination,Description="contamination">
##FILTER=<ID=duplicate,Description="evidence for alt allele is overrepresented by apparent duplicates">
##FILTER=<ID=fragment,Description="abs(ref - alt) median fragment length">
##FILTER=<ID=germline,Description="Evidence indicates this site is germline, not somatic">
##FILTER=<ID=haplotype,Description="Variant near filtered variant on same haplotype.">
##FILTER=<ID=low_allele_frac,Description="Allele fraction is below specified threshold">
##FILTER=<ID=map_qual,Description="ref - alt median mapping quality">
##FILTER=<ID=multiallelic,Description="Site filtered because too many alt alleles pass tumor LOD">
##FILTER=<ID=n_ratio,Description="Ratio of N to alt exceeds specified ratio">
##FILTER=<ID=normal_artifact,Description="artifact_in_normal">
##FILTER=<ID=orientation,Description="orientation bias detected by the orientation bias mixture model">
##FILTER=<ID=panel_of_normals,Description="Blacklisted site in panel of normals">
##FILTER=<ID=position,Description="median distance of alt variants from end of reads">
##FILTER=<ID=possible_numt,Description="Allele depth is below expected coverage of NuMT in autosome">
##FILTER=<ID=slippage,Description="Site filtered due to contraction of short tandem repeat region">
##FILTER=<ID=strand_bias,Description="Evidence for alt allele comes from one read direction only">
##FILTER=<ID=strict_strand,Description="Evidence for alt allele is not represented in both directions">
##FILTER=<ID=weak_evidence,Description="Mutation does not meet likelihood threshold">
```

Las variantes que son verdaderos positivos son etiquetadas con la etiqueta PASS en el campo FILTER del fichero resultante de la ejecución de FilterMutectCalls (somatic_oncefiltered.vcf.gz). Las que probablemente son falsos positivos son las que no han pasado alguno o todos los filtros aplicados y contendrán en el campo FILTER el nombre de los filtros que no han pasado.

En este ejemplo:

```
chr17   2297261 .       G       T       . base_qual;normal_artifact;strand_bias;weak_evidence
chr17   2329275 .       A       C       . weak_evidence
chr17   2394409 .       G       T       . PASS
```

Vemos que las dos primeras líneas no han pasado uno o más filtros y que la tercera los ha pasado todos.

Para obtener cuántas líneas de variantes detectadas contiene el fichero vcf, hay que conocer cómo es el formato de estos ficheros. En la siguiente imagen se muestra la estructura de un fichero vcf:

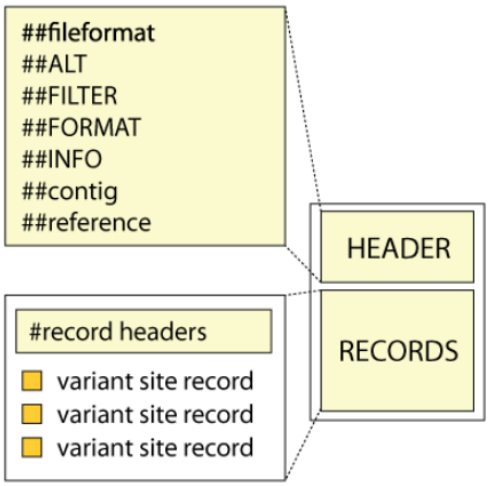


Figura 12. Estructura de un fichero vcf
(Fuente: https://gatk.broadinstitute.org/hc/en-us/articles/360035531692-VCF-Variant-Call-Format)

Después de las líneas de la cabecera (*HEADER*) y de la línea con los nombres de los campos *(#record headers*), aparece una línea para cada una de las variantes detectadas.

Filtraremos el fichero para quedarnos solo con las líneas de las variantes, contaremos todas, luego contaremos las que contienen PASS en el campo FILTER. La relación entre las que contienen PASS y el total nos da la tasa de verdaderos positivos. La relación entre las que no contienen PASS y el total nos da la tasa de falsos positivos.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ zcat somatic_oncefiltered.vcf.gz
| grep -v "#" | wc -l
164
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ zcat somatic_oncefiltered.vcf.gz
| grep -v "#" | grep "PASS" |wc -l
28
```

Tasa de verdaderos positivos = 28 / 164 = 0,17

Tasa de falsos positivos = 1- 0,17 = 0,83

8.6 Anexo VI. Instalación VarScan2

Para descargar la herramienta accedemos a la dirección http://varscan.sourceforge.net. Al acceder nos informa que debemos dirigirnos a un repositorio de GitHub (http://dkoboldt.github.io/varscan/) para la descarga de la última versión.

Descargamos el fichero .jar y luego hacemos una prueba de ejecución desde la línea de comandos:

Cambiamos el nombre del fichero .jar para manejarlo mejor

```
sonia@sonia-VirtualBox:~/tfm$ mv VarScan.v2.3.9.jar VarScan.jar
```

Luego probamos su funcionamiento (se muestran las primeras líneas de la salida del comando):

```
sonia@sonia-VirtualBox:~/tfm$ java -jar VarScan.jar
VarScan v2.3
USAGE: java -jar VarScan.jar [COMMAND] [OPTIONS]
….
```

8.7 Anexo VII. Pasos para la obtención de variantes somáticas usando VarScan2

Tomaremos como ficheros de partida, los .bam que usamos con Mutect2:

tumor.bam

normal.bam

Primero ordenamos los ficheros por posición, usando samtools y su opción sort. Creamos también los ficheros índices de estos nuevos ficheros ordenados, con la opción index de samtools.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../sam/samtools sort tumor.bam -
o tumor_sorted.bam
[bam_sort_core] merging from 2 files and 1 in-memory blocks...
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../sam/samtools sort normal.bam
-o normal_sorted.bam
[bam_sort_core] merging from 2 files and 1 in-memory blocks...
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls -l *_sorted*.*
-rw-rw-r-- 1 sonia sonia 374021620 dic  8 20:42 normal_sorted.bam
-rw-rw-r-- 1 sonia sonia 486374961 dic  8 20:37 tumor_sorted.bam
```

Convertimos estos ficheros a formato pileup:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../sam/samtools mpileup -f
hg38/Homo_sapiens_assembly38.fasta normal_sorted.bam > normal.pileup
[mpileup] 1 samples in 1 input files
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls -l normal.pileup
-rw-rw-r-- 1 sonia sonia 775501217 dic  8 22:33 normal.pileup

sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../sam/samtools mpileup -f
hg38/Homo_sapiens_assembly38.fasta tumor_sorted.bam > tumor.pileup
[mpileup] 1 samples in 1 input files
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls -l tumor.pileup
-rw-rw-r-- 1 sonia sonia 961029926 dic  8 23:08 tumor.pileup
```

Efectuamos el análisis con VarScan2:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ java -jar ../VarScan.jar somatic
normal.pileup tumor.pileup somatic_varsan
```

Al lanzar el análisis nos informa de los valores de los parámetros por defecto:

```
Normal Pileup: normal.pileup
Tumor Pileup: tumor.pileup
Min coverage:   8x for Normal, 6x for Tumor
Min reads2: 2
Min strands2:   1
Min var freq:   0.2
Min freq for hom:       0.75
Normal purity:  1.0
Tumor purity:   1.0
Min avg qual:   15
P-value thresh: 0.99
Somatic p-value:        0.05
```

Una vez concluido el análisis nos muestra un resumen del mismo:

```
11570336 positions in tumor
5696 positions shared in normal
1471 had sufficient coverage for comparison
1423 were called Reference
0 were mixed SNP-indel calls and filtered
30 were called Germline
16 were called LOH
2 were called Somatic
0 were called Unknown
0 were called Variant
```

Nos indica que las posiciones coincidentes entre la muestra normal y el tumor son 5696 de las que 1471 tienen la suficiente cobertura para ser comparadas. De estas 1471, ya se encuentran en los datos de referencia 1423 por lo que quedan 48 que se clasifican en 30 de tipo germinal, 16 LOH y solo 2 somáticas.

Concluido el análisis se generan dos ficheros, uno con la extensión .indel y otro con la extensión .snp; en el primero aparecerá una línea por cada variante de tipo inserción/eliminación encontrada; en el segundo una línea por cada variante de tipo SNP encontrada.

El fichero .snp contiene 48 líneas, el campo número 13 denominado somatic_status, indica el tipo de variante (Germline, LOH, Somatic, Unknown, Variant).

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat somatic_varsan.snp | head | cut -f 1,2,3,4,13
chrom   position        ref     var     somatic_status
chr6    29890313        C       A       Germline
chr6    29890630        G       C       Germline
chr6    29890699        T       C       Germline
chr6    29890706        C       T       Germline
chr6    29942488        G       A       Germline
chr6    29942497        G       C       Germline
chr6    29942499        T       G       Germline
chr6    29942512        G       C       Germline
chr6    29942517        C       A       Germline
```

8.8 Anexo VIII. Instalación SomaticSniper

Accedemos a la página http://gmt.genome.wustl.edu/packages/somatic-sniper/

Y en la pestaña de **instalation** se nos indica que primero instalemos una serie de paquetes de los que depende SomaticSniper

```
sonia@sonia-VirtualBox:~$ sudo apt-get install build-essential git-core cmake zlib1g-dev libncurses-dev
[sudo] password for sonia:
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Nota, seleccionando «libncurses5-dev» en lugar de «libncurses-dev»
build-essential ya está en su versión más reciente (12.1ubuntu2).
libncurses5-dev ya está en su versión más reciente (6.0+20160213-1ubuntu1).
zlib1g-dev ya está en su versión más reciente (1:1.2.8.dfsg-2ubuntu4.3).
Tal vez quiera ejecutar «apt-get -f install» para corregirlo:
Los siguientes paquetes tienen dependencias incumplidas:
 cmake : Depende: cmake-data (= 3.5.1-1ubuntu3) pero no va a instalarse
         Depende: libjsoncpp1 pero no va a instalarse
 linux-generic-hwe-16.04 : Depende: linux-headers-generic-hwe-16.04 (= 4.15.0.128.127) pero 4.15.0.126.125 va a ser instalado
E: Dependencias incumplidas. Intente «apt-get -f install» sin paquetes (o especifique una solución).
```

Aunque parece que algunos paquetes tienen otras dependencias y no se instalan, vamos a continuar con los siguientes pasos por si realmente no se necesitasen.

El siguiente paso es clonar el repositorio de **github** de SomaticSniper. Primero nos cambiamos a la carpeta del **tfm** y luego lo clonamos ahí:

```
sonia@sonia-VirtualBox:~/tfm$ git clone git://github.com/genome/somatic-sniper.git
Clonar en «somatic-sniper»...
remote: Enumerating objects: 1228, done.
remote: Total 1228 (delta 0), reused 0 (delta 0), pack-reused 1228
Receiving objects: 100% (1228/1228), 3.42 MiB | 0 bytes/s, done.
Resolving deltas: 100% (565/565), done.
Comprobando la conectividad… hecho.
```

Esto nos crea una carpeta denominada **somatic-sniper.** Cambiamos a esta carpeta y ejecutamos:

```
cmake ../
```

Nos indica que debemos instalar el programa **cmake**:

```
sonia@sonia-VirtualBox:~/tfm/somatic-sniper/build$ sudo apt install cmake
Leyendo lista de paquetes... Hecho
Creando árbol de dependencias
Leyendo la información de estado... Hecho
Tal vez quiera ejecutar «apt-get -f install» para corregirlo:
Los siguientes paquetes tienen dependencias incumplidas:
 cmake : Depende: cmake-data (= 3.5.1-1ubuntu3) pero no va a instalarse
         Depende: libjsoncpp1 pero no va a instalarse
 linux-generic-hwe-16.04 : Depende: linux-headers-generic-hwe-16.04 (= 4.15.0.128.127) pero 4.15.0.126.125 va a ser instalado
E: Dependencias incumplidas. Intente «apt-get -f install» sin paquetes (o especifique una solución).
```

Al intentar instalarlo nos da un problema de dependencias y nos sugiere ejecutar **apt-get install** con la opción **-f**, opción que permite resolver problemas de dependencias y paquetes rotos, procedemos:

```
sonia@sonia-VirtualBox:~/tfm/somatic-sniper/build$ sudo apt-get -f install
```

Una vez concluida la ejecución del comando anterior, repetimos la instalación de **cmake**:

```
sonia@sonia-VirtualBox:~/tfm/somatic-sniper/build$ sudo apt install cmake
```

En esta ocasión todo funciona correctamente. Volvemos a repetir la ejecución de la siguiente orden:

```
sonia@sonia-VirtualBox:~/tfm/somatic-sniper/build$ cmake ../
```

Después:

```
sonia@sonia-VirtualBox:~/tfm/somatic-sniper/build$ make deps
```

Luego:

```
sonia@sonia-VirtualBox:~/tfm/somatic-sniper/build$ make -j
```

Y por último:

```
sonia@sonia-VirtualBox:~/tfm/somatic-sniper/build$ make test
Running tests...
Test project /home/sonia/tfm/somatic-sniper/build
    Start 1: SniperUnitTests
1/2 Test #1: SniperUnitTests .................   Passed    0.02 sec
    Start 2: Sniper
2/2 Test #2: Sniper ..........................   Passed    1.45 sec

100% tests passed, 0 tests failed out of 2

Label Time Summary:
integration    =   1.45 sec (1 test)
unit           =   0.02 sec (1 test)

Total Test time (real) =   1.48 sec
```

Podemos encontrar el fichero **bam-somaticsniper** en la ruta **somatic-sniper/build/bin**.

```
sonia@sonia-VirtualBox:~/tfm/somatic-sniper/build$ ls -l bin
total 372
-rwxrwxr-x 1 sonia sonia 379920 dic 27 21:21 bam-somaticsniper
```

8.9 Anexo IX. Pasos para la obtención de variantes somáticas usando SomaticSniper

Probamos a ejecutar SomaticSniper primero en el Modo 1, modo por defecto, considerando las muestras tumor-normal independientes sin usar ninguna opción adicional de las disponibles.

Modo 1:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../somatic-sniper/build/bin/bam-somaticsniper -f hg38/Homo_sapiens_assembly38.fasta
 tumor.bam normal.bam snv_somaticsniper
Preparing to snipe some somatics
Using prior probabilities
Normal bam is normal.bam
Tumor bam is tumor.bam
```

Después de unos 20 minutos, el análisis concluye y podemos ver el fichero de salida snv_somaticsniper en la carpeta:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls -l snv_somaticsniper
-rw-rw-r-- 1 sonia sonia 250510 dic 27 22:19 snv_somaticsniper
```

El número de filas del fichero es:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniper | wc -l
3017
```

Al no haber utilizado la opción -F con el valor vcf, la salida se efectúa en el modo clásico. En cada línea del fichero, en este modo, aparecen 26 valores separados por tabuladores.

.

Tabla 3. Valores de cada fila del fichero resultante de un análisis con SomaticSniper. Formato clásico del fichero de salida[58]

1.	Chromosome
2.	Position
3.	Reference base
4.	IUB genotype of tumor
5.	IUB genotype of normal
6.	Somatic Score
7.	Tumor Consensus quality
8.	Tumor variant allele quality
9.	Tumor mean mapping quality
10.	Normal Consensus quality
11.	Normal variant allele quality
12.	Normal mean mapping quality
13.	Depth in tumor (# of reads crossing the position)
14.	Depth in normal (# of reads crossing the position)
15.	Mean base quality of reads supporting reference in tumor
16.	Mean mapping quality of reads supporting reference in tumor

[58] Esta tabla y las siguientes están extraídas de la página de documentación de SomaticSniper: http://gmt.genome.wustl.edu/packages/somatic-sniper/documentation.html

17. Depth of reads supporting reference in tumor
18. Mean base quality of reads supporting variant(s) in tumor
19. Mean mapping quality of reads supporting variant(s) in tumor
20. Depth of reads supporting variant(s) in tumor
21. Mean base quality of reads supporting reference in normal
22. Mean mapping quality of reads supporting reference in normal
23. Depth of reads supporting reference in normal
24. Mean base quality of reads supporting variant(s) in normal
25. Mean mapping quality of reads supporting variant(s) in normal
26. Depth of reads supporting variant(s) in normal

Nos interesa más la salida en formato vcf, cuyos valores para cada línea contienen los siguientes campos:

Tabla 4. Valores de cada fila del fichero resultante de un análisis con SomaticSniper. Formato vcf para el fichero de salida

1. Chromosome
2. Position
3. ID (unused)
4. Reference base
5. Alternate bases (comma separated)
6. Quality (unused)
7. Filters (unused)
8. INFO (unused)
9. FORMAT specification for each sample
10. NORMAL sample data
11. TUMOR sample data

Tabla 5. Campo FORMAT, fichero vcf salida de SomaticSniper

ID	***Number***	***Type***	***Description***
GT	1	String	Genotype
IGT	1	String	Genotype when called independently (only filled if called in joint prior mode)
DP	1	Integer	Total read depth
DP4	4	Integer	Number of high-quality ref-forward bases, ref-reverse, alt-forward, and alt-reverse bases
BCOUNT	4	Integer	Occurrence count for each base at this site (A,C,G,T)
GQ	1	Integer	Genotype quality
JGQ	1	Integer	Joint genotype quality (only filled if called in joint prior mode)
VAQ	1	Integer	Variant quality
BQ	.	Integer	Average base quality of each base in the call, reported in alphabetical order (A,C,G,T)
MQ	1	Integer	Average mapping quality across all reads.
AMQ	.	Integer	Average mapping quality of each base in the call, reported in alphabetical order (A,C,G,T)
SS	1	Integer	Variant status relative to non-adjacent normal: 0=wildtype, 1=germline, 2=somatic, 3=LOH, 4=unknown
SSC	1	Integer	Somatic Score

Nos interesará fijarnos en el valor del campo FORMAT del tumor, y dentro de este en el valor del identificador SS, que puede tomar los siguientes valores:

0=wildtype, 1=germline, 2=somatic, 3=LOH, 4=unknown

Por otro lado, se recomienda la ejecución del comando usando algunas de las opciones que tiene. En concreto la opción -Q con el valor ajustado a 40 y las opciones -G y -L. La opción -Q permite filtrar la salida, de forma que no se muestran aquellas variaciones (snv) con una calidad menor que la indicada. La opción -G y -L permiten reducir los posibles falsos positivos sin afectar mucho a la sensibilidad.

Procedemos a ejecutar, en el Modo 1, la orden con la opción -F vcf y el resto de las opciones anteriormente mencionadas:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../somatic-sniper/build/bin/bam-somaticsniper -Q 40 -G -L -F vcf -f
 hg38/Homo_sapiens_assembly38.fasta tumor.bam normal.bam snv_somaticsniper.vcf
```

Como siempre, en el fichero vcf resultante tendremos las líneas de cabecera y luego las correspondientes a las variantes detectadas:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniper.vcf | head
##fileformat=VCFv4.1
##fileDate=20201227
##phasing=none
##reference=file://hg38/Homo_sapiens_assembly38.fasta
##FORMAT=<ID=GT,Number=1,Type=String,Description="Genotype">
##FORMAT=<ID=IGT,Number=1,Type=String,Description="Genotype when called independently (only filled if called in
joint prior mode)">
##FORMAT=<ID=DP,Number=1,Type=Integer,Description="Total read depth">
##FORMAT=<ID=DP4,Number=4,Type=Integer,Description="# high-quality ref-forward bases, ref-reverse, alt-forward a
nd alt-reverse bases">
##FORMAT=<ID=BCOUNT,Number=4,Type=Integer,Description="Occurrence count for each base at this site (A,C,G,T)">
##FORMAT=<ID=GQ,Number=1,Type=Integer,Description="Genotype quality">
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniper.vcf | grep -v "#" | wc -l
136
```

En total son 136 líneas de variantes.

Si analizamos el contenido de una línea:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniper.vcf | grep -v "#" | head -n 1
chr6    29944538        .       A       C       .       .       .       GT:IGT:DP:DP4:BCOUNT:GQ:JGQ:VAQ:BQ:MQ:AMQ:SS:SSC
        0/0:0/0:33:20,7,0,6:27,4,1,0:53:.:0:31:57:59:0:.    1/1:1/1:10:0,1,0,9:1,9,0,0:43:.:54:33:53:58:2:69
```

El valor de los campos descritos en la Tabla 4 es:

Chromosome	chr6
Position	29944538
ID (unused)	.
Reference base	A
Alternate bases (comma separated)	C
Quality (unused)	.
Filters (unused)	.
INFO (unused)	.
FORMAT specification for each sample	GT:IGT:DP:DP4:BCOUNT:GQ:JGQ:VAQ:BQ:MQ:AMQ:SS:SSC
NORMAL sample data	0/0:0/0:33:20,7,0,6:27,4,1,0:53:.:0:31:57:59:0:.
TUMOR sample data	1/1:1/1:10:0,1,0,9:1,9,0,0:43:.:54:33:53:58:2:69

En concreto el valor que nos interesa es el remarcado en verde en la tabla anterior.

Procedemos a filtrar las 136 líneas para quedarnos con las variantes de tipo 2, que son las somáticas: SS=2.

Usamos el comando cut para quedarnos con el último campo de las líneas que contiene el valor de FORMAT del tumor. Y luego con grep nos quedamos con las líneas que tengan al final ($) el patrón *:2:cualquier combinación de los números del 0 al 9* ([0-9]*).

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniper.vcf | grep -v "#" | cut -f 11 | grep ":2:[0-9]*$" | wc
 -l
136
```

Como vemos, el número de líneas sigue siendo 136, por tanto, son 136 las variantes somáticas detectadas con las opciones utilizadas al ejecutar SomaticSniper.

Podemos repetir lo mismo, con las mismas opciones, pero en el Modo 2:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ../somatic-sniper/build/bin/bam-somaticsniper -Q 40 -G -L -J -F vcf -f
hg38/Homo_sapiens_assembly38.fasta tumor.bam normal.bam snv_somaticsniperM2.vcf
Using priors accounting for somatic mutation rate. Prior probability of a somatic mutation is 0.010000
Preparing to snipe some somatics
Using prior probabilities
Normal bam is normal.bam
Tumor bam is tumor.bam
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep -v "#" | wc -l
59
```

Ahora el número de variantes detectadas se reduce a 59.

De las cuales son todas somáticas.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep -v "#" | cut -f 11 | grep ":2:[0-9]*
$" | wc -l
59
```

Si queremos saber en qué cromosomas se han detectado las variantes podemos mostrar este último fichero con cat y more para verlo paso a paso. No son demasiadas líneas. El vistazo rápido nos indica que los cromosomas son el chr6, el 17 y el 11. Para comprobarlo contamos:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep "chr6" | wc -l
6
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep "chr17" | wc -l
50
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ cat snv_somaticsniperM2.vcf | grep "chr11" | wc -l
3
```

Sumando 6+50+3=59 total líneas de variantes detectadas.

8.10 Anexo X. Instalación Strelka2

Primero descargamos el fichero comprimido que contiene la distribución de Strelka2:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ wget https://github.com/Illumina/strelka/releases/download/v
2.9.2/strelka-2.9.2.centos6_x86_64.tar.bz2
```

A continuación, descomprimimos el archivo:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ tar xvjf strelka-2.9.2.centos6_x86_64.tar.bz2
```

Luego efectuamos la ejecución de la prueba recomendada relacionada con la detección de variantes somáticas:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ bash strelka-2.9.2.centos6_x86_64/bin/runStrelkaSomaticWorkf
lowDemo.bash
```

Cuyo resultado es exitoso:

```
**** Completed demo workflow execution.


**** Starting comparison to expected results.
**** Expected results dir: strelka-2.9.2.centos6_x86_64/bin/../share/demo/strelka/expectedResults
**** Demo results dir: ./strelkaSomaticDemoAnalysis/results/variants


**** No differences between expected and computed results.


**** Demo/verification successfully completed
```

También se proporciona otra prueba para la detección de variantes germinales, pero no la ejecutamos.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ bash strelka-2.9.2.centos6_x86_64/bin/runStrelkaGermlineWork
flowDemo.bash
```

Una vez instalado se crea una carpeta que contiene la distribución de Strelka2 y la carpeta con las demos. (strelka-2.9.2.centos6_x86_64, strelka-2.9.2.centos6_x86_64).

Dentro de la carpeta strelka-2.9.2.centos6_x86_64, está la carpeta bin, dentro de la cual está un fichero Python que permitirá la configuración necesaria para la ejecución del análisis somático (`configureStrelkaSomaticWorkflow.py`):

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls -l strelka-2.9.2.centos6_x86_64/bin
total 36
-rwxr-xr-x 1 sonia sonia 8537 mar  2  2018 configureStrelkaGermlineWorkflow.py
-rw-r--r-- 1 sonia sonia  606 mar  2  2018 configureStrelkaGermlineWorkflow.py.ini
-rwxr-xr-x 1 sonia sonia 6085 mar  2  2018 configureStrelkaSomaticWorkflow.py
-rw-r--r-- 1 sonia sonia 2580 mar  2  2018 configureStrelkaSomaticWorkflow.py.ini
-rwxr-xr-x 1 sonia sonia 3465 mar  2  2018 runStrelkaGermlineWorkflowDemo.bash
-rwxr-xr-x 1 sonia sonia 3384 mar  2  2018 runStrelkaSomaticWorkflowDemo.bash
```

El resto de los ficheros son para el análisis de variantes germinales, los script para la ejecución de las demos, y los .ini de configuración.

8.11 Anexo XI. Pasos para la obtención de variantes somáticas usando Strelka2

Para la obtención de variantes somáticas con Strelka2 hay que efectuar dos pasos. El primero consiste en la configuración de los datos de entrada y las opciones y el segundo en la ejecución de flujo de trabajo que permite la obtención de las variantes somáticas.

Para la configuración usamos la siguiente orden:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ strelka-2.9.2.centos6_x86_64/bin/configureStrelkaSomaticWork
flow.py --normalBam normal.bam --tumorBam tumor.bam --referenceFasta hg38/Homo_sapiens_assembly38.fasta -
-runDir strelka2_somatic

Successfully created workflow run script.
To execute the workflow, run the following script and set appropriate options:

/home/sonia/tfm/tutorial_11136/strelka2_somatic/runWorkflow.py
```

Tras la ejecución de la anterior orden de configuración aparece el directorio strelka2_somatic, que contiene el fichero Python runWorkflow.py que usaremos para la ejecución final del análisis.

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136$ ls strelka2_somatic
results  runWorkflow.py  runWorkflow.py.config.pickle  workspace
```

Ejecutamos el siguiente paso:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ strelka2_somatic/runWorkflow.py -m local
```

Una vez concluido el análisis se almacena el resultado en la carpeta results/variants

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ ls
somatic.indels.vcf.gz  somatic.indels.vcf.gz.tbi  somatic.snvs.vcf.gz  somatic.snvs.vcf.gz.tbi
```

En esta dirección podemos ver información sobre el modo de interpretar los resultados del análisis:

https://github.com/Illumina/strelka/blob/v2.9.x/docs/userGuide/README.md#outputs

Como resultado del análisis somático obtenemos un fichero que contiene todas las variantes de tipo SNVs descubiertas en la muestra tumoral, en nuestro caso el fichero **somatic.snvs.vcf.gz;** y otro fichero que contiene las variantes de tipo indel: **somatic.indels.vcf.gz**.

Contemos las líneas de cada uno de estos ficheros:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ zcat somatic.snvs.vcf.gz | grep -v "#" | wc -l
2806
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ zcat somatic.indels.vcf.gz | grep -v "#" | wc -l
80
```

Veamos cuántas pasan los filtros:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ zcat somatic.indels.vcf.gz | grep "PASS" | wc -l
1
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ zcat somatic.snvs.vcf.gz | grep "PASS"| wc -l
67
```

Los cromosomas en los que son detectadas las variantes son el chr17, una variante de tipo indel:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ zcat somatic.indels.vcf.gz | grep "PASS"
chr17   81647906        .       T       TTAAC   .       PASS    SOMATIC;QSI=58;TQSI=1;NT=ref;QSI_NT=51;TQSI_NT=1;SGT=ref->het;M
Q=60.00;MQ0=0;RU=TAAC;RC=1;IC=2;IHP=3;SomaticEVS=17.28  DP:DP2:TAR:TIR:TOR:DP50:FDP50:SUBDP50:BCN50     4:4:4,4:0,0:0,0:4.59:0.
00:0.00:0.00    28:28:0,0:27,28:3,3:26.97:0.02:0.00:0.00
```

Y también el chr17 en las variantes de tipo snv:

```
sonia@sonia-VirtualBox:~/tfm/tutorial_11136/strelka2_somatic/results/variants$ zcat somatic.snvs.vcf.gz | grep "PASS"
| grep "chr17" | wc -l
67
```