Trabajo de Fin de Grado

Grado en Ingeniería Informática

Herramientas de apoyo para el análisis y procesado de datos del secuenciador de ADN MinION

Supporting tools for data analysis and processing using the MinION DNA sequencer.

Orlandy Ariel Sánchez Acosta.

La Laguna, 04 de junio de 2018

D. **José L. Roda García**, con N.I.F. 43.356.123-L profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como tutor

D. **Marcos Colebrook Santamaría**, con N.I.F. 43.787.808-V profesor Titular de Universidad adscrito al Departamento de Ingeniería Informática y de Sistemas de la Universidad de La Laguna, como cotutor

C E R T I F I C A (N)

Que la presente memoria titulada:

"Herramientas de apoyo para el análisis y procesado de datos del secuenciador de ADN MinION"

ha sido realizada bajo su dirección por D. **Orlandy Ariel Sánchez Acosta**, con N.I.F. 42.289.970-P.

Y para que así conste, en cumplimiento de la legislación vigente y a los efectos oportunos firman la presente en La Laguna a 04 de junio de 2018.

Agradecimientos

A mis tutores Dr. José Luis Roda y Dr. Marcos Colebrook, por la ayuda y paciencia en el desarrollo de este proyecto, además de la motivación para seguir mejorando.

A Dr. Carlos Pérez por su ayuda.

A Dr. Carlos Flores, por permitirme ser partícipe de una de sus ideas y así conocer una rama que, hasta el momento, no había estudiado.

A mis amigos y compañeros, por aguantarme. Pedro y Joaquín por esas tardes de estudios en el carrel. Mención especial a María por su apoyo.

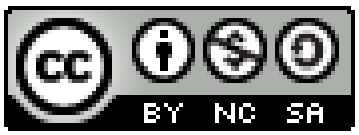
A Manu, por ser como un hermano y estar en las buenas y en las malas.

A mis hermanas Zorey y Orda, que siempre me han apoyado y que me ayudaron a no pasarlo tan mal después de unas cuantas operaciones.

A mi padre, que siempre ha velado que nunca nos faltara nada y ser mi compañero de juegos.

A mi madre, por ser un ejemplo en todos los sentidos y enseñarme que si luchas puedes conseguir lo que te propongas.

Licencia



Resumen

Este proyecto es una colaboración entre biólogos e ingenieros informáticos de la ULL. Este trabajo tiene como objetivo desarrollar un cuaderno utilizando Jupyter para trabajar con un MinION, un secuenciador de ADN. El Dr. Carlos Flores y su equipo usan el secuenciador MinION y, por tanto, están al día de lo que ocurre con el dispositivo.

Entre líneas, en un foro, alguien comentó que estaban detectando anomalías con secuencias obtenidas de MinION por lo que el Dr. Carlos Flores y su equipo se dispusieron a comprobar si esto era cierto. Tras secuenciar una bacteria y al intentar ensamblar estas secuencias se dieron cuenta de que efectivamente existían anomalías, en las que se formaban bucles y no permitían que se completara el ensamblado del genoma de la bacteria.

El cuaderno Jupyter desarrollado tiene como objetivo realizar una serie de filtros a los datos proporcionados por el Dr. Carlos Flores de los cuales se podrán ver y extraer los datos filtrados gráficamente y en formatos específicos para su posterior ensamblado.

Palabras clave: ADN, BLAST, Bioinformática, Secuenciación del genoma, MinION, Jupyter Notebook.

Abstract

This project is a collaboration between biologists and computer engineers from ULL. This work aims to develop a notebook using Jupyter to work with a MinION, a DNA sequencer. Dr. Carlos Flores and his team use the MinION sequencer and are therefore up to date on what is happening with the device.

Between the lines, in a forum, someone commented that they were detecting anomalies with sequences obtained from the MinION, so Dr. Carlos Flores and his team set out to check if this was true. After sequencing a bacterium and attempting to assemble these sequences, they realized that there were indeed anomalies in which loops formed and did not allow the bacterium to complete the assemblage.

The notebook aims to make a series of filters to the data provided by Dr. Carlos Flores from which it will be possible to see and extract the data filtered graphically and in specific format for later assembly.

Keywords: DNA, BLAST, Bioinformatics, Genome sequencing, MinION, Jupyter Notebook.

Índice General

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

1.1 ¿Qué es la Bioinformática?

Según Philip Bourne [51], "*la Bioinformática se ha convertido en el intérprete del lenguaje genómico del ADN y está intentando descifrar lenguajes más complejos en los que las proteínas son los sustantivos, las interpretaciones son la sintaxis, las rutas metabólicas son las oraciones y los sistemas vivos son el volumen complejo*".

Las tecnologías están generando una gran cantidad de información sin precedentes en la historia de la biología. Por ello, hace unos años, nace una nueva disciplina científica, la Bioinformática [6], que es la intersección entre la Biología y la Computación. La Bioinformática responde a las necesidades como la obtención de datos, almacenamiento, análisis e integración que la investigación genómica genera.

Se ha convertido en una parte importante en muchas áreas de la biología. En la biología molecular experimental, aplicando técnicas tales como el procesamiento de imágenes y la extracción de los datos útiles a partir de grandes cantidades de datos en bruto. En el campo de la genética y la genómica, ayuda en la secuenciación, análisis en el genoma y a la observación de las mutaciones. Por otro lado, desempeña un papel importante mediante la minería de datos en la literatura biológica, desarrollo de ontologías de genes biológicos, permitiendo una organización y consultas de datos biológicos.

Uno de los aspectos más importantes de la Bioinformática es la identificación de genes dentro de una larga secuencia de ADN [2].Hasta el desarrollo de la Bioinformática, la única manera de localizar los genes a lo largo de un cromosoma fue estudiar su comportamiento en organismos vivos o aislar el ADN y estudiarlo en un tubo de ensayo (*in vitro*). La Bioinformática permite a los científicos hacer conjeturas acerca de dónde se encuentran los genes

simplemente mediante el análisis de datos de la secuencia utilizando un ordenador.

1.2 Secuenciación de ADN

La secuenciación de ADN es el proceso que determina la secuencia de bases de los nucleótidos [A, T, C, G] de un fragmento de ADN. Hoy en día, con el equipo y los materiales adecuados, secuenciar un fragmento de ADN es relativamente sencillo.

Secuenciar un genoma completo (todo el ADN de un organismo) sigue siendo una tarea más compleja. El proceso consiste en romper el ADN del genoma en muchos trozos más pequeños, secuenciar dichos trozos y ensamblar las secuencias en una única y larga "secuencia consenso". Sin embargo, hoy en día secuenciar un genoma es más rápido y menos costoso de lo que resultó el Proyecto Genoma Humano. Por ejemplo, durante el periodo de abril de 1999 a junio de 2000 se estimó que generar un "borrador" inicial del genoma humano costó aproximadamente 300 millones de dólares en todo el mundo [50].En 2004 esa cantidad había descendido hasta los 14 millones de dólares aproximadamente. En el año 2017 se sitúa en torno a los 1.000 dólares [4].

1.2.1 Secuenciación de Sanger

La secuenciación del ADN tiene distintas maneras de llevarse a cabo, una de ellas es la secuenciación mediante Sanger.

En 1975, Frederick Sanger desarrolló el método de secuenciación de ADN conocido como método de Sanger o Secuenciación de Sanger. Dos años más tarde empleó esta técnica para secuenciar el genoma del bacteriófago Phi-X174 [32], el primer ácido nucleico secuenciado totalmente en la historia. Este trabajo fue la principal base para el Proyecto Genoma Humano.

Este método de secuenciación consiste en la polimerización del ADN [1] y el uso de dideoxinucleótidos que sirven como terminadores de la reacción. En la actualidad la relación de secuenciación se basa en una modificación de la PCR (*Polymerase Chain Reaction*, Reacción en Cadena de la Polimerasa) con dideoxinucleótidos marcados con fluoruros y se resuelve mediante electroforesis capilar.

1.2.2 Secuenciación química

El método de secuenciación química fue propuesto por Maxxam y Gilbert, fue publicado en 1977 en la revista PNA [3], 10 meses antes que Sanger, mencionado anteriormente, publicara el método de secuenciación enzimática.

Esta técnica consiste en romper las moléculas marcadas con reacciones químicas específicas para cada una de las cuatro bases [A, T, G, C]. Cuatro partes proporcionales de la muestra se tratan bajo condiciones distintas, posteriormente el tratamiento con piperidina [42] rompe la molécula de ADN a nivel de base modificada. De ese modo se genera una serie de fragmentos marcados a partir del final radiactivamente hasta el primer lugar de "corte" de cada molécula. Los fragmentos posteriormente se separan por tamaño mediante electroforesis en gel, separando los productos de las cuatro reacciones en cuatro carreras distintas, pero una al lado de la otra. Para visualizar los fragmentos marcados en cada reacción, se hace una autoradiografía del mismo, lo que proporciona una imagen de una serie de bandas oscuras correspondientes a los fragmentos marcados con el radioisótopo, a partir de las cuales se puede inferir la secuencia.

Este procedimiento ha quedado en desuso debido a su complejidad técnica, el uso extensivo de productos químicos peligrosos y dificultades para escalarla.

1.3 MinION

Desde que el Proyecto Genoma Humano se completó hace más de una década, numerosos científicos han tratado de diseñar nuevas estrategias, con menor tasa de error y de una forma más rápida y barata. En búsqueda de abaratar los costes se desarrollaron los denominados secuenciadores de segunda generación [15], capaces de generar cientos de miles de reacciones de secuencias en paralelo (secuenciadores de verdadero alto rendimiento [*high-throughput*]) gracias a la inmovilización de las reacciones en una superficie sólida. De esta forma, la cantidad de reactivos necesarios se minimiza al máximo (podrían llamarse nanoreacciones) y se abarata el coste por base leída.

En los últimos años apareció lo que se puede percibir como la tercera generación; una que permite la lectura libre de cationes de ampliación de moléculas de ADN individualmente en largos períodos consecutivos [49].

Actualmente, esta nueva generación está dominada por dos métodos: la secuenciación nanopore y la secuenciación en tiempo real de una sola molécula (SMRT), definidas por Oxford Nanopore Technologies (ONT) [22] y Pacific Biosciences (PacBio) [46], respectivamente.

Desde la introducción del primer dispositivo de secuenciación de nanopore disponible en el mercado, el MinION de Oxford Nanopore technologies y el inicio del programa MinION Access (MAP) en 2014 [20], el campo de la secuenciación ha avanzado a gran velocidad; actualmente hay un gran número de nuevas aplicaciones y mejoras a las ya existentes que son publicadas periódicamente.

Para su funcionamiento, MinION utiliza una pieza desechable llamada *flowcell*, en la que se coloca una muestra de ADN previamente tratada con un químico específico. Los nucleótidos o letras que componen el ADN se diferencian debido a los cambios de corriente eléctrica que se producen durante el paso por la membrana. La energía necesaria para su funcionamiento se extrae de un puerto USB del ordenador al que está conectado y además dispone de su propio programa de análisis. El sistema está diseñado para detectar muestras complejas como sangre o plasma y se puede adaptar para secuenciar ADN, ARN, detectar proteínas y pequeñas moléculas utilizando técnicas basadas en detección por nanoporos.

Figura 1. Colocación de una muestra de ADN en MinION

Las ventajas de MinION frente a otros dispositivos de secuenciación son numerosas. Tanto su tamaño, aproximadamente el tamaño de un teléfono

móvil, y su costo, 1000$ por un kit de arranque. El tiempo de ejecución en MinION también es razonable en relación tiempo-coste-rentabilidad. Además, a medida que MinION va obteniendo resultados, estos pueden consultarse y ser tratados en tiempo real.

Dado su tamaño y precio, relativamente bajo comparado con otros dispositivos que realizan funciones similares, MinION es muy utilizado en la comunidad científica. Se ha usado para secuenciar el genoma completo del virus de la gripe [33] y para ver la evolución del virus del Zika y el Ébola en tiempo real, ayudando a los científicos a romper con la cadena de transmisión de dichos virus [55]. Por otro lado, también la NASA ha usado el dispositivo en la Estación Espacial Internacional (ISS) para comprobar la fiabilidad del MinION.

1.4 Jupyter Notebook

Jupyter Notebook [40], anteriormente conocido como IPython Notebook, es una herramienta web muy simple, es un intérprete del lenguaje de programación Python, que funciona con celdas independientes de código y de texto enriquecido.

En el momento en el que IPython Notebook pasó a ser Jupyter Notebook, esta herramienta añadió soporte para otros lenguajes de programación, actualmente soporta más de 40 leguajes de programación, como pueden ser R, Julia, Scala y el mismo Python.

Una de las principales características de Jupyter es que permite presentar código de forma diferente. En lugar de archivos ejecutables con comentarios, o archivos de texto/plano/HTML/LaTex con fragmentos de código estático, un *notebook* puede contener hasta dos tipos distintos de celdas. Para la parte de documentación se permite introducir texto en formato HTML, Markdown [29] o LaTex. Por otro lado, Jupyter cuenta con celdas de código, en las que se permite escribir código ejecutable, con un espacio del documento reservado para obtener salida del código ejecutado.

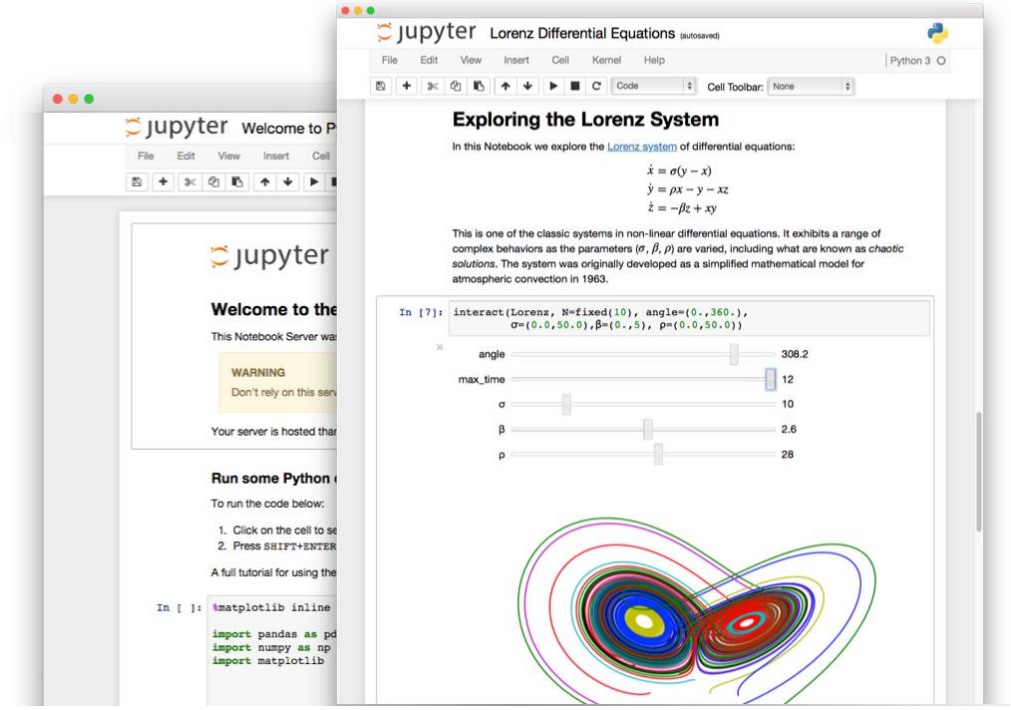


Figura 2. Jupyter Notebook

Jupyter da la posibilidad de poder trabajar con varios lenguajes simultáneamente en el mismo notebook, esta característica lo hace muy flexible. Por otro lado, se ha convertido en un gran aliado en el campo de la computación científica y la ciencia de datos al soportar lenguajes de programación con gran uso en estos campos dada la facilidad que tiene para incorporar librerías que ayuden a un mejor desempeño.

1.4.1 JupyterLab

A raíz de Jupyter Notebook nace JupyterLab [27], el cual se pensó para sustituir a Jupyter Notebook a partir de la versión 1.0. JupyterLab ofrece todo lo que los componentes básicos heredados del clásico Jupyter Notebook, en una interfaz de usuario flexible y poderosa que se puede extender a través de extensiones de terceros.

Recientemente salió su versión 1.0. Para el desarrollo de este Trabajo de Fin de Grado se empezó con Jupyter Notebook, pero a lo largo del desarrollo se ha pasado a probar JupyterLab, siendo su uso mucho más intuitivo que el de Jupyter Notebook, pero sin perder toda su gran funcionalidad.

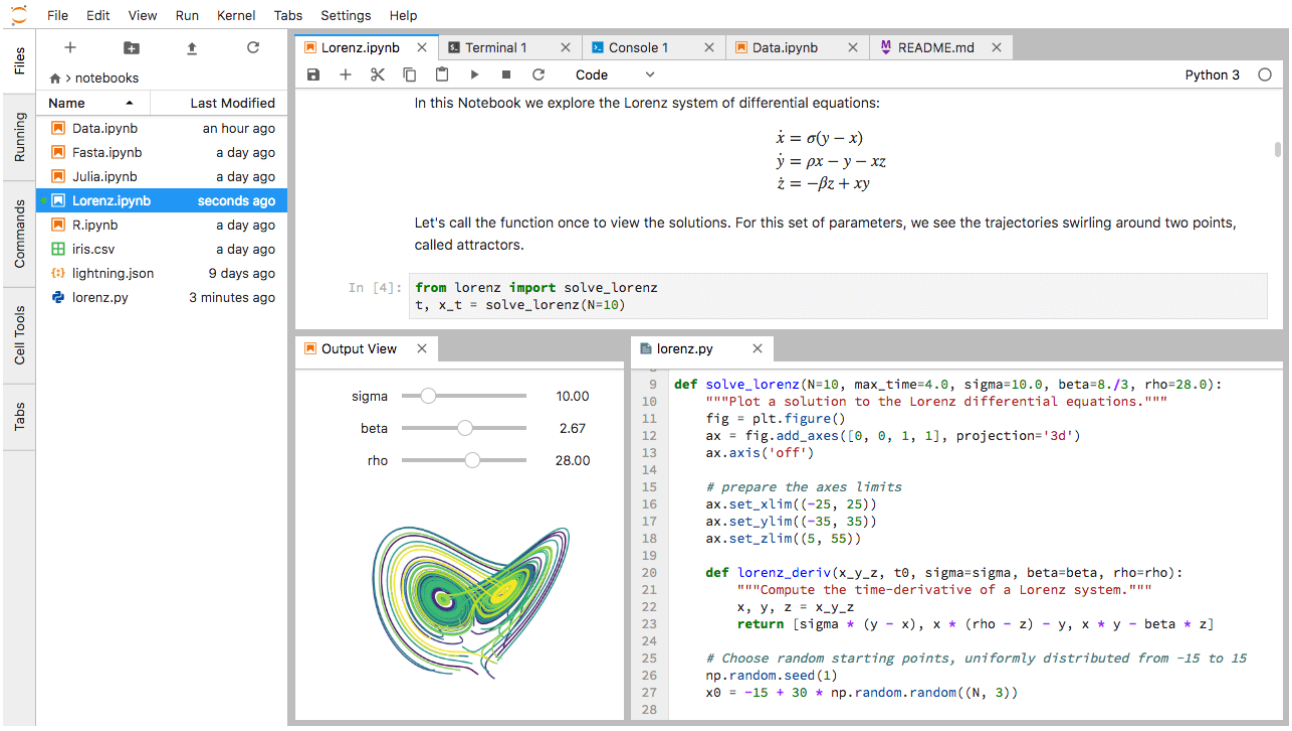


Figura 3. Interfaz de JupyterLab

1.5 BLAST

Un problema muy común en Bioinformática es la búsqueda de secuencias que tengan alguna coincidencia con alguna otra secuencia ya existente (ya secuenciada) de una base de datos. Por ejemplo, si se tiene una secuencia de un virus, se podría buscar en una base de datos secuencias similares para, entre otras cosas, asignar una especie.

BLAST (*Basic Local Alignment Search Tool*) da solución a este problema. Esta herramienta fue desarrollada por los Institutos Nacionales de Salud del gobierno de EE.UU., por lo que es de dominio público y puede usarse gratuitamente desde el servicio del Centro Nacional para la Información Biotecnológica (NBCI) [34].

BLAST es un programa informático de alineamiento de secuencias de tipo local, ya sea de ADN, ARN o de proteínas. El programa es capaz de comprobar una secuencia contra una gran cantidad de secuencias que se encuentran en una base de datos.

BLAST puede se puede utilizar mediante su aplicación web, a la cual se puede acceder sin necesidad de registro, y se puede utilizar mediante su suite

local disponible para Windows, Linux o Mac. También cuentan con API para desarrolladores.

Los biólogos utilizan BLAST normalmente en su versión web. La web es bastante configurable y de ella se pueden obtener unos resultados u otros, esto es, que mediante la configuración la búsqueda puede ser más o menos estricta a la hora de comparar.

En la misma se pueden obtener los resultados, la aplicación permite ver directamente en la web con quien encontró coincidencias indicando un porcentaje de coincidencia entre la secuencia buscada y la encontrada. Por otro lado, si bien es cierto que no necesita de autenticación alguna para realizar una búsqueda, si se cuenta con acreditación, y la página guarda los resultados de la búsqueda por un plazo de 48 horas, pudiendo acceder a dichos resultados en cualquier momento dentro del plazo establecido.

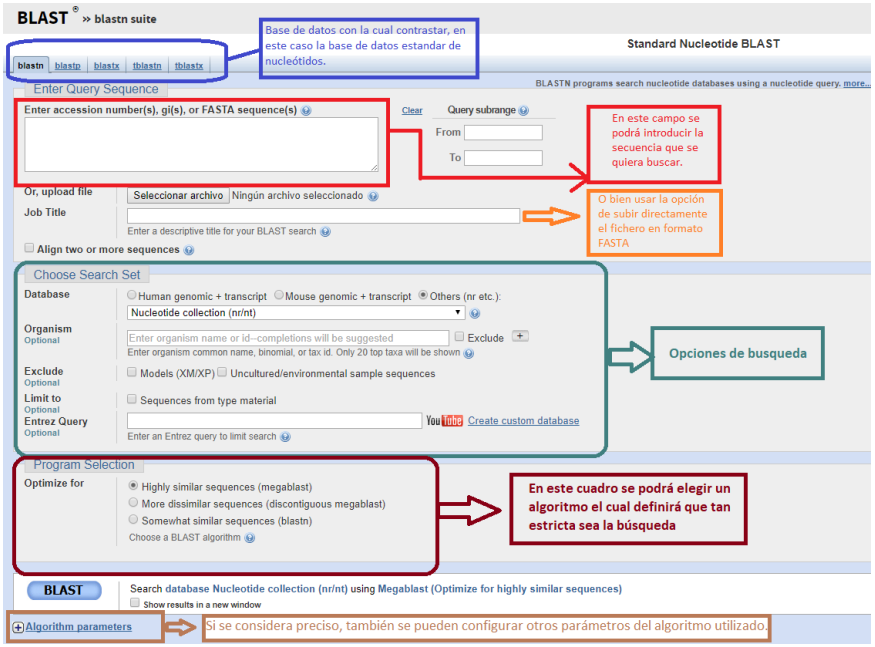


Figura 4. Interfaz de la versión web de BLAST [56]

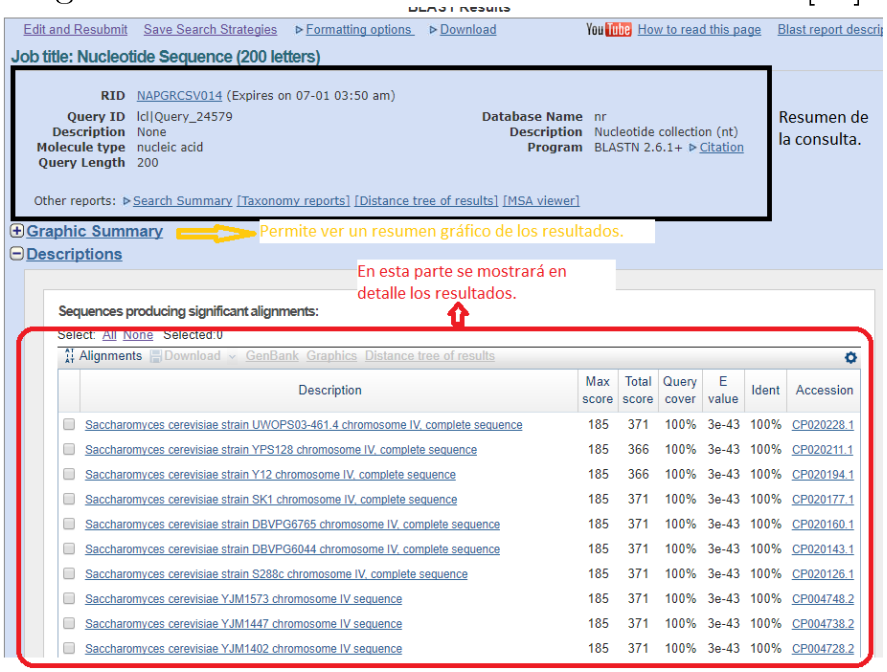


Figura 5. Resultados tras utilizar BLAST con una secuencia en coincidencia con otros nucleótidos

Nota: para ver exactamente en qué coincide la secuencia bastará con ir al enlace de la coincidencia.

Cabe destacar que, aunque BLAST puede dar coincidencias no quiere decir que esa secuencia sea buena, es decir, las bases de datos utilizadas por BLAST son las que tiene el NCBI y éstas son utilizadas por la comunidad científica y también ponen sus propios datos a la disposición de otros. Al suceder esto, una

secuencia puede tener coincidencia con una secuencia errónea si cuando se colocó en las bases de datos esta presentaba un error. Esto es importante dado que posteriormente se utilizará BLAST para comprobar las secuencias erróneas, de manera que si se encuentra coincidencia puede que este sea un error.

En el siguiente capítulo presentamos el proyecto desarrollado con las herramientas y técnicas descritas en este apartado.

1.6 Objetivos y requisitos

El objetivo principal de este Trabajo de Fin de Grado es desarrollar un notebook de Jupyter en el que se analicen los ficheros generados por el secuenciador MinION.

El notebook de Jupyter permitirá buscar errores en las secuencias, antes de ensamblar, generadas por MinION de manera que se pueda optimizar el ensamblaje de dichas secuencias. Los ficheros de MinION con los que se ha trabajado tienen el formato FASTQ [48] FASTA [8] el cual es un formato libre con el cual trabajar sin necesidad de utilizar software intermedio como en el caso de FAST5.

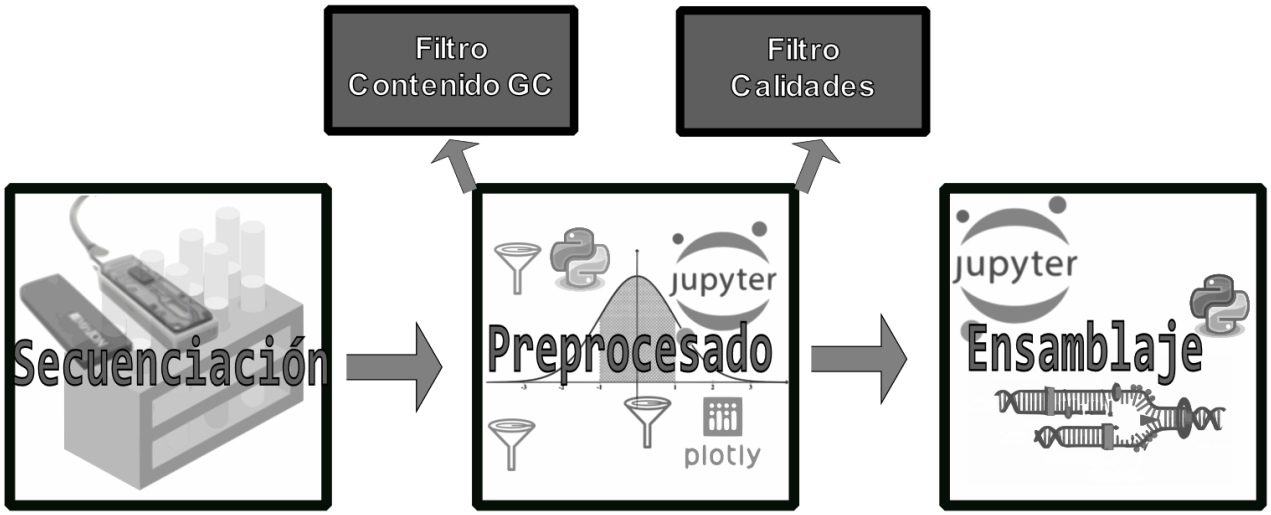


Figura 6. Esquema inicial, la herramienta a desarrollar será la parte del preprocesado

Los datos con los que trabajaremos serán datos de una bacteria secuenciados con la química R9 [13]. Esta herramienta complementará otra herramienta desarrollada con anterioridad en el Trabajo de Fin de Grado [25] realizado Héctor Rodríguez Pérez en 2016.

Capítulo 2. Diseño y desarrollo del cuaderno

2.1 Requisitos para el diseño del notebook en Jupyter

Una vez realizado un análisis de las herramientas se fijaron distintos requisitos con los que se debe dotar el notebook de Jupyter:

- Se deberá poder trabajar con ficheros en formatos con los que se trabajan actualmente en el campo de la genómica.
- Utilizando las librerías que se consideren adecuadas, se deberán poder mostrar gráficamente los resultados. De esta manera el usuario podrá visualizar previamente los datos obtenidos una vez aplicados los filtros pudiendo así ver si efectivamente se han conseguido mejoras y decidir si parar la ejecución o terminar con la extracción de los datos.
- Para aprovechar al máximo la herramienta no solo se podrán ver los datos gráficamente, sino que también se deberá dar la posibilidad de exportar los datos, separando las lecturas buenas de las malas.
- También se deberá tomar en cuenta los usuarios que no deseen separar las secuencias buenas de las malas en ficheros distintos, sino que deseen verlas directamente en el notebook.
- Aprovechando la flexibilidad que ofrece Jupyter también se deberá poder mostrar una tabla de coincidencias entre los distintos filtros, es decir, una vez se hayan filtrado los datos y separados las secuencias buenas de las malas se compararán las coincidencias de las lecturas malas. También se deberá dar posibilidad de separar entre lecturas buenas y malas como si de otro filtro se tratara.

- Utilizar las bases de datos genómicas, mediante BLAST, para asegurar que los métodos usados y datos obtenidos son fiables.

2.2 Formato y estructura de los datos.

2.2.1 Entradas

En el análisis de datos de secuencias de nucleótidos, como las que genera el secuenciador MinION se trabaja con el formato FAST5 que luego se suele convertir a los formatos de ficheros FASTQ y FASTA. Estos suelen contener una o más entradas donde cada una corresponde a una secuencia o lectura.

El formato FASTQ almacena secuencias y calidades Phred [52] en un único archivo, siendo conciso y compacto. FASTQ es ampliamente utilizado en Bioinformática por tanto por lo general se toma FASTQ como estándar. Existen distintos tipos de FASTQ entre los que podemos encontrar Solexa/Illumina o Sanger/Ilumina que se diferencian en las calidades, ya que escalan de manera diferente.

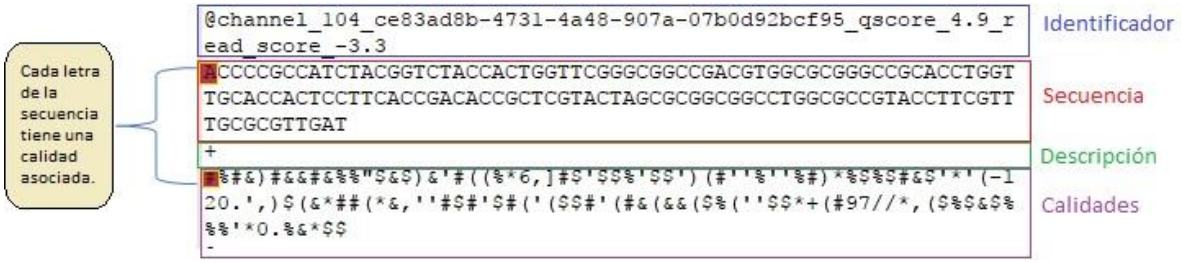


Figura 7. Formato FASTQ

Notación del formato FASTQ: un fichero FASTQ normalmente utiliza líneas por secuencia.

- **Línea 1:** Empieza por el carácter por el carácter '@' seguido del identificador de la secuencia.
- **Línea 2:** En esta línea estará conformada por los caracteres de la secuencia [A, T, G, C]
- **Línea 3:** La tercera línea comienza con el carácter '+' y está opcionalmente el identificador y una descripción.

- **Línea 4:** En la cuarta y última línea de la secuencia se encontrará las calidades asociadas a la secuencia (línea 2), por tanto, ésta debe contener tantos símbolos como letras de la secuencia, es decir, un símbolo por cada letra de la secuencia.

El formato FASTA, al igual que el FASTQ, almacena secuencias. Cada línea representa una secuencia. La simplicidad del formato hace fácil manipular y analizar secuencias utilizando herramientas de procesado de texto y lenguajes de script como Python o PERL.

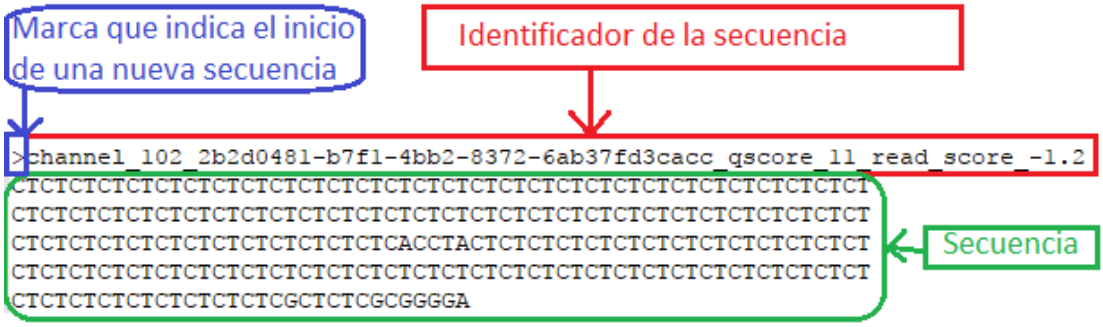


Figura 8. Formato FASTA

Notación del formato FASTA: un fichero FASTA utiliza una única línea por secuencia. En esta línea la información se guarda de la siguiente manera: cada línea empieza por el símbolo '>' seguido de un identificador y a menudo con información adicional.

2.2.2 Salidas

En la herramienta se da la posibilidad de configurar las salidas. Esto incluye, aparte del directorio donde se quiere guardar, el formato en el que se van a dividir las secuencias (secuencias buenas y malas). Estas salidas podrán ser igual que las de entrada, FASTQ y FASTA, y además gráficamente.

Por cada uno de los filtros se creará un gráfico, el cual se guardará en formato HTML de manera que la representación sea interactiva con el usuario, dando la posibilidad de hacer 'zoom' y ver, por ejemplo, la posición de una secuencia en concreto con solo situar el cursor encima del gráfico.

Los gráficos también cuentan con la funcionalidad de descargar la imagen en el formato de imagen PNG o guardarla en la nube, en cuyo caso se deberá poder contar con una cuenta en Plotly [39].

2.2.3 Estructura de directorios

Para facilitar la utilización de la herramienta se recomienda utilizar la siguiente estructura de ficheros.

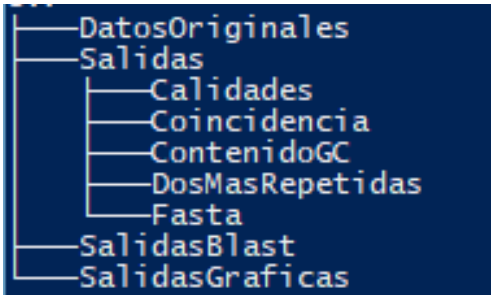


Figura 9. Estructura de directorios utilizados

2.3 Herramientas utilizadas

Antes de pasar al desarrollo se decidió analizar las herramientas/librerías disponibles que puedan resultar de ayuda al desarrollo. A continuación, se dará un listado de las librerías analizadas, además un pequeño resumen de la razón por la que se elige o descarta dicha librería.

- **Scikit-bio [17]:** es una librería de código abierto, utiliza la licencia BSD para Python 3 y proporciona una serie de estructuras de datos, algoritmos y recursos educativos para la Bioinformática.
- **Biopy [18]**: es una pequeña librería para Python 2.7 que permite explorar y desarrollar ideas filogenéticas. Biopy es útil para generar datos simulados y la realización de un análisis sin necesidad de conexión a Internet. También incluye comandos para utilizar en una terminal que pueden resultar útil para los usuarios.
- **Biopython [19]:** es un conjunto de herramientas para la computación biológica, y está disponible para Python 3. Además, esta librería es el resultado de una colaboración de librerías y aplicaciones que responden a las necesidades de trabajo actual y futuro de la Bioinformática en Python. El código de esta librería está bajo la licencia Biopython licence [7]. Es libre y compatible con la mayoría de las licencias.

  La razón por la que se eligió esta librería y no las dos anteriores es que cuenta con una documentación muy amplia y a su vez proporciona al desarrollador ejemplos del código para entender mejor su

funcionamiento. Además, existen muchos hilos en foros en la que se habla de la librería, la cual es de gran ayuda de cara posibles problemas.

- **Matplotlib [30]**: es una librería gráfica en 2D, esta trabaja a partir de listas o arrays en Python y su extensión matemática NumPy. Además, proporciona una API, pylab, diseñada para recordar a la de MATLAB.
- **Plotly:** es una plataforma moderna para inteligencia de negocios y ciencia de datos. Cuenta con API'S para distintos lenguajes de programación entre los que se encuentra Python.
    - **Plotly en Python:** es una biblioteca gráfica de Python que permite crear gráficos interactivos, con calidad para poderlos publicar en línea.

  La razón por la que se eligió Plotly y no Matplotlib es que esta permite crear gráficos interactivos, además está también permite mayor personalización y tiene buena integración con Jupyter Notebook, puesto que se pueden incrustar los gráficos interactivos, en HTML directamente en una celda de Jupyter Notebook sin necesidad de salir para poder interactuar con el gráfico. Por otro lado, también permite descargar el gráfico como imagen y/o guardar el gráfico en la nube.
- **Numpy [35]:** es una extensión de Python, que le agrega mayor soporte para vectores y matrices, constituyendo una biblioteca de funciones matemáticas de alto nivel para operar con esos vectores o matrices.

  La principal razón por la que se eligió esta librería es por sus funcionalidades con los vectores o matrices, además de que se acopla muy bien con las funcionalidades de Biopython, dado que requiere que se instale dicha librería ya que la utiliza internamente.
- **Pandas [37]:** es la librería de Python para la manipulación y análisis de datos. En particular, Pandas ofrece estructuras de datos y operaciones para la manipulación de tablas numéricas, series, paneles (Panel, Panel4D y Panel ND).

  Esta librería se complementa muy bien con las estructuras de datos de Numpy y es una de las librerías más utilizadas en Python. Por ello,

hay bastante documentación con la que buscar en caso de que surjan problemas.

- **Beautiful Soup [10]:** es una librería de Python para extraer datos de archivos HTML y XML. Proporciona métodos con los cuales navegar, buscar y modificar datos de árbol de análisis sintáctico. Utiliza un conjunto de herramientas que permiten diseccionar un documento y extraer lo que se necesite.

  Esta librería es muy completa y es muy utilizada para hacer uso de técnicas de web *scraping*, lo cual facilita su utilización.

- **URLLib2 [53]:** es una librería que define funciones y clases que ayudan a la apertura de URL (principalmente HTTP) fácilmente, la cual está dotada de métodos para, por ejemplo, realizar autenticación básica, redirecciones, cookies y muchas otras funcionalidades.

  Esta librería, además de ser completa, se complementa muy bien con la librería Beautiful Soup, mencionada anteriormente, que hace que su uso sea bastante amigable y necesario para realizar extracciones de datos de páginas web.

2.4 Notebook generado

Una vez analizadas y elegidas las librerías con la que vamos a trabajar pasamos a la construcción del Notebook. Inicialmente no teníamos ninguna aplicación que sirviera de referencia en lo que pretendíamos hacer, por tanto, el desarrollo se hizo en distintas fases e ir viendo como adecuar la herramienta según se iba avanzando.

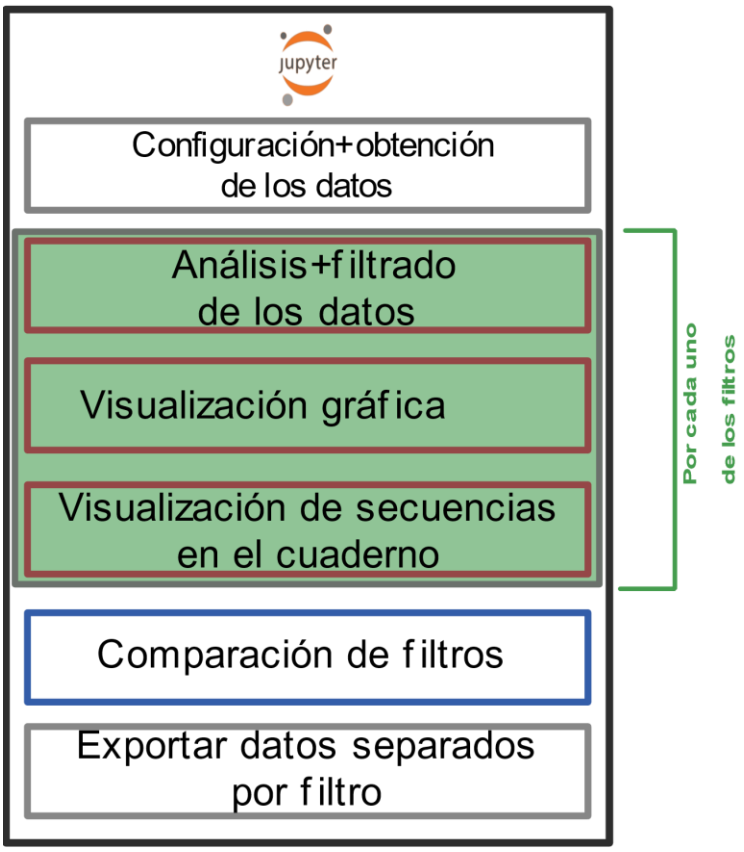


Figura 10. Estructura del Notebook generado

El notebook generado se encontrará dividido en distintas partes. En el primer bloque estarán los datos de entrada, así como los distintos parámetros de configuración. En el segundo bloque con los datos y parámetros del primer bloque se pasará a realizar un análisis de los datos para luego aplicar un filtrado, por consiguiente, se mostrarán una gráfica con el resultado del filtro aplicado (esto se repite por cada uno de los filtros).

Luego de tener los resultados de cada filtro se realiza una comparación y esos datos se contrastan con BLAST. Por último, se exportarán los resultados en ficheros separados para su posterior utilización.

Dado que el código está escrito en Python se han utilizado las normas PEP8 [38] para seguir un estándar en el código en la medida de lo posible. Por otro lado, en la organización de las celdas del notebook, puesto que no hay de momento ninguna norma que diga cómo se debe organizar, se intentó seguir unas transparencias, *Clean Code in Jupyter Notebooks* [11], donde se habla del tema.

2.4.1 FASE 1: Filtro de por repetición de caracteres.

Al principio del proyecto, se analizó manualmente el contenido del fichero FASTQ que teníamos como muestra. Entre las lecturas se aprecian las secuencias *"fallidas"* ya que mostraban un comportamiento atípico en bacterias. Los fallos se producían repitiendo dos caracteres, de [A, T, G, C], por tanto, se pasó a crear un método que permitiera buscar en las secuencias el porcentaje de los dos caracteres más repetidos.

Una vez detectados los dos caracteres más repetidos se pasaba a categorizar la lectura como buena o fallida. Para poder hacer la discriminación previamente se debería definir un umbral. Este umbral nos servirá como punto de corte, es decir, todas las lecturas cuyos dos caracteres más repetidos, sumados los porcentajes, supera el umbral fijado se considerará como lectura fallida.

Para dar los resultados visualmente, utilizando Plotly, se crearon dos gráficos; un gráfico de barras donde se puede apreciar todas las secuencias, diferenciadas por el color, un color para las lecturas buenas y otro para las fallidas, y aparte un gráfico de pastel mostrando el porcentaje de los fallos obtenidos en todas las lecturas analizadas.

2.4.2 FASE 2: Filtro por Contenido GC.

Una vez se realizó el método mencionado en la FASE 1 se pasó a contrastar que los resultados eran válidos. Para ello nos reunimos con el Dr. Carlos Flores, quien nos indicó que los resultados eran lógicos pero que la forma de obtenerlos era muy arbitraria, es decir, dado que el usuario es quien fija el umbral, la cantidad de fallos dependerá de cuan alto o bajo sea el umbral. Nos aconsejó que utilizáramos el Contenido GC [23].

En genética, GC, Contenido GC (Contenido de guanina [24] y citosina [12]) es una característica del genoma de un organismo o de cualquier pedazo de ADN o ARN expresado generalmente como porcentaje, y representa la cantidad de pares Guanina-Citosina en la molécula de ADN o genoma que está siendo investigado. Con dicho porcentaje se puede utilizar, por ejemplo, para clasificar organismos en taxonomías.

Para calcular el Contenido GC se debía analizar cada secuencia del fichero, al igual que el método anterior, utilizando los métodos de los que provee la

librería Biopython, aprovechando el uso de la librería también se pudo obtener el Contenido GC.

Para poder discriminar entre lecturas buenas o fallidas, y de la forma menos arbitraria posible, nos dispusimos a realizar una distribución normal de los datos obtenidos del Contenido GC. Para realizar la distribución normal se decidió utilizar la librería Matplotlib, dicha librería cuenta con la funcionalidad de generar una distribución normal de manera gráfica.

El problema con Matplotlib es su poca flexibilidad, ya que permitía ver la distribución, pero dada su gestión de los datos una vez realizada la distribución no se podía acceder a los datos originales con los que se realizó dicha distribución. Esto presentaba un problema, ya que luego no se podría extraer las secuencias erróneas.

Por tanto, se decidió utilizar Plotly como se utilizó en el método inicial para los gráficos de barra y de pastel. Plotly permitía más personalización y dejaba acceder a cierta parte de los datos, pero, al igual que Matplotlib, no podía acceder a todos los datos, por tanto, no se podría separar las secuencias. Así, para realizarla se decidió hacer dicha distribución propia, es decir, sacando la media, la desviación típica y fijando unos límites con estos cálculos. Para los límites se utilizó la formula estadística $[\bar{x} \pm iq * \sigma]$, donde $\bar{x} = media$, $iq =$ intervalo de confianza que se podrá configurar por el usuario, $\sigma =$ desviación típica.

Una vez se tenían todos estos cálculos se pasaba a realizar un diagrama de puntos, en el que cada punto representaba una secuencia, el eje "x" representa la posición dentro del fichero y el eje "y" el Contenido GC. Además, el gráfico cuenta con unas líneas que muestran los límites calculados de manera que todo lo que esté dentro de esos límites se consideraría lecturas buenas y todo lo que esté por fuera de estos, sería considerado como lecturas fallidas.

Una vez se obtuvieron estos resultados se pasó a verificar si estos datos y los métodos estadísticos empleados, eran los correctos. El Dr. D. Carlos Pérez comentó que los métodos eran buenos pero que, si se quería afinar más los resultados, sería bueno utilizar la media recortada [31] en la que para hacer el cálculo de la media se dejan fuera los datos atípicos de manera que se evite distorsión en el cálculo de la media. Para este cálculo se decidió dejar los datos

atípicos en un 10%, esto quiere decir que se quitarán de los datos, el 5% de valores muy bajos, y el 5% muy alto.

A nivel de código no supuso grandes cambios dado que una vez se ordenen los datos solo quedaría saber cuántas secuencias equivale el 5% para luego quitarlos. Sin embargo, en los resultados sí se apreció una importante variación, pasando a tener una diferencia de un 11% entre los datos antes y después de la utilización de la media recortada.

Además, al igual que el método anterior, se pueden ver las secuencias fallidas en el mismo Notebook sin necesidad de salir del mismo. También se exportan a ficheros separados, cumpliendo con los requisitos, los datos obtenidos.

2.4.3 FASE 3: Filtro por calidades

Una vez se obtuvieron unos resultados menos arbitrarios se decidió mirar otras características de las secuencias, en este caso, de las secuencias en FASTQ, pues, como se mencionó anteriormente, las secuencias en el formato FASTQ tienen información que indica la calidad de cada uno de los caracteres de cada secuencia. Partiendo de esto, se pasó a analizar esta característica. Para ello se usaron los métodos de los que dispone Biopython para poder obtener las calidades de las secuencias.

Como en la fase anterior se obtuvieron buenos resultados de manera automática, para las calidades se decidió utilizar el mismo método de realizar la distribución manualmente y con la media recortada. Una vez obtenida la distribución se podía ver una leve diferencia entre los porcentajes obtenidos con el Contenido GC y con las calidades.

Una vez reunidos con el Dr. D. Carlos Flores, nos indicó que en las calidades no se debería limitar en la parte superior, dado que mientras más altas son las calidades mejor resultado es y, por tanto, no se debería categorizar como lectura errónea.

Una vez quitado el límite superior se obtuvo un porcentaje mayor de lecturas erróneas.

2.4.4 FASE 4: Búsqueda de coincidencias de secuencias en los resultados de los filtros.

Al tener diferencia en los resultados, manualmente, se decidió comprobar si las secuencias eran las mismas, llegando a la conclusión que no lo eran. Pero como los resultados siempre dependerán del tamaño del fichero se decidió crear un método para comparar las lecturas erróneas obtenidas con ambos métodos, Contenido GC y Calidades.

Para poder realizar las comparaciones, las opciones disponibles eran comprobar los caracteres de cada una de las secuencias obtenidas en un método y buscándola en el otro o bien a la hora de separar, entre lecturas buenas y erróneas, cambiar la estructura de datos de un vector a un diccionario, de manera que al diccionario se le pueda añadir un índice, que será la posición del fichero donde se encontró la lectura errónea.

Como tanto en el método de Contenido GC como en el de las Calidades se utiliza el mismo fichero, las posiciones también serán las mismas, de manera que a la hora de comparar si el índice de un método está en el otro, habrá coincidencias.

Para que el usuario viera de manera visual este proceso se utilizó la librería Pandas, de forma que pueda distinguir en columnas las posiciones de cada secuencia calificada como errónea por los métodos de Contenido GC y Calidades. Cuando exista coincidencia se mostrará en distinto color de las que no coinciden, de forma que el usuario pueda ver rápidamente cuales secuencias coindicen en ambos métodos.

2.4.5 FASE 5: Contrastar secuencias fallidas con BLAST.

Como se mencionó en los requisitos de la herramienta, se debería utilizar la base de datos genómica BLAST para contrastar los datos obtenidos. Como se habló en capítulos anteriores, BLAST es una base de datos genómica la cual se puede utilizar para buscar coincidencia de secuencias en una gran base de datos.

En este caso se pretende utilizar BLAST como método de corroboración, es decir, una vez se obtengan las lecturas fallidas se pasarán por BLAST para ver si existe alguna coincidencia.

BLAST puede utilizarse en local, tanto por consola como a través de comandos en el mismo notebook. El inconveniente es que esta base de datos habría que descargarla e instalarla localmente, lo cual exige configuraciones adicionales que para un usuario normal no le resultaría fácil. Además, el tamaño de dicha base de datos es muy grande por lo que aparte de mucho espacio libre requiere una buena conexión a Internet para tardar lo menos posible en utilizarla. Se debe tener en cuenta que tener la base de datos en local, se necesitará un mantenimiento adicional durante el cual no se podría utilizar.

Existe, también, la posibilidad de utilizar BLAST mediante la aplicación web, tal como se mencionó en capítulos anteriores y, en este caso, con Biopython ya que cuenta con métodos que permiten realizar consultas a BLAST.

```
8 from Bio.Blast import NCBIWWW
9 from Bio import SeqIO
10
11 for i, sec in enumerate(SeqIO.parse('Salidas/Fasta/errores.fasta',
12                                      'fasta')):
13     result = NCBIWWW.qblast("blastn",
14                             "nt",
15                             sec.format("fasta"))
16
```

Figura 11. Manera de utilizar BLAST mediante Biopython.

Para saber si se encontró coincidencia o no en BLAST utilizando este método se deberá guardar el contenido de, en este caso, la variable *result*. Una vez guardado se puede examinar ya sea manualmente o bien mediante Python. Normalmente la salida se guardará en formato XML, pero puesto que BLAST puede guardar la salida en distintos formatos cualquiera de ellos valdría.

```
<?xml version="1.0"?>
<!DOCTYPE BlastOutput PUBLIC "-//NCBI//NCBI BlastOutput/EN" "
http://www.ncbi.nlm.nih.gov/dtd/NCBI_BlastOutput.dtd">
<BlastOutput>
  <BlastOutput_program>blastn</BlastOutput_program>
  <BlastOutput_version>BLASTN 2.6.1+</BlastOutput_version>
  <BlastOutput_reference>Zheng Zhang, Scott Schwartz, Lukas Wagner, and Webb Miller (2000), &quot;A
  greedy algorithm for aligning DNA sequences&quot;, J Comput Biol 2000; 7(1-2):203-14.
  </BlastOutput_reference>
  <BlastOutput_db>nr</BlastOutput_db>
  <BlastOutput_query-ID>Query_193145</BlastOutput_query-ID>
  <BlastOutput_query-def>No definition line</BlastOutput_query-def>
  <BlastOutput_query-len>3827</BlastOutput_query-len>
  <BlastOutput_param>
    <Parameters>
      <Parameters_expect>10</Parameters_expect>
      <Parameters_sc-match>1</Parameters_sc-match>
      <Parameters_sc-mismatch>-2</Parameters_sc-mismatch>
      <Parameters_gap-open>0</Parameters_gap-open>
      <Parameters_gap-extend>0</Parameters_gap-extend>
      <Parameters_filter>L;m;</Parameters_filter>
    </Parameters>
  </BlastOutput_param>
<BlastOutput_iterations>
<Iteration>
  <Iteration_iter-num>1</Iteration_iter-num>
  <Iteration_query-ID>Query_193145</Iteration_query-ID>
  <Iteration_query-def>No definition line</Iteration_query-def>
  <Iteration_query-len>3827</Iteration_query-len>
<Iteration_hits>
</Iteration_hits>
  <Iteration_stat>
    <Statistics>
      <Statistics_db-num>42576813</Statistics_db-num>
      <Statistics_db-len>84415982</Statistics_db-len>
      <Statistics_hsp-len>0</Statistics_hsp-len>
      <Statistics_eff-space>0</Statistics_eff-space>
      <Statistics_kappa>0.46</Statistics_kappa>
      <Statistics_lambda>1.28</Statistics_lambda>
      <Statistics_entropy>0.85</Statistics_entropy>
    </Statistics>
  </Iteration_stat>
</Iteration>
</BlastOutput_iterations>
</BlastOutput>
```


Figura 12. Archivo XML con los resultados de una petición a BLAST de la cual no se obtuvo coincidencias

Para saber si se obtuvo coincidencias se deberá examinar el archivo XML. En él se encontrará una etiqueta llamada **<Hit>** que irá dentro de la etiqueta **<Iteration_hits>,** dentro de esta etiqueta irán todas las coincidencias que se obtuvieron.

Figura 13. Toda la información relacionada con una coincidencia

En el caso de hacer una única consulta a BLAST con Biopython no existe ningún problema. Sin embargo, el problema llega cuando se pretende hacer una gran cantidad de peticiones. Para empezar, no se puede enviar una ráfaga de peticiones por lo que se deberá poner un tiempo de espera para ejecutar múltiples solicitudes.

Aunque el tiempo podría ser un inconveniente según la cantidad de peticiones que se deseen realizar, el verdadero problema surge de la inestabilidad de BLAST con Biopython; es decir, no existe ningún problema, al menos de lo probado en el desarrollo de esta herramienta, con la secuencia en sí. No obstante, cuando se ejecutan varias peticiones el tiempo de respuesta puede variar. En la página web también existe este inconveniente, pero, al menos en las pruebas, ninguna pasaba de 10 minutos. En cambio, mediante Biopython sí excedían ese tiempo.

Para intentar buscar una razón se decidió probar con todas las secuencias erróneas de uno de los métodos mencionados anteriormente, en este caso se utilizaron los del método de Calidades, y hacer lo mismo tanto en la página web como mediante el uso de Biopython en un Notebook.

En la versión web, como ya se mencionó anteriormente, la secuencia que más tardó duró unos 10 minutos, en cambio los tiempos en el notebook variaban y

en ocasiones llegó a durar alrededor de una hora. Luego se intentó ejecutar por segunda vez el mismo fichero y los resultados eran distintos a los obtenidos anteriormente.

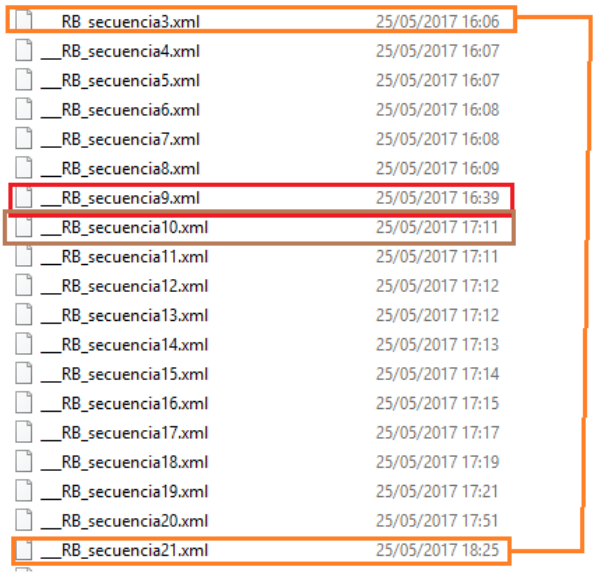


Figura 14. Captura de los tiempos de las secuencias utilizando BLAST con Biopython

Luego de tratar de buscar una solución viable se descartó el uso de la librería BioPython para acceder a BLAST y se decidió realizar un *script* a medida, explicado más adelante, para llevar a cabo este proceso.

2.4.6 FASE 6: Reestructuración del notebook.

Para poder notar la diferencia entre los filtrados explicados anteriormente se decidió calcular el Contenido GC del fichero FASTQ que se da como entrada, es decir, el fichero que se va a analizar y el fichero de salida con las lecturas buenas de cada uno de los filtros.

Teniendo en cuenta de que el Contenido GC es un porcentaje, se calculó la frecuencia relativa para cada una de las secuencias, de manera que luego se pueda calcular la frecuencia relativa acumulada y así obtener el Contenido GC total del antes y después de cada filtro.

Una vez terminado el desarrollo se pasó a mirar qué funcionalidades podrían mejorar la herramienta. Teniendo en cuenta que a la hora de filtrar por Calidades solo se puede hacer con archivos con formato FASTQ y que el Contenido GC se puede filtrar tanto en archivos con formato FASTQ y

FASTA, se decidió separar esos filtros en distintos notebooks, de esta manera, cuando se crea preciso, el usuario podrá realizar un único filtro, al hacer aplicar un único filtro salta el paso de las comparaciones. Al final quedarían tres notebooks, uno en el que se pueden hacer ambos filtros y uno para cada filtro.

Dado que Jupyter Notebook funciona con celdas y que estas celdas están codificadas, en Python se puede disponer de todas las funcionalidades del propio lenguaje. Por tanto, se decidió factorizar extrayendo el código que se repetía en las distintas celdas del Notebook en archivos .py (propios de Python). Finalmente, se han obteniendo unas 4 clases y en el Notebook se ha pasado de 89 celdas a 69 celdas más ordenadas y optimizadas.

Además, luego de utilizar varias veces la herramienta se observó que el uso de la librería Plotly en su versión Community tenía dos inconvenientes; para ejecutarlo se necesitaría estar registrado y además configurar un *token* con el que se asociaría todos los gráficos a la cuenta de Plotly. Por otro lado, al ser una versión Community solo permite 50 ejecuciones en su versión 1 y 250 ejecuciones en su versión 2. Para evitar este problema, se decidió utilizar una solución que es aceptada en Jupyter Notebook, aunque no está disponible para todas las plataformas, que es tener Plotly offline. Esta opción permite ejecutar todas las veces que se considere necesaria la herramienta.

2.4.7 FASE 7: Obtención de datos de prueba.

Para seguir probando la herramienta se decidió utilizar otros datos. Se necesitó utilizar, nuevamente, una de las herramientas de NCBI, en este caso la herramienta *Sequence Read Archive* (SRA) [44]. Esta pone a disposición de la comunidad de investigación datos de secuencias biológicas. En esta herramienta se puede buscar secuencias concretas, conociendo su identificador, o buscar por términos, en este caso se necesitaba algún fichero secuenciado con MinION y la química R9. Una vez se encuentra la secuencia que se desee, se deberá utilizar el kit de herramientas que facilita NCBI. Es una aplicación de consola en la que se puede poner el identificador del fichero deseado, y este se descargará en local para luego poder trabajar. La muestra descargada es una que lleva por nombre "*Other Sequencing of Mycobaterium bovis*" [36].

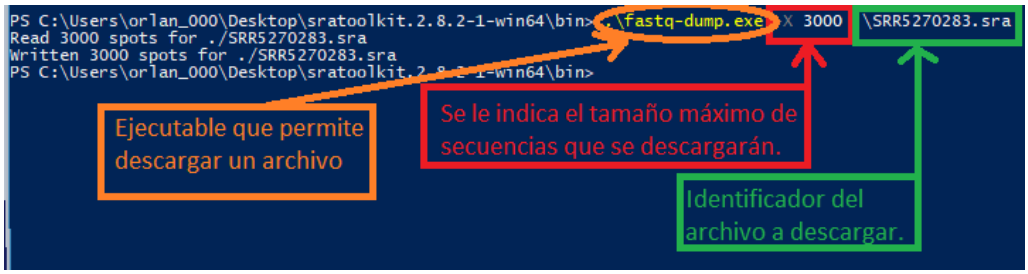


Figura 15. Manera de descargar muestras de SRA mediante consola

Con este fichero no se podría utilizar BLAST ya que el fichero está sacado de las bases de datos de NCBI, por tanto, en la secuencia tendría coincidencias. Se debe tener en cuenta que, aunque una secuencia esté en las bases de datos de NCBI no significa que la secuencia sea correcta.

2.4.8 FASE 8: Contrastar secuencias fallidas con BLAST con web scrapping

Dado los problemas mencionados en las fases de los apartados anteriores con la inconsistencia del uso de BioPython para hacer consultas con BLAST se buscaron alternativas, pero en la mayoría de los casos las opciones eran la ya utilizada, BioPython, o una versión similar, pero en lugar del lenguaje de programación Python se utilizaba el leguaje R [43].

Por tanto, se pensó en utilizar *web scraping* [57] directamente con la aplicación web, mencionada en capítulos anteriores, pero está hecha de alguna manera que no se muestren los parámetros obtenidos de los campos del formulario con el que se hace la consulta.

Para lograr el objetivo de hacer comparaciones con la base de datos BLAST de las secuencias filtradas como erróneas, se decidió utilizar directamente la API de BLAST [16] y utilizando tácticas de *web scraping*, obtener los resultados.

Tras analizar todos los parámetros disponibles se pasó a elegir los que se iban a utilizar. A continuación, se detalla cuáles son esos parámetros. Antes hay que comentar que los parámetros dependen, valga la redundancia, de un parámetro concreto, el parámetro CMD, el cual determinará si la consulta es de tipo PUT, GET o de tipo DELETE [54], aunque este último no se utilizará en este proceso.

- **CMD = PUT**

- **QUERY:** Consulta de búsqueda, esta tiene los siguientes formatos:
  - **Accession [5]**
  - **GI [45]**
  - **FASTA**
- **DATABASE:** Base de datos de ensamblado de genomas, tiene muchas opciones, para saber cuál elegir sería conveniente mirar la guía [9] en el que se podrá consultar todas las bases de datos disponibles. En el caso particular con el que se trabajó en el desarrollo de este Trabajo de Fin de Grado se utiliza la base de datos *Nucleotide Colletion* (nr/nt).
- **PROGRAM:** Tipo de búsqueda BLAST, esta dispone de distintos programas para según lo que se esté trabajando utilizar un programa u otro, ver la guía [8] para obtener más detalles de cada uno de estos programas. Los tipos pueden ser:
  - **BLASTn**
  - **BLASTp**
  - **BLASTx**
  - **tBLASTn**
  - **tBLASTx**

  Para el caso particular de este Trabajo de Fin de Grado, se utilizó BLASTn.

- **CMD = GET**

  Para realizar la petición GET primero se debe realizar una petición PUT previa ya que el resultado de esta es un parámetro llamado RID. Una vez se sabe cuál es el RID se podrá obtener el resultado de BLAST con los siguientes parámetros.
  - **RID:** (*The Request ID*) Identificador de la búsqueda que se ha obtenido de una petición PUT.
  - **FORMAT_TYPE:** es el formato con el que se obtendrá el resultado de la petición. A continuación, las opciones disponibles:

- **HTML**
- **TEXT**
- **XML**
- **XML2**
- **JSON2**

De entre todos los formatos, para este Trabajo de Fin de Grado se utilizó TEXT, es decir, texto plano.

Una vez decididos los parámetros se pasó a ver qué devuelve, en formato texto, que pueda ser útil para detectar si la secuencia buscada tiene alguna similitud con otra secuencia almacenada de la base de datos.

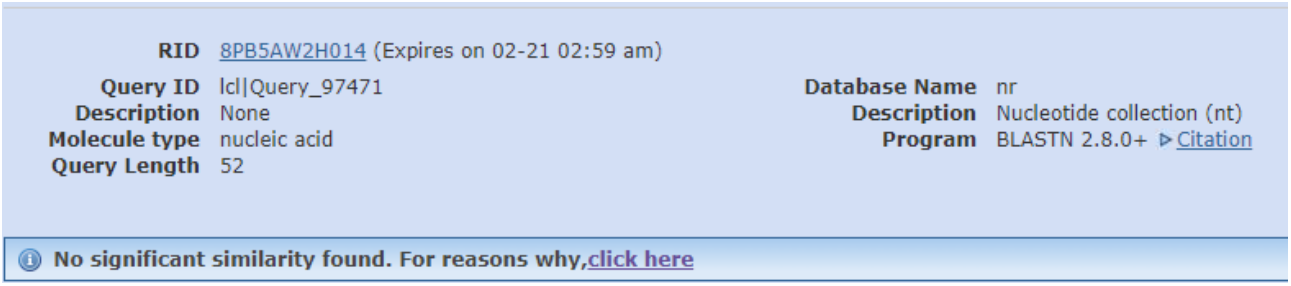


Figura 16. Resultado de BLAST cuando no se encuentra similitud con la secuencia requerida

Cuando no encuentra similitud, como se distingue en la imagen anterior, muestra el mensaje, aparte de los parámetros de búsqueda, "*No significant similarity found*". Puesto que esto es fácilmente identificable y los datos de similitud no se utilizan para nada más en este Trabajo de Fin de Grado se decidió utilizarlo para indicar al usuario si hay o no similitud. De otra manera y como se mencionó en capítulos anteriores, se necesitaría buscar un campo **<Iteration_hits>** en el archivo XML, que se tendría que descargar, y que se obtiene como resultado.

Cuestiones que se deben tener en cuenta a la hora de utilizar estos scripts.

- BLAST no permite secuencias de más de 8110 caracteres, hasta donde se ha llegado a probar.
- Si se van a realizar consultas masivas, se deberá añadir un tiempo de espera entre consulta y consulta ya que, de no hacerse, BLAST podría

detectar las peticiones como un intento de ataque, por tanto, pasará a rechazar las peticiones. El tiempo mínimo estipulado es de 10 segundos.

- Se debe tener en cuenta que, dependiendo de la secuencia, tardará más o menos en obtener los resultados. Por tanto, sería adecuado que entre una consulta PUT y una GET exista un tiempo prudente.
- Por último, si se realizan consultas masivas muy seguidas, BLAST podría añadir las consultas a una cola con poca prioridad, y esto puede añadir un tiempo mayor de respuesta.

Capítulo 3.
Resultados obtenidos

3.1 Resultados por filtro de las dos letras más repetidas.

Como se mencionó anteriormente, en la Fase 1, el método empleado para filtrar las dos letras más repetidas es muy arbitrario y por tanto los resultados siempre van a depender de cuan alto sea el umbral fijado. En este caso el umbral se colocó en un 80%, es decir, que para que se considerase como un error la combinación de las dos letras más repetidas debía superar este umbral.

Se obtuvieron los siguientes resultados.

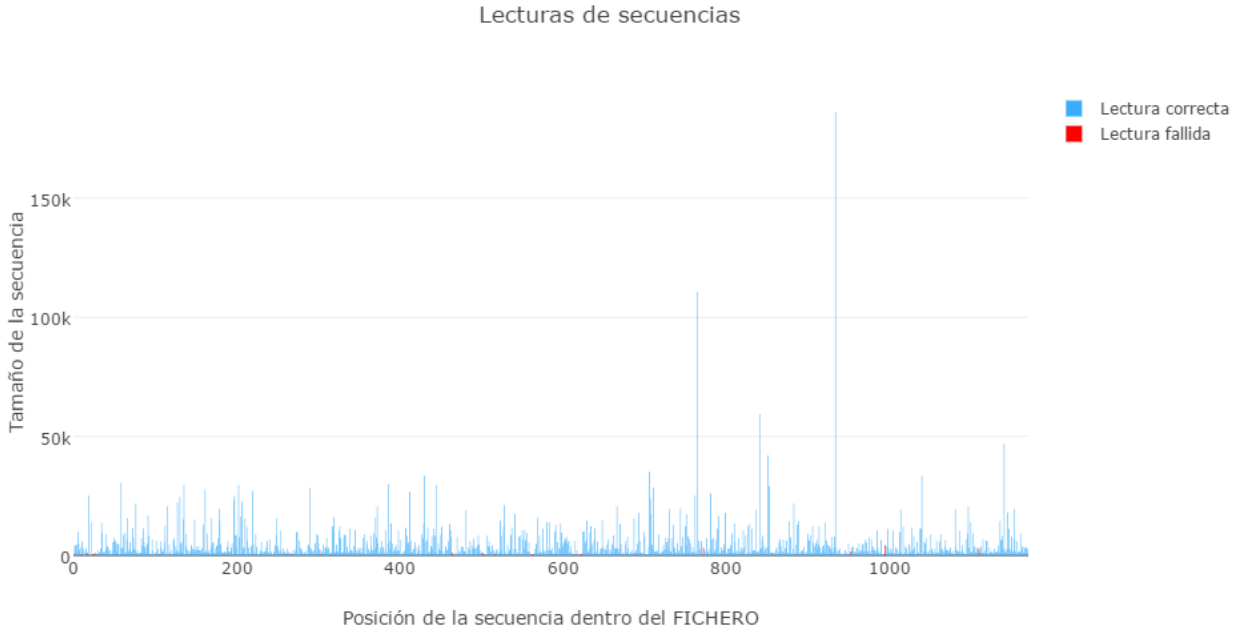


Figura 17. Resultados de aplicar el filtro de las dos letras más repetidas con un umbral al 80%

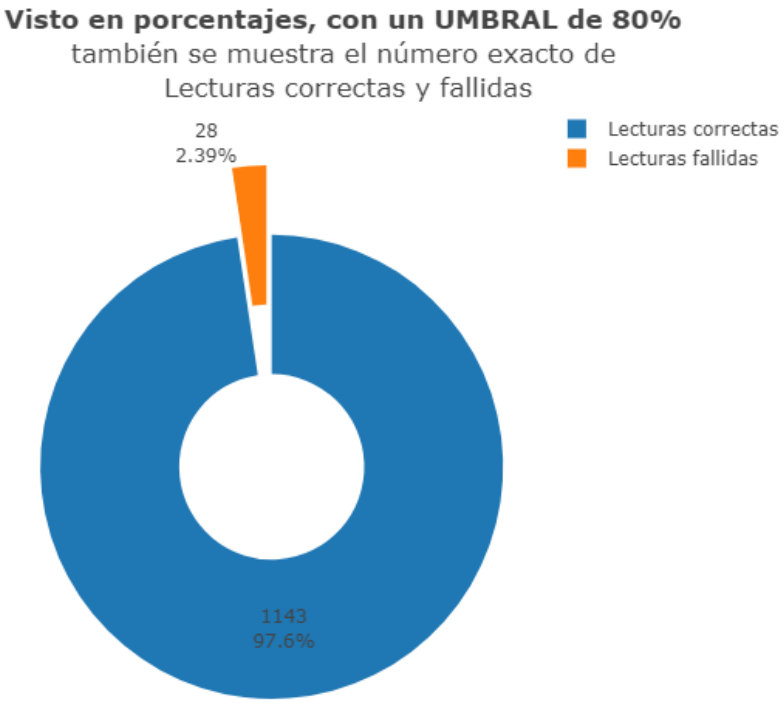


Figura 18. Porcentajes de aplicar el filtro de las dos letras más repetidas con un umbral al 80%

Mientas más alto sea el umbral, el porcentaje de lecturas correctas y fallidas subirán y bajarán respectivamente.

3.2 Resultados por filtro del Contenido GC

Una vez descartado el primer filtro se decidió usar un método propio de la biología, Contenido GC, donde se obtuvieron resultados muy distintos a los que se habían obtenido con anterioridad.

En este filtro el parámetro que se puede configurar es el intervalo de confianza con el cual se deberán hacer los cálculos para fijar los límites superior e inferior. Dicho parámetro se fijó en un 0.95, y luego se realizaron los cálculos adecuados para poder utilizarlo, que dieron lugar al límite de 1.64.

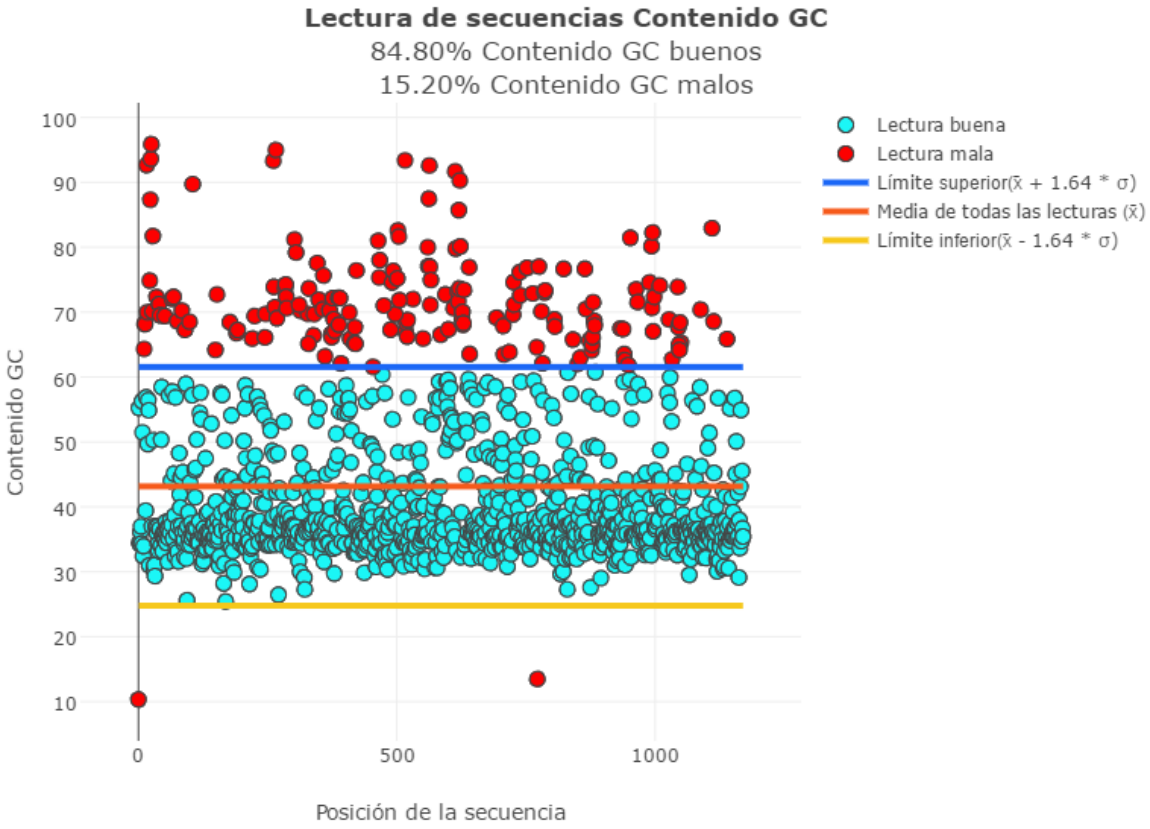


Figura 19. Aplicación del filtro de Contenido GC con un intervalo de confianza de un 0.95

3.3 Resultados por filtro de Calidades.

En el caso del filtrado por calidades, como se mencionó anteriormente, no se toma en cuenta el límite superior, cosa que si se toma en cuenta en el Contenido GC.

Al igual que en el filtro anterior se fijó un intervalo de confianza a un 0.95%, lo cual, como se dijo antes, se convirtió en un 1.64 luego de aplicar los cálculos necesarios.

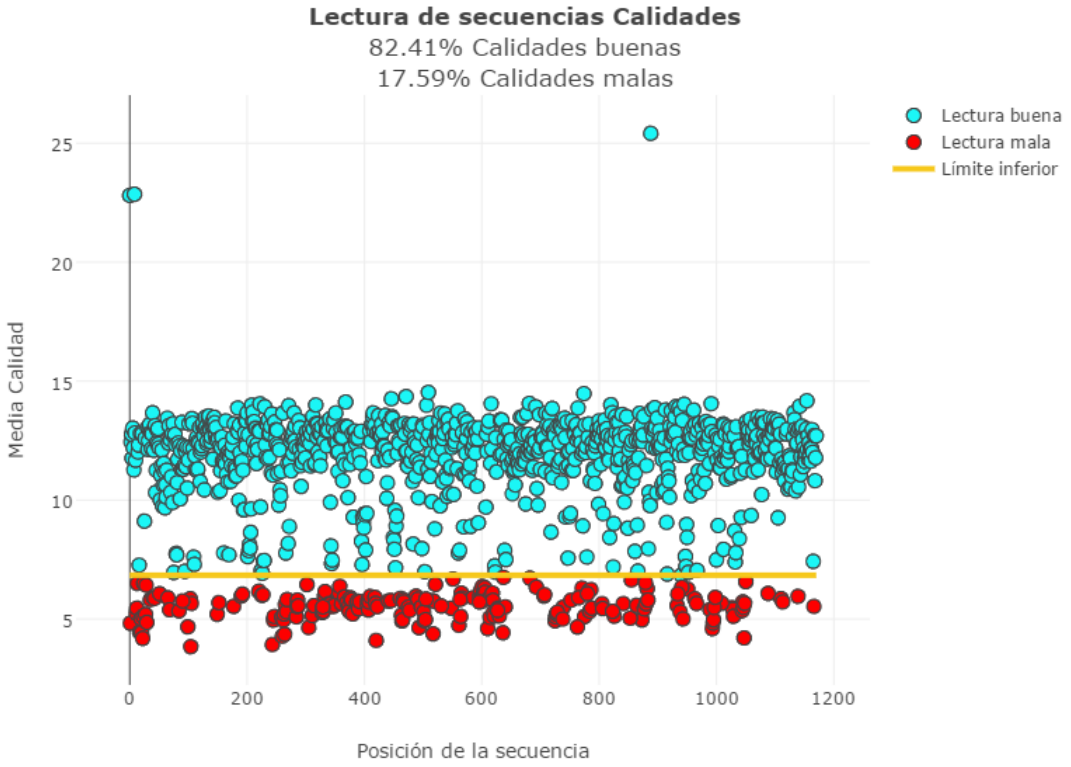


Figura 20. Resultados aplicar el filtro de Contenido GC con un intervalo de confianza de un 0.95

3.4 Comparativa de los filtros de Contenido GC y Calidades.

Para las comparativas, como se mencionó en la Fase 4 del desarrollo, se utilizó la librería Pandas la cual mediante un DataFrame permite tener columnas donde cada celda contiene un valor obtenido de un array o diccionario.

Para este caso, se utilizaron, además, distintos colores para poder dar una manera visualmente rápida de ver las coincidencias y las no coincidencias. Las celdas con color rojo significan que no hay coincidencias entre filtros y las celdas azules significan que sí se encontró coincidencias entre los filtros.

Out[47]:

	Calidades	Contenido GC
0	-	0
1	1	-
2	11	11
3	12	12
4	13	13
5	-	16
6	17	17
7	19	-
8	20	-
9	22	22
10	23	23
11	24	24
12	-	25
13	27	27
14	28	28
15	29	-
16	35	35
17	40	40
...	...	...
224	1110	1110
225	1113	1113
226	1139	1139
227	1166	-

Figura 21. DataFrame de la comparativa entre los distintos filtros

3.5 Resultados de las coincidencias obtenidas de la comparación de secuencias erróneas en los filtros de Contenido GC y Calidades, contrastadas con BLAST.

Inicialmente, como se mencionó en la Fase 5, se intentó utilizar la librería Biopython para contrastar con BLAST si los resultados obtenidos de los distintos filtros eran correctos o no, pero esta no funcionaba correctamente.

Más tarde, en la Fase 8, se decidió acceder con el uso de *web scrapping* y Python a la aplicación web de BLAST. Una vez realizado este proceso se pudieron contrastar los datos con BLAST.

- **Errores detectados que NO encontró coincidencias en BLAST:** 148
- **Errores detectados que SI encontró coincidencias en BLAST:** 8
- **Porcentaje de errores:**94.87 (Porcentaje que **NO** obtuvo evidencia en BLAST)
- **Porcentaje de acierto:**5.13 (Porcentaje que **SI** obtuvo evidencia en BLAST)

Figura 22. Resultados obtenidos de la búsqueda en BLAST

3.6 Resultados Generales

En general, tras aplicar los distintos filtrados, obtuvimos diferencias significativas entre algunos filtros y más similares en otros.

Resultados/Filtros	**Filtro de las dos letras más repetidas**	**Filtro por Contenido GC**	**Filtro por Calidades**
Nº lecturas fallidas	28	178	206
Nº Lecturas buenas	1143	993	965
% Lecturas fallidas	2.39%	15.20%	17.59%
% Lecturas buenas	97.6%	84.80%	82.41%

Tabla 1. Resultados generales tras aplicar los filtros

Además, para buscar la diferencia una vez extraídas las lecturas fallidas, se realizó un experimento donde se calcula el Contenido GC de todo el fichero FASTQ antes y después de aplicar los filtros de Contenido GC y Calidades, de manera que se pueda apreciar si hay diferencia o no una vez aplicados los filtros. Se obtuvieron los siguientes resultados:

Resultados/Filtro	Filtro por Contenido GC	Filtro por Calidades
Contenido GC total antes	44.33%	44.33%
Contenido GC total después	39.47%	39.64%

Tabla 2. Resultados del Contenido GC antes y después de aplicar los filtros

Al final, observando las coincidencias, se obtuvieron los siguientes resultados:

Resultado/filtro	Filtro por Contenido GC	Filtro por Calidades	Coincidencias entre filtros
Lecturas erróneas	178	206	156
Porcentaje	15.20%	17.59%	13.32%

Tabla 3. Resultados de las coincidencias

Capítulo 4.
Conclusiones y líneas futuras

Llevar a cabo este Trabajo de Fin de Grado ha supuesto un reto, se tenía una idea de lo que se hacía en la rama de Bioinformática que, aunque la idea no era errónea, la realidad es que no era más que la punta del *iceberg*.

Dado que este Trabajo de Fin de Grado es un trabajo investigación, se participó en muchas reuniones con los tutores y con el Dr. Carlos flores y con parte de su equipo, por lo que ha sido una experiencia nueva y gratificante que permiten hacerse una idea de lo que puede llegar a ser un entorno de investigación

Por otro lado, es evidente que con el paso años las ciencias y las tecnologías avanzan, unas más rápidas que otras, y en algunos casos el avance es bastante significativo. En la Genética se puede ver el avance de forma muy clara, hace unas décadas para secuenciar ADN las máquinas utilizadas eran bastante caras y grandes, lo que dificultaba su uso en, por ejemplo, zonas afectas por un virus. Hoy en día se puede tener un secuenciador directamente conectado a un teléfono móvil [47], aunque este aun esté en desarrollo. En cuanto a MinION, secuenciador utilizado para obtener los datos con los que se trabajó en este proyecto, sigue con su proyección de futuro. Cada vez hay más profesionales que lo utilizan lo cual hace que tenga una gran comunidad.

De cara a la parte Informática se ha podido, por ejemplo, integrar en sistemas con Big Data, por ello nosotros, los Ingenieros Informáticos, somos de gran utilidad en este campo.

Con todo lo que esto implica, se podrá tener mejor información del estado de salud de las personas, de manera que se podrán, por ejemplo, prevenir enfermedades genéticas o tener más información de enfermedades raras para las cuales en un futuro se tenga tratamiento.

En el Trabajo de Fin de Grado, como se ha explicado a lo largo del documento, se han realizado distintos cuadernos con. Aquí en cada proceso se

le explica al usuario lo que está haciendo, aprovechando las ventajas que nos ofrece Jupyter con sus distintos tipos de celdas, de manera que para utilizarlo no se requiere mayor conocimiento informático que el instalar Jupyter y las librerías que se utilizan en los distintos cuadernos.

Uno de los mayores inconvenientes para llevar a cabo el desarrollo de este Trabajo de Fin de Grado ha sido encontrar información relacionada con las lecturas erróneas utilizando MinION. Teniendo en cuenta esto, desde el punto de vista informático (todo lo relacionado con los procesos que se llevan a cabo dentro del cuaderno), se brinda, al usuario, la mayor información posible de lo que se está haciendo. Además, se muestra gráficamente, para tener mejor percepción visual, el resultado de cada proceso y a modo de resumen algunas estadísticas de lo mostrado.

También se han dotado los cuadernos con funcionalidades para que se puedan exportar en los formatos soportados, FASTA y FASTQ. Todos estos datos se pueden utilizar para, por ejemplo, pasarlo por otro filtro o ensamblarlo.

En resumen, ha sido una experiencia gratificante el poder trabajar con profesionales de otras disciplinas. Tener contacto con el mundo de la investigación, conocer como nos afecta el presente y el futuro de la secuenciación de genoma, ha sido una gran experiencia. Hace darnos cuenta de que todo lo aprendido a lo largo de la carrera sirve para poder adaptarnos a un mundo profesional.

4.1 Líneas futuras

En un futuro, el trabajo desarrollado se podría probar con ADN humano que es mucho más complejo que el de una bacteria, que es el que se utilizó como muestra para el desarrollo de este Trabajo de Fin de Grado.

Por otro lado, se pretende tener los distintos cuadernos en contendores *Docker* para mayor comodidad del usuario.

También se extenderá el cuaderno para ofrecer más formatos de salida, y para poder enlazar con otros proyectos de ensamblado que ya están en funcionamiento, como un paso previo al ensamblaje.

Capítulo 5.
Summary and Conclusions

Carrying out this Final Degree Project has been a challenge, we had an idea of what was being done in the branch of Bioinformatics that, although the idea was not wrong, the reality is that it was only the tip of the iceberg.

Given that this Final Degree project is a research project, I participated in many meetings with the tutors and with Dr. Carlos Flores and part of his team, so it has been a new and rewarding experience that gives us an idea of what a research environment can be like.

On the other hand, over the years science and technology have advanced, some faster than others, and in some cases the progress is quite significant. In Genetics you can see the progress very clearly, a few decades ago to sequence DNA machines used were quite expensive and large, which made it difficult to use them in, for example, areas affected by a virus. Today you can have a sequencer directly connected to a mobile phone [47], even though it is still in development. As for MinION, the sequencer used to obtain the data used in this project, it continues with its future projection. More and more professionals are using it, which makes it a great community. About the Informatics part, for example, it has been possible to integrate Big Data into systems, which is why we, the Computer Engineers, are of great importance in this field.

With all that this implies, it will be possible to have better information on the state of people's health, so that you can, for example, prevent genetic diseases or have more information on rare diseases which will make them treatable in the future.

In the Final Degree Project, as explained throughout the document, different Jupyter notebooks have been produced, in which we take advantage of Jupyter's features. Here in each process the user is explained what he is doing, so to use it does not require more computer knowledge than installing Jupyter and libraries used in the different notebooks.

One of the biggest drawbacks of the development has been finding information related to misreading of MinION. For this reason, the greatest information is exposed to the user of the data which we work with. Showing graphically to have a better visual perception of what has been obtained, as well as statistical data as a summary.

The notebooks have also been equipped with functionalities so that they can be exported in the supported formats, FASTA and FASTQ. All this data can be used to, for example, pass it through another filter or assemble it.

In summary, it has been a rewarding experience to be able to work with professionals from other disciplines. Having contact with the world of research, knowing how the present and future of genome sequencing affects us, has been a great experience. It makes us realize that everything we have learned throughout the course of our careers helps us to adapt to a professional world.

5.1 Future Lines.

In the future, the work developed could be tested with human DNA which is much more complex than that of a bacterium, which is the one used as a sample for the development of this Final Degree Project.

On the other hand, it is intended to have the different notebooks in Docker containers for user comfort.

In addition to adapting it for more output formats, it can be linked to other assembly projects, which are already in production, as a step prior to assembly.

Capítulo 6. Presupuesto

Para el presupuesto total del trabajo se deberá tener en cuenta tanto los recursos humanos, como todos los materiales que se han necesitado a lo largo del desarrollo de la herramienta. Las horas de trabajos incluyendo también todo el proceso de documentación y aprendizaje, puesto que extraer los datos y realizar los gráficos supone tener conocimiento para saber que no se están extrayendo datos erróneos, todo esto tiene una curva de aprendizaje la cual requiere tiempo de trabajo.

Por otro lado, para trabajar con MinION, no solo se necesita conocimientos de genómica para poder trabajar con él, sino que también para ponerlo en funcionamiento. Ello requiere que se preparen unas muestras con todo lo que esto implica, por lo que es necesario tener los materiales concretos con los que se van a trabajar. El Dr. Carlos Flores proporcionó los ficheros con las secuencias obtenidas del MinION que ellos disponen.

6.1 Recursos humanos

6.1.1 Biólogo

Tarea	Horas	Precio	Total
Fase de documentación	70	15 €/h	1.050 €
Preparación de muestras	10	15 €/h	150 €
Poner en funcionamiento el MinION	10	15 €/h	150 €

Tabla 4. Resumen del presupuesto para el Biólogo

6.1.2 Ingeniero Informático

Tarea	Horas	Precio	Total
Fase de documentación	100	20 €/h	2.000 €
Desarrollo y resultados	80	20 €/h	1.600 €
Redacción de la memoria	15	20 €/h	300 €

Tabla 5. Resumen del presupuesto para el Ingeniero Informático

6.2 Costes materiales

Material	Precio
Ordenador portátil Acer V3	900 €
MinION Starter pack Basic	1.000 €
Material de laboratorio	300 €

Tabla 6. Resumen del presupuesto para los materiales

El MinION Startar pack Basic cuenta con los materiales necesarios para hacer las primeras pruebas, en caso de necesitar más material se pueden adquirir en la página oficial.

6.3 Costes totales

Ítem	Coste
Biólogo	1.300 €
Ingeniero informático	3.900 €
Materiales	2.200 €
Total	7.400 €

Tabla 7. Resumen total del presupuesto

Bibliografía

[1] ADN polimerasa. es.wikipedia.org/wiki/ADN_polimerasa

[2] Ácido desoxirribonucleico. es.wikipedia.org/wiki/ADN_polimerasa.

[3] Maxam, A. M., & Gilbert, W. (1977). A new method for sequencing DNA. Proceedings of the National Academy of Sciences of the United States of America, 74(2), 560–564.

[4] Así es el superordenador que descifra el genoma humano. elpais.com/tecnologia/2017/03/30/actualidad/1490875903_152817.html

[5] Accession Number: www.ncbi.nlm.nih.gov/Sequin/acc.html

[6] Bioinformatics and Genomics. www.crg.eu/en/programmes/programmes-groups/bioinformatics-and-genomics

[7] Biopython license. github.com/biopython/biopython/blob/master/LICENSE.rst

[8] BLAST topics. blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=BlastHelp

[9] BLAST: databases. nihlibrary.ors.nih.gov/bioinfo/BLAST/Blast.htm

[10] Beautiful Soup Documentation. www.crummy.com/software/BeautifulSoup/bs4/doc/

[11] Clean Code in Jupyter Notebooks: es.slideshare.net/katenerush/clean-code-in-jupyter-notebooks

[12] Citosina: www.genome.gov/glossarys/index.cfm?id=44

[13] Chemistries R9 for nanopore. nanoporetech.com/about-us/news/update-new-r9-nanopore-faster-more-accurate-sequencing-and-new-ten-minute-preparation

[14] Kellis, M. Prof, & Galagan, J. Prof. (2016). Computacional Biology: Genomes, Networks, Evolution (Ed. rev.). Recuperado de ocw.mit.edu/ans7870/6/6.047/f15/MIT6_047F15_Compiled.pdf

[15] Rocío Bautista Moreno, R. (2010). ¿Cómo funciona? [pdf] Malaga, pp.1-2. Disponible en: www.encuentros.uma.es/encuentros128/comofunciona128.pdf [Accedido 29 Mayo. 2017].

[16] Common URL API: Documentación de la API para acceder a BLAST: ncbi.github.io/blast-cloud/dev/api.html

[17] Documentación scikit-bio: scikit-bio.org/docs/0.5.0/index.html

[18] Documentación Biopy: www.cs.auckland.ac.nz/~yhel002/biopy/index.html

[19] Documentación Biopython: biopython.org/wiki/Biopython

[20] DNA Sequencing with MinION. nanoporetech.com/

[21] Documentation Biopython. biopython.org/

[22] DNA: nanopore sequencing. nanoporetech.com/applications/dna-nanopore-sequencing

[23] GC-Content: en.wikipedia.org/wiki/GC-content

[24] Guanina: www.genome.gov/glossarys/index.cfm?id=96

[25] Herramienta Bioinformática usando Jupyter para el secuenciador de ADN MinION. riull.ull.es/xmlui/handle/915/3089

[26] IBM knowledge Center, que es blast. www.ibm.com/support/knowledgecenter/es/SSEPGG_8.2.0/com.ibm.db2.ii.doc/opt/c0007271.htm

[27] JupyterLab: github.com/jupyterlab/jupyterlab

[28] MinION: el secuenciador de bolsillo. revistageneticamedica.com/2014/10/06/minion-secuenciador-de-bolsillo/

[29] Markdown cells. daringfireball.net/projects/markdown/

[30] Matplotlib: matplotlib.org/

[31] Media recortada: www.uv.es/webgid/Descriptiva/25_otras_medias.html

[32] Matthes, M., Weisbeek, P. J., & Denhardt, D. T. (1982). Mechanism of replication of bacteriophage phi X174 XIX. Initiation of phi X174 viral

strand DNA synthesis at internal sites on the genome. Journal of Virology, 42(1), 301–305.

[33] Wang, Jing & Elizabeth Moore, Nicole & Deng, Yi-Mo & Eccles, David & Hall, Richard. (2015). MinION nanopore sequencing of an influenza genome. Frontiers in Microbiology. 6. 766. 10.3389/fmicb.2015.00766.

[34] Nacional Center for Biotechnology Information. www.ncbi.nlm.nih.gov/

[35] NumPy: www.numpy.org/

[36] Other Sequencing of Mycobacterium bovis: www.ncbi.nlm.nih.gov/sra/SRX2581491[accn]

[37] Pandas: pandas.pydata.org/

[38] PEP8: Style Guide for Python Code: www.python.org/dev/peps/pep-0008/

[39] Plotly. plot.ly/

[40] Project Jupyter. jupyter.org/

[41] Plotly offline: plot.ly/python/offline/

[42] Piperidina. es.wikipedia.org/w/index.php?title=Piperidina&oldid=96722012.

[43] R: Lenguaje de programación: es.wikipedia.org/wiki/R_(lenguaje_de_programaci%C3%B3n)

[44] Sequence Read Archive: www.ncbi.nlm.nih.gov/sra

[45] Sequence Identifiers: a historical note. www.ncbi.nlm.nih.gov/Sitemap/sequenceIDs.html

[46] SMRT Science, SMRT Sequencing. pacb.com/smrt-science/smrt-sequencing/

[47] SmigdgION: Nanopore sensing for use whith mobile divices. nanoporetech.com/products/smidgion

[48] Cock, P. J. A., Fields, C. J., Goto, N., Heuer, M. L., & Rice, P. M. (2010). The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. Nucleic Acids Research, 38(6), 1767–1771. doi.org/10.1093/nar/gkp1137

[49] Van Dijk, E. L., Auger, H., Jaszczyszyn, Y. & Thermes, C. Ten years of next-generation sequencing technology. Trends in Genetics 30, (418-426) (2014).

[50] The Cost of Sequencing a Human Genome. genome.gov/sequencingcosts/

[51] Bourne, P. E., 2004 The future of bioinformatics. In 2nd Asia-Pacific Bionformatics Conference (APBC2004).

[52] Understanding Qualities. maq.sourceforge.net/qual.shtml

[53] URLLib2. docs.python.org/2/library/urllib2.html

[54] Using HTTP Methods for RESTful Services: www.restapitutorial.com/lessons/httpmethods.html

[55] Virus del Zika y ébola: www.theguardian.com/science/2016/feb/03/from-ebola-to-zika-tiny-mobile-lab-gives-real-time-dna-data-on-outbreaks

[56] Web BLAST: blast.ncbi.nlm.nih.gov/Blast.cgi

[57] Web Scraping: sitelabs.es/web-scraping-introduccion-y-herramientas/