UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

***DashQC*: *pipeline* y *dashboard* para el control de calidad de datos de secuenciación de nueva generación**

Trabajo de graduación presentado por Luis Fernando Quezada Bendaña
para optar al grado académico de Licenciado en Ingeniería
Bioinformática

Guatemala,

2022

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

***DashQC*: *pipeline* y *dashboard* para el control de calidad de datos de secuenciación de nueva generación**

Trabajo de graduación presentado por Luis Fernando Quezada Bendaña
para optar al grado académico de Licenciado en Ingeniería
Bioinformática

Guatemala,

2022

Vo.Bo.:

(f) _____

MSc. Luis Augusto Franco López

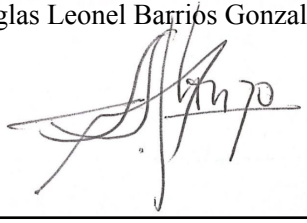
Tribunal Examinador:

(f) _____

MSc. Luis Augusto Franco López

(f) _____

Ing. Douglas Leonel Barrios Gonzalez



(f) _____

Ing. Sergio Rodolfo Alonzo Lemus

Fecha de aprobación: Guatemala, 9 de diciembre de 2022.

Prefacio

El presente proyecto surge de la necesidad de desarrollar herramientas bioinformáticas para trabajar con el *Big Data* biológico. Los avances tecnológicos requieren de una constante innovación y creación de software que automatiza y mejora procesos cotidianos. Este trabajo va mucho más lejos que los objetivos planteados, se ha buscado lograr una contribución significativa para mejorar la investigación científica profesional y acercar, tan solo un poco más, los datos biológicos a la mano humana.

Agradezco a mis padres por haberme dado la oportunidad de estudiar en esta universidad de prestigio y por todo el apoyo y amor incondicional que me han dado a lo largo de estos años para poder convertirme en la persona que soy hoy en día. También quiero agradecer la ayuda recibida, el tiempo invertido y los consejos de mi asesor, Augusto Franco, para hacer de este trabajo un proyecto profesional y realmente aplicable. De tercero, pero no de último, ahora y siempre, doy gracias a Dios por darme la motivación, fuerza, oportunidad y luz durante toda esta travesía universitaria para llegar justo a este momento.

Índice

Lista de figuras

Lista de cuadros

Resumen

Las mejoras tecnológicas y la automatización han aumentado la velocidad y reducido los costos de la secuenciación. Hoy en día, le llamamos secuenciación de nueva generación o, por sus siglas en inglés, *Next generation sequencing* (NGS). Sin embargo, le secuenciación no es perfecta y los resultados van acompañados de un puntaje de calidad para poder determinar su usabilidad. El formato FASTQ (.fastq) es un formato basado en texto para almacenar una secuencia biológica con sus correspondientes puntuaciones de calidad. Existe la necesidad de utilizar herramientas especializadas para poder leer e interpretar la información que contienen estos archivos. La herramienta actual (FastQC) produce gráficos que resultan difíciles de comprender y requieren de un conocimiento más profundo sobre el tema para interpretarlos. Mencionado esto, se introduce el proyecto *DashQC*, donde se establece un *pipeline* interactivo (por medio de un *dashboard*) capaz de realizar pruebas de control de calidad estandarizadas y desplegar los resultados de forma interactiva con distintas herramientas para asistir en su interpretación, a partir de archivos FASTQ. El *dashboard* incluye gráficas interactivas con texto inteligente de interpretación para asistir en la comprensión de los resultados de control de calidad. Asimismo, permite una visualización de resultados de muestras de forma grupal o individual. La herramienta provee una interpretación guiada para cada muestra y permite la descarga de informes automatizados para guardar y compartir el control de calidad realizado.

Abstract

Technological improvements and automation have increased the speed and reduced the costs of sequencing. Today, we call it: next generation sequencing (NGS). However, sequencing is not perfect and the quality scores need to be analyzed to determine its usability. The FASTQ (.fastq) format is a text-based format for storing a biological sequence with its corresponding quality scores. There is a need to use specialized tools to read and interpret the information contained in these files. The current tool (FastQC) generates plots that are difficult to understand and require more in-depth knowledge of the subject to interpret. Having mentioned this, the DashQC project is introduced, where an interactive pipeline is established (by means of a dashboard) capable of performing standardized quality control tests and displaying the results interactively with different tools to assist in the interpretation, based on FASTQ files. The dashboard includes interactive graphs with smart text interpretation to assist in understanding the results. It also allows group or individual visualization of samples. The tool provides guided interpretation for each sample and allows the download of automated reports to save and share the quality control results.

CAPÍTULO 1

Introducción

Secuenciar significa determinar el orden de los cuatro bloques de construcción químicos del ácido desoxirribonucleico (ADN) y ácido ribonucleico (ARN), llamados bases o nucleótidos, que componen una molécula (NHGR, 2022b). La secuencia de ADN, por ejemplo, proporciona toda la información genética de un organismo y puede ser utilizada para estudiar enfermedades genéticas, identificar mutaciones, genes, marcadores moleculares, explorar la filogenética del organismo, cromosomas y evolución, etc. Desde la finalización del Proyecto del Genoma Humano, las mejoras tecnológicas y la automatización han aumentado la velocidad y reducido los costos hasta el punto en que se puede secuenciar de forma rutinaria. Algunos laboratorios pueden secuenciar más de 100,000 millones de bases por año (NHGR, 2022b), y hoy en día un genoma completo se puede secuenciar por debajo de $1000, cuando en un principio tenía un costo de $300 millones (NIH, 2021). A esto llamamos secuenciación de nueva generación (NGS), procesamiento paralelo masivo. Sin embargo, le secuenciación no es perfecta y los resultados van acompañados de un puntaje de calidad.

El formato FASTQ (.fastq) es un formato basado en texto para almacenar la secuencia biológica (secuencia de nucleótidos) con sus correspondientes puntuaciones de calidad. Tanto la letra de secuencia como la puntuación de calidad están codificadas con un solo carácter ASCII para mayor brevedad. Estos archivos pueden ser interpretados y leídos por herramientas como FastQC (Andrews, 2019b), FASTX-Toolkit (Hannon, 2010) y Bowtie (JHU, 2021) para hacer análisis específicos. Los archivos en formato FASTQ de una secuenciación se pueden obtener también de bases de datos como el SRA (Sequence Read Archive) (NCBI, 2022b). SRA es el repositorio público más grande de datos de secuenciación de alto rendimiento y almacena datos de secuenciación sin procesar e información de alineación para mejorar la reproducibilidad y facilitar nuevos descubrimientos a través del análisis de datos (NCBI, 2022b).

Existen distintas herramientas que son capaces de leer estos archivos y procesar la información dentro de ellos para distintos fines. El principal propósito de un archivo FASTQ es almacenar el valor de calidad por base de lectura y por ello, la herramienta de FastQC ha sido una de las herramientas de control de calidad para datos de secuencias de alto rendimiento más populares.

Dado estos antecedentes, se introduce la idea principal de este proyecto, donde se propone, hacer un *pipeline* interactivo. *DashQC* es capaz de recibir como entrada archivos FASTQ, realizar pruebas de control de calidad estandarizadas (FastQC) y desplegar los resultados de forma interactiva con distintas herramientas para asistir en la interpretación de los resultados. Estas herramientas incluyen gráficas interactivas con anotaciones para asistir en la interpretación de cada una de las pruebas de control de calidad, visualización de resultados de muestras de forma grupal o individual y detalles sobre las razones de fallo que podrían suceder. Se utiliza coloración y marcadores intuitivos acoplado a un resumen textual explicando el resultado de la(s) prueba(s) (tanto positivo como negativo). Además, es posible recibir una interpretación generada para cada muestra por cada prueba de control de calidad.

Asimismo, es posible descargar informes automatizados (en PDF, WORD y HTML) de análisis con las interpretaciones y recomendaciones (aplicables en laboratorio y/o en computadora hacia los datos de entrada) ya redactadas. Los informes son generados desde el *dashboard* en una sección designada para reportería lo cual permite al usuario seleccionar en su equipo (computadora o teléfono) el lugar para guardar los reportes (control de versiones) y puede compartir estos reportes con otras personas.

El presente proyecto se ha llevado a cabo con éxito por medio de la creación de un *pipeline*, que automatiza, asiste y presenta el control de calidad de múltiples muestras de datos de secuenciación de nueva generación y representa los resultados con una herramienta gráfica (*dashboard*), de forma visual e interactiva.

CAPÍTULO 2

Justificación

En la actualidad, con la secuenciación de nueva generación, obtener la secuencia biológica de un organismo, gen o proteína se ha vuelto más accesible, en términos financieros, y más rápido, en términos de procesamiento y obtención de resultados. Las bases de datos con información biológica crecen exponencialmente y varias son de libre acceso, lo cual permite que existan datos abiertos a toda la comunidad científica y pueden ser utilizados para análisis en todos los campos de genómica, proteómica, farmacología, bioinformática, etc.

El crecimiento acelerado de los datos de secuenciación ha inspirado a el desarrollo de herramientas para la investigación científica. Sin embargo, muchas herramientas no reciben actualizaciones frecuentemente y con el paso de los años se ha vuelto difícil la interpretación de sus resultados. Requiriendo, de esta forma, que el usuario (ya sea un científico, biólogo, químico, genetista, etc.) conozca con profundidad qué buscar en los resultados, cómo interpretarlos y cuáles son los pasos siguientes por tomar en base a estos. *DashQC*, ahorra el tiempo que toma volverse un experto en interpretación de resultados de control de calidad, mitiga errores de usuario y se convierte en una ventaja en cualquier estudio o análisis.

Por otra parte, gran cantidad del software existente o servicios de control de calidad y depuración de datos de secuenciación tienen altos costos por cada una de las muestras. *DashQC* utiliza software y librerías *open source* para poner a disponibilidad estos servicios sin ningún costo y con libre acceso. Adicionalmente, abre las puertas a el control de calidad de múltiples muestras, reportería automatizada e interactiva y permite su uso en un marco de trabajo mayor.

CAPÍTULO 3

Objetivos

3.1. Objetivo general

Automatizar, asistir y presentar el proceso de control de calidad de datos de secuenciación de alto rendimiento de forma visual, interactiva e intuitiva.

3.2. Objetivos específicos

- Desarrollar un *pipeline* en Bash y R para leer uno o más archivos FASTQ, realizar su control de calidad y representar los resultados de una forma estructurada.
- Desarrollar una herramienta gráfica (*dashboard*) para desplegar de forma interactiva los resultados de control de calidad.
- Guiar al usuario final en la comprensión de las pruebas y resultados de control de calidad e introducirle a una serie de interpretaciones y recomendaciones reales para darle un análisis completo.
- Permitir al usuario generar reportes científicos de forma parametrizada para guardar y compartir, de forma estática, interactiva o editable los resultados de sus análisis.
- Hacer uso de herramientas visuales y diseños de interfaz intuitivos para lograr una interacción usuario-herramienta fluida.

CAPÍTULO 4

Alcance

La secuenciación tiene un gran impacto en campos como la biología molecular, genómica, medicina, farmacología, investigación de enfermedades y otros. En su mayoría, los secuenciadores de hoy en día están limitados a secuenciar segmentos cortos de ADN, por lo que es necesario realizar su ensamblaje mediante software y luego hacer el análisis control de calidad adecuado.

Para ello, *DashQC*, se propone como un software capaz de leer los archivos de calidad de salida de los secuenciadores y permite identificar problemas que se hayan originado, tanto en el secuenciador, como en la información original o de referencia que fue utilizada. *DashQC* provee estadísticas, visualizaciones y resúmenes textuales que adecuan la información en términos simples y comprensibles para lograr que el usuario final pueda tomar la mejor decisión sobre sus datos.

DashQC permite el análisis de n muestras bajo las pruebas de QC estándar establecidas por FastQC y digiere la información para presentarla amigablemente en la interfaz. Esto quita de la ecuación el tiempo que toma a un individuo investigar, volverse conocedor del tema, e interpretar los resultados o identificar los problemas en sus muestras. Ahora bien, *DashQC* no toma decisiones por el usuario, se limita únicamente a proveer toda la información y respaldo estadístico necesario para que el propio usuario pueda discernir entre sus opciones y definir los próximos pasos.

Este es un proyecto que genera un impacto positivo en cualquier proyecto que involucre datos de secuenciación de ADN, especialmente las lecturas generadas por un secuenciador de alto rendimiento. Establece un ambiente íntegro, no solo para navegar y comprender la usabilidad de estos datos, si no también permite establecer una linea base sobre la cual basar un estudio o proyecto de mayor magnitud.

CAPÍTULO 5

Marco teórico

5.1. Bioinformática

5.1.1. Definición formal

La bioinformática es una disciplina científica que consiste en utilizar la tecnología informática para recopilar, almacenar, analizar y difundir datos e información biológica, como las secuencias de ADN, aminoácidos y datos genéticos. Hoy en día, gran cantidad de científicos y médicos alrededor del mundo utilizan bases de datos que organizan e indexan esa información biológica para aumentar nuestra comprensión de la salud y enfermedades existentes (NHGR, 2022a).

5.1.2. Campo de estudio

La bioinformática es un campo de la ciencia computacional que tiene que ver con el análisis de secuencias de moléculas biológicas. Suele referirse a los genes, el ADN, el ARN o las proteínas. Además, la bioinformática es un útil para comparar genes y otras secuencias dentro de un organismo o entre organismos. De igual forma, se utiliza para estudiar las relaciones evolutivas entre organismos y encontrar los patrones que existen en las secuencias para averiguar su función (NHGR, 2022a). Esta disciplina promueve el uso eficaz de los datos biológicos y biomédicos. Almacena, analiza e interpreta el *Big Data* generado por los experimentos de las ciencias de la vida. Este campo multidisciplinar está impulsado por expertos de diversas procedencias: biólogos, informáticos, matemáticos, estadísticos y físicos (SIB, 2018). La bioinformática utiliza técnicas informáticas que se aplican en otros campos también, como la inteligencia artificial, y que incluyen el reconocimiento de patrones, construcción de algoritmos, modelado y la visualización de datos. También es la base actual de la biotecnología gracias a la cual se podrán desarrollar fármacos más eficaces y tratamientos genéticos, lo que convierte a los bioinformáticos en un perfil digital del futuro (Iberdrola, 2020).

5.1.3. Aplicaciones

Existen varias actividades que rodean a la bioinformática, en las cuales esta se puede aplicar. Una actividad fundamental es el análisis de secuencias de ADN y proteínas mediante diversos programas y bases de datos disponibles *online*. Con el aumento exponencial de la información biológica disponible y los avances en la tecnología, cualquier persona con acceso a internet y a los sitios web pertinentes puede ahora descubrir libremente la composición de las moléculas biológicas, como los ácidos nucleicos y las proteínas, utilizando herramientas bioinformáticas básicas. Esto no implica que la manipulación y el análisis de los datos genómicos sea una tarea sencilla. La bioinformática es una disciplina en evolución, y los expertos utilizan complejos programas para analizar, predecir y almacenar datos de secuencias de ADN y proteínas (Bayat, 2002).

Algunas de las aplicaciones principales son: medicina personalizada (adaptando tratamientos a la genética de la persona), descubrimiento de nuevos fármacos, terapia génica (especialmente en las enfermedades causadas por genes individuales que se han visto afectados), agricultura (proteómica, metabolómica) e incluso para el tratamiento de desechos de plástico y eliminación de residuos radiactivos (Iberdrola, 2020). Asimismo, la bioinformática se utiliza para el análisis de la variación y la expresión de los genes, el análisis y la predicción de la estructura y la función de las proteínas y la predicción de las redes de regulación de los genes. De igual forma, se puede aplicar para crear entornos de simulación para la modelización de células completas, la modelización compleja de la dinámica y las redes de regulación de los genes, y la presentación y el análisis de las vías moleculares para comprender las interacciones entre los genes y las enfermedades (Bayat, 2002).

5.2. Secuenciación de nueva generación (NGS)

5.2.1. ¿Qué es NGS?

Secuenciación de nueva generación o secuenciación masiva paralela son términos que describen una tecnología de secuenciación de ADN que ha revolucionado la investigación genómica. Con NGS se puede secuenciar un genoma humano completo en un solo día. En cambio, la anterior tecnología de secuenciación Sanger, utilizada para descifrar el genoma humano, requería más de una década para entregar el borrador final (Behjati y Tarpey, 2013).

En los últimos años, NGS ha crecido bastante, la producción ha aumentado y los costos han bajado. Existen varias plataformas de NGS que utilizan tecnologías de secuenciación para hacer la secuenciación de millones de pequeños fragmentos de ADN en paralelo. Los análisis bioinformáticos se utilizan para unir estos fragmentos mediante el mapeo de las lecturas individuales con un genoma de referencia. La secuenciación de nueva generación puede utilizarse para secuenciar genomas enteros o limitarse a áreas específicas de interés, incluidos genes codificantes (exomas completos) o pequeños números de genes individuales (Behjati y Tarpey, 2013).

5.2.2. Limitaciones de NGS

Las principales limitaciones de NGS en la actualidad son la infraestructura necesaria, la capacidad informática, el almacenamiento y la experiencia del personal necesaria para analizar e interpretar correctamente los datos (Behjati y Tarpey, 2013). El volumen de datos debe gestionarse con habilidad para extraer la información importante en una interfaz clara y entendible. Por muchos avances que se hayan producido en los últimos años, siguen existiendo muchos retos en todo el flujo de trabajo de NGS, desde la preparación de las muestras hasta el análisis de los datos. A medida que se produzcan nuevos avances en las tecnologías subyacentes, surgirán soluciones, pero también nuevos retos.

5.2.3. Datos de secuenciación

Las aplicaciones de NGS se están expandiendo rápidamente, lo que exige métodos eficientes y creativos de almacenamiento, análisis y visualización de datos. El reto principal al que se enfrentan los investigadores es la enorme cantidad de datos que se generan. El archivo BAM (archivo de alineación semi comprimido) para una sola muestra de genoma completo humano es de 90GB. Un proyecto de secuenciación promedio de 100 muestras, por ejemplo, generaría 9TB de archivos BAM (GEN, 2017).

El coste de la secuenciación ha bajado, algunos han llegado a la conclusión de que la secuenciación de muestras para las que hay abundante material será más fácil y posiblemente incluso más barata a futuro. Ahora, cuando se trata de analizar esta gran cantidad de datos, existe una gran variedad de opciones para elegir (herramientas que analizan datos "-omicos"). Los investigadores pueden sentirse fácilmente abrumados cuando tratan de encontrar la mejor opción (GEN, 2017).

El *raw output* de todas las máquinas de secuenciación de nueva generación es el formato BCL. En el formato de archivo BCL cada *base call* se registra a medida que la máquina llama realmente a esa base. A diferencia del formato de archivo FASTQ, en el que el registro de la *base call* se realiza después de la lectura de toda la secuencia (ABM, 2016).

El formato BCL, sin embargo, no es útil para algo más que para la máquina de secuenciación. Si se ejecutan varias muestras en el secuenciador, los datos deben clasificarse para separar las lecturas de muestras diferentes, a este proceso se le llama demultiplexación (ABM, 2016). En la demultiplexación, con la herramienta *bcl2fastq*, los datos BCL se convierten al formato FASTQ, de uso universal. Este programa asignará un nombre a cada lectura, que incluye su ubicación en la celda de flujo, así como el índice especificado, y las llamadas de base que constituyen cada lectura con sus puntuaciones de calidad correspondientes (ABM, 2016).

5.3. FASTA / FASTQ

5.3.1. ¿Qué son y qué información contienen?

La tecnología de secuenciación de Illumina utiliza la generación de clústeres y la química de secuenciación por síntesis (SBS) para secuenciar y almacenar los datos de las *base calls* en archivos BCL. Cuando se completa la secuenciación, las *base calls* en los archivos BCL deben convertirse en datos de calidad-secuencia (conversión de BCL a FASTQ) (Illumina, 2021).

Un archivo FASTQ es un archivo de texto que contiene los datos de la secuencia de los *clusters* para una o varias muestras. Si las muestras no fueron multiplexadas, el paso de demultiplexación no debe hacerse, únicamente la conversión a FASTQ. Los archivos FASTQ pueden contener hasta millones de entradas y pueden tener un tamaño de varios *megabytes* o *gigabytes*, lo que frecuentemente los hace demasiado grandes para abrirlos en un editor de texto normal (Illumina, 2021). Por lo general, no es necesario visualizar los archivos FASTQ, ya que son archivos de salida intermedios que se utilizan como entrada para herramientas que realizan análisis posteriores (por ejemplo, *DashQC*). Si se requiere explorar un archivo FASTQ para solucionar problemas o por inspección, será necesario un editor de texto que pueda manejar archivos muy grandes o a través de la línea de comandos (CLI).

5.3.2. Formato FASTQ

El formato de archivo FASTQ se utiliza universalmente para representar los datos de secuenciación. Este formato consta de cuatro líneas para cada lectura. La primera línea comienza con un carácter "@", seguido del nombre del identificador del secuenciador. La segunda línea es la secuencia de la lectura en sí, la tercera línea es simplemente un símbolo -"(que actúa como espaciador) y la cuarta son las puntuaciones de calidad de las bases de la segunda línea (ABM, 2016).

Figura 1: Composición del formato de archivos FASTQ universal.

(ABM, 2016)

Las puntuaciones de calidad asignadas a cada base se denominan puntuaciones *Phred*, y están vinculadas a la probabilidad de que el secuenciador haya hecho el *base call* incorrecto. Esta puntuación se expresa como una función logarítmica.

$$Q(p) = -10 \log_{10} p \tag{1}$$

donde p es la probabilidad de un hacer un *base call* incorrecto y *Q(x)* es la puntuación de calidad *Phred* resultante.

Puntuación de calidad (Phred)	Precisión del *base call*
10	90 %
20	99 %
30	99.9 %
40	99.99 %

Cuadro 1: Tabla de puntuaciones de calidad ejemplo.

(ABM, 2016)

Las probabilidades son calculadas por el secuenciador determinando la forma del pico de fluorescencia, la resolución y cualquier solapamiento potencial en cada base. Las puntuaciones de calidad tienden a decrecer cerca del final de las lecturas a causa del solapamiento de la fluorescencia, que ocurre debido a la ruptura incompleta del colorante (ABM, 2016). Este es un factor que afecta la calidad en cuanto más larga sea la lectura. Es decir, es común tener lecturas marcadas como de baja calidad al final de las secuencias.

5.3.3. ¿Cómo se obtienen?

La generación de archivos FASTQ es el primer paso de todos los flujos de trabajo de análisis en la secuenciación.

Por ejemplo, *MiSeq Reporter* y *Local Run Manager en MiniSeq* (Illumina, 2021), una vez finalizado las corridas, colocan los archivos FASTQ en carpetas de salida, y de igual forma lo hacen otras herramientas y secuenciadores. En otros casos, las ejecuciones son cargadas en *BaseSpace Sequence Hub* y la generación de archivos FASTQ se produce automáticamente después de que se cargue por completo. Con respecto al equipo de Illumina, 2021, el software de conversión *bcl2fastq* puede utilizarse para generar archivos FASTQ a partir de datos generados por sus secuenciadores.

Sequence Read Archive (SRA)

El SRA de NIH es el almacenamiento principal de datos de secuenciación de alto rendimiento. Incluye a NCBI, el Instituto Europeo de Bioinformática (EBI) y la Base de Datos

de ADN de Japón (DDBJ). Los datos enviados a cualquiera de las tres organizaciones se comparten entre ellas y se archivan en el SRA de forma universal (NCBI, 2022c).

Dentro del SRA existen niveles jerárquicos, los cuales fueron implementados para tener una mejor roganización de los datos guardados en este archivo. Cada ejecución SRA pertenece a un experimento SRA, cada experimento SRA pertenece a una muestra SRA, y cada muestra SRA pertenece a un estudio SRA (RIDOM, 2022). La SRA acepta datos de todo tipo de proyectos de secuenciación, incluidos los estudios que implican a sujetos humanos o sus metagenomas, que pueden contener secuencias humanas.

Para descargar archivos del SRA, se puede utilizar distintas formas, una de ellas es el *Advanced Search Builder*, que permite buscar por números de accesión (individual o lista), utilizar operadores booleanos o navegar los niveles jerárquicos anteriormente mencionados (RIDOM, 2022). De igual forma, es posible la descarga de archivos de datos de secuencias mediante el SRA Toolkit (NCBI, 2022a). Este *tool-kit* permite el uso de herramientas como *fasterq-dump*, *sam-dump* y *prefetch*. Además, NBCI también a puesto a disposición servicios en la nube para hacer su descarga a través de AWS y otras herramientas *online* (RIDOM, 2022).

5.4. Control de calidad (QC)

5.4.1. QC de datos de secuenciación

Las tecnologías de secuenciación de nueva generación (NGS) han ampliado el alcance de la investigación genómica; sin embargo, esta puede verse afectada por una serie de imperfecciones que surgen durante los procesos de preparación de bibliotecas y secuenciación, que pueden afectar negativamente a la calidad de los datos crudos para los análisis posteriores. Estos problemas incluyen los perfiles de error específicos de la plataforma, la desviación de los tamaños óptimos de los fragmentos de la biblioteca, la variación en las proporciones de las secuencias duplicadas introducidas por el sesgo de la amplificación del PCR, la variación sistemática en las puntuaciones de calidad a lo largo de la lectura de la secuencia y los sesgos en la generación de la secuencia impulsados por la composición de las bases (Trivedi y col., 2014; Andrews, 2019b).

El control de calidad de datos de secuenciación de nueva generación basado en la alineación de los datos se vuelve más complicado ya que en muchos casos no se dispone de un genoma bien ensamblado para tomar de referencia. Los secuenciadores han evolucionado para brindar métricas de control de calidad fiables e informativas, eliminando así, la necesidad de una referencia de alta calidad (Trivedi y col., 2014).

5.4.2. Herramientas existentes

Se han publicado varias herramientas de software útiles para analizar problemas de calidad en los datos de secuenciación, incluida la baja calidad de las bases, contaminación con secuencias adaptadoras y sesgos en la composición de las bases. El proceso de control de calidad (QC) se da por evaluación de la calidad de las lecturas crudas mediante métricas

generadas por la plataforma de secuenciación (por ejemplo, puntuaciones de calidad) o calculadas directamente a partir de las lecturas (por ejemplo, composición de bases). Una de las herramientas más populares para la generación de estas métricas de calidad es FastQC (Andrews, 2019b). FastQC y otras herramientas similares son útiles para evaluar la calidad general de un proceso de secuenciación y se utilizan ampliamente en entornos de producción de datos NGS como punto de control de calidad inicial.

El componente *preqc* de SGA (Simpson, 2014) puede predecir las características del genoma y las métricas de control de calidad, incluyendo la longitud de los fragmentos y los niveles de duplicación; sin embargo, estas métricas pueden ser subestimadas masivamente. Otras herramientas, como PRINSEQ (Schmieder y Edwards, 2011) y FASTX-Toolkit (Hannon, 2010), pueden predecir la tasa de duplicación de fragmentos o lecturas sin una referencia, pero tienen importantes limitaciones. Dado que estas técnicas se basan en algoritmos de agrupación de secuencias, las secuencias idénticas, que pueden o no ser duplicadas, pueden ser eliminadas erróneamente. Estos enfoques consumen mucho tiempo y memoria informática, y pueden crear cuellos de botella en un entorno de producción de alto rendimiento en el que es necesario un control de calidad rápido y eficiente de los datos NGS en bruto (Trivedi y col., 2014).

Como fue mencionado anteriormente, otros pasos de control de calidad que se realizan habitualmente implican el mapeo de las lecturas con una referencia conocida para calcular una serie de métricas a partir de los perfiles de alineación. Estos incluyen niveles de duplicación de fragmentos o secuencias, estimaciones de los tamaños de inserción de la biblioteca, etc. (ABM, 2016). Estas métricas se calculan de forma rutinaria cuando se dispone de una referencia bien establecida, pero este no siempre es el caso, sobretodo con genomas nuevos (Trivedi y col., 2014). Por esta razón, cobran importancia los métodos de control de calidad que pueden analizar las métricas generales, puntuaciones de calidad, errores de secuenciación y secuencias sobrerrepresentadas sin un genoma de referencia.

5.4.3. FastQC

Los secuenciadores modernos de alto rendimiento pueden generar millones de secuencias en una sola ejecución. Antes de analizar esta secuencia para sacar conclusiones biológicas, siempre hay que realizar algunas comprobaciones sencillas de control de calidad para asegurarse de que los datos tienen un buen aspecto y de que no hay problemas o sesgos que puedan afectar a su uso. La mayoría de los secuenciadores generan un informe de control de calidad como parte de su proceso de análisis, pero éste suele centrarse únicamente en la identificación de los problemas generados por el propio secuenciador (Andrews, 2019b). FastQC es una herramienta que proporciona un informe de control de calidad que puede detectar problemas originados en el secuenciador o en el material de la biblioteca de partida.

El análisis en FastQC se realiza mediante una serie de módulos de análisis. Cada módulo es una prueba de control de calidad estándar y su resultado se muestran con una marca verde los resultados normales, con un triángulo naranja los ligeramente anormales y con una cruz roja los que son inusuales (Figura 2). A continuación se presente una breve descripción de cada uno de los módulos y su contenido.

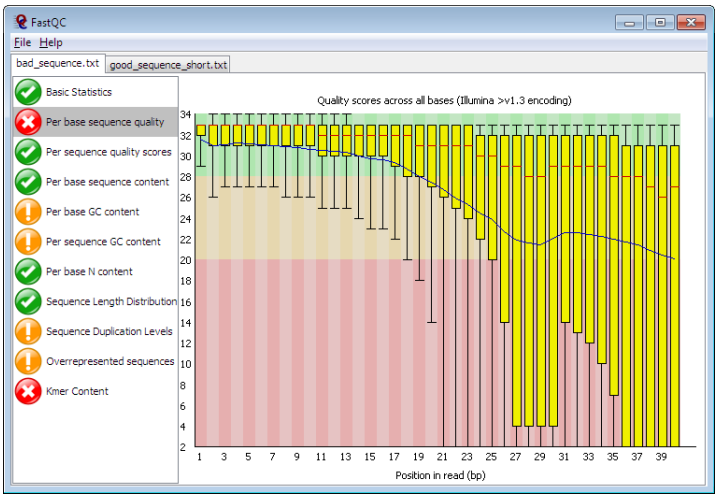


Figura 2: Interfaz gráfica de FastQC.

(Andrews, 2019b)

Basic Statistics

El módulo de *Basic Statistics* (Andrews, 2019b) genera algunas estadísticas de composición simples para el archivo FASTQ analizado. Entre ellas, el nombre del archivo, tipo de archivo, codificación (codificación ASCII de los valores de calidad), el total de secuencias (un recuento del número total de secuencias procesadas), secuencias filtradas (número de secuencias eliminadas), longitud de la secuencia (longitud más corta y más larga) y el %GC (contenido guanina-citosina).

Per base sequence quality

La prueba de *Per base sequence quality* evalúa el rango de valores de calidad en todas las bases en cada posición del archivo FASTQ. Se utiliza un gráfico de tipo caja y bigote donde el eje Y del gráfico muestra las puntuaciones de calidad y el eje X la posición de la lectura, permitiendo comparar la mediana, el rango intercuartilar, máximos, mínimos y la calidad media. Se colorea el fondo del gráfico por zonas de buena calidad (verde), calidad razonable (naranja) y mala calidad (rojo). En general, cuanto más alta sea la puntuación, mejor será la llamada de base. La calidad, en la mayoría de las plataformas, se suele degradar a medida que avanza la corrida, por lo que es común ver que las llamadas de base entren en el área naranja (calidad razonable) hacia el final de una lectura. Esta prueba arroja alerta si el cuartil inferior de cualquier base es inferior a 10 puntos de calidad, o si la mediana de cualquier base es inferior a 25 puntos de calidad. Dará un fallo si el cuartil inferior de cualquier base es inferior a 5 puntos de calidad o si la mediana de cualquier base es inferior a 20 puntos de calidad (Andrews, 2019b).

Per sequence quality scores

El módulo de puntuación de calidad por secuencia permite ver si un subconjunto de secuencias tiene valores de calidad bajos. Si una proporción significativa de las secuencias de una corrida tiene calidad baja, esto podría indicar algún tipo de problema sistemático de la corrida (por ejemplo, problemas en la celda de flujo). Se emite una alerta si la calidad media observada con mayor frecuencia es inferior a 27 puntos de calidad (lo que equivale a una tasa de error del 0.2 %). Se produce una falla si la calidad media observada con más frecuencia es inferior a 20 puntos de calidad (lo que equivale a un porcentaje de error del 1 %) (Andrews, 2019b).

Per base sequence content

La prueba de *Per base sequence content* evalúa la proporción de las cuatro bases/nucleótidos normales del ADN de cada posición de la lectura. Para esto se utiliza un gráfico que visualiza en el eje Y la proporción (%) encontrada para adenina (A), citosina (C), guanina (G) y timina (T). Al utilizar una librería aleatoria se espera que haya poca o ninguna diferencia entre las diferentes bases de una secuencia, por lo que las líneas de este gráfico deberían ser paralelas entre sí, es decir, mostrar proporciones estables y complementarias. No deberían estar enormemente desequilibradas entre sí. Observar fuertes sesgos puede indicar la presencia de una secuencia sobrerrepresentada que está contaminando la librería. Este módulo emite una alerta si la diferencia entre A y T, o G y C es superior al 10 % en cualquier posición. Dará una falla si la diferencia entre A y T, o G y C es superior al 20 % en cualquier posición (Andrews, 2019b).

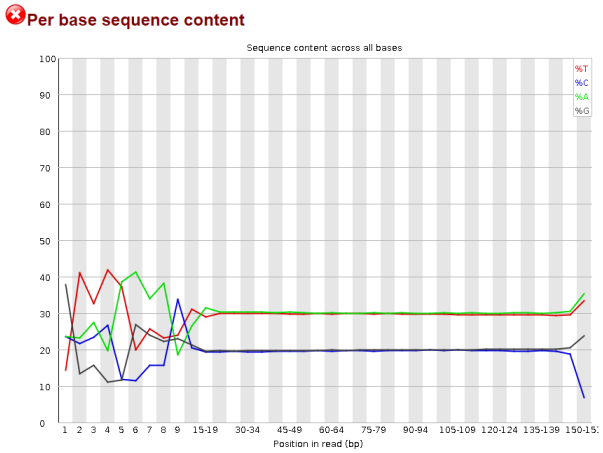


Figura 3: Gráfico de FastQC para el módulo de *Per base sequence content*.

(Andrews, 2019b)

Per sequence GC content

Esta prueba mide el contenido de GC en toda la longitud de cada secuencia de un archivo y lo compara con una distribución normal modelada del contenido de GC. En una librería aleatoria normal se esperaría ver una distribución aproximadamente normal del contenido de GC en la que el pico central corresponde al contenido general de GC del genoma subyacente. Dado a que no se conoce el contenido GC del genoma, el contenido modelo de GC se calcula a partir de los datos observados y se utiliza para construir una distribución de referencia. Una distribución con una forma inusual podría indicar una librería contaminada o algún otro tipo de subconjunto sesgado. El módulo no marcará como un error una distribución normal desplazada. Se produce una alerta si la suma de las desviaciones de la distribución normal representa más del 15 % de las lecturas y se indica un fallo si la suma de las desviaciones de la distribución normal representa más del 30 % de las lecturas (Andrews, 2019b).

Per base N content

Existen ocasiones en las que un secuenciador no puede hacer una llamada de base con suficiente confianza y, en este caso, sustituye la llamada con una N en lugar de una base convencional. Este módulo analiza el porcentaje de llamadas de base en cada posición para la que se ha llamado a una N. Si esta proporción es muy elevada se indica en el análisis que la corrida no ha sido capaz de interpretar los datos lo suficientemente bien como para realizar llamadas de bases válidas. Este módulo emite una alerta si alguna posición muestra un contenido de N superior al 5 % y genera una falla si cualquier posición muestra un contenido de N>20 % (Andrews, 2019b).

Sequence length distribution

Algunos secuenciadores de alto rendimiento generan fragmentos de secuencias de longitud uniforme, pero otros pueden contener lecturas de longitudes muy variables. Varios *pipelines* recortan las secuencias para eliminar llamadas de base de baja calidad. Este módulo analiza la distribución de los tamaños de los fragmentos. En muchos casos, se van a obtener fragmentos de un solo tamaño, pero para los archivos FASTQ de longitud variable esta prueba resaltará aquellas cantidades relativas de cada tamaño diferente de fragmento de secuencia que se encontró y emitirá una alerta si este es el caso. Se emite un error si se encuentra algún fragmento de tamaño 0 (Andrews, 2019b).

Sequence duplication levels

En una librería diversa, la mayoría de las secuencias aparecerán sólo una vez en el conjunto final. Un nivel bajo de duplicación puede indicar un nivel muy alto de cobertura de la secuencia objetivo. Por otro lado, un nivel alto de duplicación indica algún tipo de sesgo (contaminación de la librería, sobre amplificación por PCR, etc.). Este módulo analiza el grado de duplicación de cada secuencia del conjunto y también el número de secuencias por diferente grado de duplicación. Para reducir los requisitos de memoria de esta prueba,

FASTQC analiza las secuencias que aparecen en las primeras 200,000 secuencias, pero esto debería ser suficiente para obtener una estimación buena de los niveles de duplicación en todo el archivo. Cada secuencia se rastrea hasta el final del archivo para obtener un recuento representativo del nivel global de duplicación. Este módulo emitirá una alerta si las secuencias duplicadas representan más del 20 % del total. Dará una falla si las secuencias duplicadas representan más del 50 % del total (Andrews, 2019b).

Adapter Content

El módulo de *Adapter Content* hará un análisis genérico de todos los *Kmers* (subcadenas de longitud k contenidas dentro de una secuencia biológica) en la librería para encontrar aquellos que no tienen una cobertura uniforme. Esto puede encontrar un número de diferentes fuentes de sesgo que puede incluir la presencia de secuencias adaptadoras de lectura que se acumulan en el extremo de sus secuencias. Sin embargo, también puede encontrar que la presencia de cualquier secuencia sobrerrepresentada (como díameros adaptadores). Aunque el análisis *Kmer* puede teóricamente detectar este tipo de contaminación, según Andrews, 2019b, no siempre está claro. Por lo tanto, este módulo analiza e identifica conjuntos de *Kmers* definidos y proporciona una visión de la proporción total.

Esta prueba hace un recuento porcentual acumulativo de la proporción de secuencias adaptadoras en cada posición de la lectura, por lo que los porcentajes que se ven sólo aumentarán a medida que la longitud de la lectura continúe. Este módulo emitirá una alerta si alguna secuencia está presente en más del 5 % de todas las lecturas y emitirá un fallo en la prueba si alguna secuencia está presente en más del 10 % de todas las lecturas. Fallar esta prueba no indica un problema como tal, solamente que es necesario recortar las secuencias con adaptadores antes de proceder a cualquier análisis posterior (Andrews, 2019a).

Overrepresented sequences

Una librería normal de secuenciación de alto rendimiento contendrá un conjunto diverso de secuencias. Encontrar una secuencia sobrerrepresentada en el conjunto significa que es altamente significativa (desde el punto de vista biológico) o indica que la librería está contaminada, o no es tan diversa como se esperaba. Este módulo enumera todas las secuencias que constituyen más del 0.1 % del total. Para conservar la memoria, sólo se analizan las primeras 200,000 secuencias. Por lo tanto, es posible que una secuencia sobrerrepresentada pero que no aparezca al inicio de la lectura por alguna razón no sea detectada por este módulo. Para cada secuencia sobrerrepresentada, esta prueba buscará coincidencias en una base de datos de contaminantes comunes e informará del mejor *match* que se encuentre. Las coincidencias deben tener al menos 20pb de longitud y no tener más de 1 desajuste (Andrews, 2019b).

Encontrar una coincidencia no significa necesariamente que ésta sea la fuente de la contaminación, pero guiar en la dirección correcta. Dado que la detección de duplicación requiere una coincidencia de secuencia exacta en toda la longitud de la secuencia, cualquier lectura de más de 75pb de longitud se trunca a 50pb a efectos de este análisis. El efecto que esto tiene en los resultados de esta prueba es provocando una sub-representación de las secuencias más largas que contengan errores de secuenciación o contenido adaptador (Andrews,

2019a). Este módulo emitirá una alerta si se encuentra que alguna secuencia representa más del 0.1 % del total. Dará una falla si se encuentra alguna secuencia que represente más del 1 % del total (Andrews, 2019b).

5.4.4. FastQC en R

La herramienta FastQC, escrita por Andrews, 2019b, en el *Babraham Institute*, es la más utilizada para realizar el control de calidad de los datos de secuenciación de alto rendimiento. FastQC produce, para cada muestra, un informe HTML y un archivo ZIP, que contiene los archivos *fastqc_data.txt* y *summary.txt*. Ambos de estos archivos contienen los datos utilizados para cada módulo de FastQC, así como los resultados guardados en forma de texto. Por esto, el paquete *fastqcr* (Alboukadel Kassambara y Ahmed, 2018) en el lenguaje de programación R, provee una forma sencilla y documentada para el parseo de datos de los archivos mencionados anteriormente. Además, contiene funciones de ayuda para lectura, automatización, agregación y análisis de resultados de FastQC para un gran número de muestras (Figura 4).

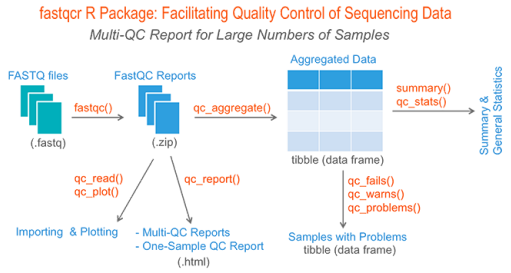


Figura 4: Flujo de información y funciones de agregación con el paquete *fastqcr* en R.

(Alboukadel Kassambara y Ahmed, 2018)

El paquete de *fastqcr* permite instalar y correr FastQC desde R. Esto es de suma importancia para poder incluir el procesamiento de datos estándar de FastQC dentro de un flujo de trabajo más complejo y automatizado. Por medio de la función *fastqc* del paquete, se puede ejecutar múltiples corridas para n muestras dentro de un directorio específico. Además habilita el uso de *multithreading*, para hacer rápido y eficiente el proceso a la hora de correr el control de calidad. La función *qc_aggregate* hace posible agregar varios informes de FastQC en un *data frame*. Luego, este cuadro de datos complejo, puede descomponerse y utilizarse para extrapolación de resultados y datos crudos. *qc_fails* y *qc_warns* se utilizan para filtrar pruebas de control de calidad que encontraron cualquier tipo de problema con los datos junto con los identificadores de las muestras que los presentaron. *qc_stats* es una función que permite adentrarse a los datos de las muestras desde una perspectiva individual y habilita la manipulación de datos crudos por prueba de control da calidad, parseados del *fastqc_data.txt* original de cada muestra. Finalmente, existen otras funciones de gran utilidad dentro de este paquete como *qc_unzip*, *qc_read_collection* y *qc_read* que ayudan con la manipulación, lectura y trabajo de los archivos ZIP producidos al correr FastQC (CRAN, 2019).

5.4.5. Interpretación de QC

El módulo de estadísticas básicas permite la revisión de métricas generales de la muestra. Es recomendado hacer un seguimiento del número total de lecturas secuenciadas para cada muestra y asegurarse de que la longitud de la lectura y el contenido de %GC son los esperados. Las lecturas duplicadas no deben tener un porcentaje elevado, ya que pueden indicar un sesgo, como la sobre-amplificación de la PCR o un nivel muy alto de cobertura de la secuencia objetivo.

Uno de los módulos de análisis más importantes es el de *Per base sequence quality*. Visualizar la distribución de las puntuaciones de calidad en cada posición de la lectura permite identificar si se ha producido algún problema durante la secuenciación y ver si es necesario filtrarla o repetirla. Las puntuaciones de calidad parecen disminuir desde el principio hasta el final de las lecturas. En el caso de las lecturas generadas por la secuenciación de Illumina, esto no es inesperado, y existen causas conocidas para este descenso de la calidad. Para interpretar mejor este gráfico, es útil entender los diferentes perfiles de error de secuenciación. Entre ellos se encuentran: la intensidad y pureza de la señal fluorescente, decadencia de la señal, desfase, sobre-agrupamiento y avería de la instrumentación. Además, cualquier descenso repentino de las puntuaciones de calidad podría indicar un problema severo (Khetani, 2020).

Con respecto a la prueba de *Per sequence quality scores* es esperado que la mayoría de nuestras lecturas tengan una puntuación de calidad media alta sin aumentos repentinos en los valores de calidad baja; lo más importante es que la mayor proporción de datos se encuentren en las categorías de alta calidad. Para la prueba *Per sequence GC content* generalmente es una buena idea observar si el contenido de GC del pico central corresponde al % de GC esperado para el organismo. Además, la distribución debe ser normal, a menos que haya secuencias sobrerrepresentadas (picos agudos en una distribución normal) o contaminación con otro organismo (curva amplia) (Khetani, 2020).

El módulo de *Sequence duplication levels* puede ayudar a identificar una librería de baja complejidad, que podría ser el resultado de demasiados ciclos de amplificación por PCR. Algunas sugerencias para ajustar posibles errores son ajustar el número de ciclos de PCR, la cantidad de entrada o la cantidad de secuenciación para futuras librerías. Por otro lado, la tabla de secuencias sobrerrepresentadas ayuda a identificar la contaminación, como las secuencias de vectores o adaptadores. Por ejemplo, si el contenido de %GC estaba desfasado en el módulo anterior, esta tabla puede ayudar a identificar la fuente (Khetani, 2020).

5.5. Data science y visualización de datos

5.5.1. ¿Qué es data science?

La ciencia de los datos o *data science* es el estudio de los datos para extraer ideas significativas e *insights*. Es un campo multidisciplinar que combina principios y prácticas de los campos estadística y la ingeniería informática para analizar grandes cantidades de datos. Los distintos análisis desarrollados en este campo pueden ayudar a los científicos

de datos a plantear y responder preguntas como qué ha pasado, por qué ha pasado, qué pasará y qué se puede hacer con los resultados (Efron y Hastie, 2021). Este es un campo importante porque combina herramientas, métodos y tecnología para generar significado y nuevos descubrimientos a partir de los datos.

La ciencia de datos se utiliza para estudiar los datos de cuatro maneras principales. La primera manera es el análisis descriptivo, examina los datos para obtener información sobre lo que ha ocurrido o lo que está ocurriendo en el entorno. Se caracteriza por la visualización de los datos, como gráficos, mapas, tablas o narraciones generadas. La segunda, análisis de diagnóstico, es un examen profundo y detallado de los datos para entenderlos. Se caracteriza por técnicas como el *drill-down*, descubrimiento de datos, minería de datos y las correlaciones (todos parte de un análisis exploratorio completo). Se pueden realizar múltiples operaciones y transformaciones de datos en un determinado conjunto para descubrir patrones únicos en cada una de estas técnicas. La tercera, análisis predictivo, utiliza los datos históricos para hacer predicciones precisas sobre los patrones de datos que pueden ocurrir en el futuro. Se caracteriza por técnicas como el aprendizaje automático, la predicción, estudio de patrones y el modelado predictivo. La cuarta, análisis prescriptivo, lleva los datos predictivos al nivel de implementación. Es posible analizar las implicaciones de las diferentes opciones y recomendar el mejor curso de acción. Utiliza en conjunto la visualización de datos, la simulación y el procesamiento de eventos complejos (Efron y Hastie, 2021).

5.5.2. Proceso de la ciencia de los datos

Obtención de datos

Los datos pueden ser preexistentes, adquiridos o pertenecientes a un repositorio de datos de Internet. Los científicos de datos pueden extraer datos de bases de datos internas o externas, registros de un servidor web, redes sociales o adquirirlos de fuentes de terceros (AWS, 2022).

Limpieza de datos

La depuración de datos, es el proceso de normalización de los datos según un formato predeterminado. Incluye la gestión de los datos que faltan, la corrección de los errores de los datos y la eliminación de los valores atípicos de los datos. Algunos ejemplos de depuración de datos son la estandarización de formatos de variables, corrección de errores ortográficos, espacios adicionales o caracteres inválidos y corrección de inexactitudes matemáticas.

Exploración de datos

La exploración de datos es el análisis preliminar que se utiliza para planificar las estrategias de modelado de datos posteriores. Los científicos de datos obtienen una comprensión inicial de los datos utilizando estadísticas descriptivas y herramientas de visualización. Luego, se exploran los datos para identificar patrones interesantes que puedan ser estudiados (AWS, 2022).

Modelación de datos

Se utilizan algoritmos para obtener conocimientos más profundos, predecir resultados, prescribir el mejor curso de acción y generar un modelo. El modelo puede probarse con datos de prueba predeterminados para evaluar la precisión de los resultados y puede ser ajustado iterativamente para mejorar los resultados (AWS, 2022).

Interpretación de datos y resultados

Principalmente se busca convertir los datos en acción. Se elaboran diagramas, gráficos y tablas para representar tendencias y predicciones. La síntesis de datos ayuda con la comprensión de los descubrimientos e implementar los resultados de forma eficaz (AWS, 2022).

5.5.3. Visualización y storytelling

La visualización de datos es la representación de datos mediante el uso de gráficos comunes, como diagramas, gráficos de dispersión, gráficos de pie, barra, infografías e incluso animaciones. Estas representaciones visuales de la información comunican relaciones de datos complejas y conocimientos basados en datos de una manera fácil de entender (Wilke, 2019). La visualización de datos se utiliza habitualmente para estimular la generación de ideas en los equipos. A menudo se aprovechan durante las sesiones de *brainstorming* y de *Design Thinking* al comienzo de un proyecto, ya que ayudan a recopilar diferentes perspectivas. También se utilizan para la construcción de herramientas automatizadas que son utilizadas en un proyecto para la toma de decisiones. El descubrimiento visual y la visualización ayuda a los analistas, los científicos de datos y otros profesionales a identificar patrones y tendencias dentro de un conjunto de datos masivo. La visualización de los datos, sin duda, es un paso fundamental en el proceso de la ciencia de los datos (IBM, 2021).

Con tantas herramientas de visualización de datos disponibles, también ha habido un aumento de la visualización de información ineficaz. La comunicación visual debe ser sencilla y deliberada para garantizar que la visualización de los datos ayude a su público objetivo a llegar a la idea o conclusión deseada. Seguir buenas prácticas de diseño y orden puede ayudar a garantizar que la visualización de datos sea útil y clara; de igual forma, es importante establecer el contexto, conocer al público objetivo, elegir un elemento visual adecuado (gráfico, tabla, forma, color, figura, etc.) y mantenerlo simple (IBM, 2021).

El *storytelling* de datos es la capacidad de comunicar eficazmente los conocimientos de un conjunto de datos mediante narraciones y visualizaciones. Puede utilizarse para contextualizar los datos e inspirar la toma de una acción. El cerebro humano tiene preferencia por las historias en lugar de los datos puros dado que estos son demasiada información para procesarse a simple vista y se necesita simplificar y resumir la información que vale la pena destacar. La narración de datos puede ayudar a convertir los datos en acción, es por eso que sin una comunicación eficaz, datos importantes pueden pasar desapercibidos e incluso ser descartados (HBS, 2021).

5.5.4. R y Rstudio

R es un lenguaje de programación de software libre y provee un entorno para la computación estadística. El lenguaje R es ampliamente utilizado entre los estadísticos y los mineros de datos para desarrollar software estadístico con fines de análisis de datos. La aplicación R se instala en una computadora y utiliza los recursos disponibles para procesar el lenguaje de programación R. RStudio se integra con R como un IDE *open source* para proporcionar más funcionalidades (BYU, 2022). Es una herramienta flexible que ayuda a crear análisis legibles, dar mantenimiento a código, anotar comentarios, generar vistas previas de gráficos (Figura 5), llevar un control de las variables y bases de datos cargadas en el entorno, manejo de librerías y otras funcionalidades.

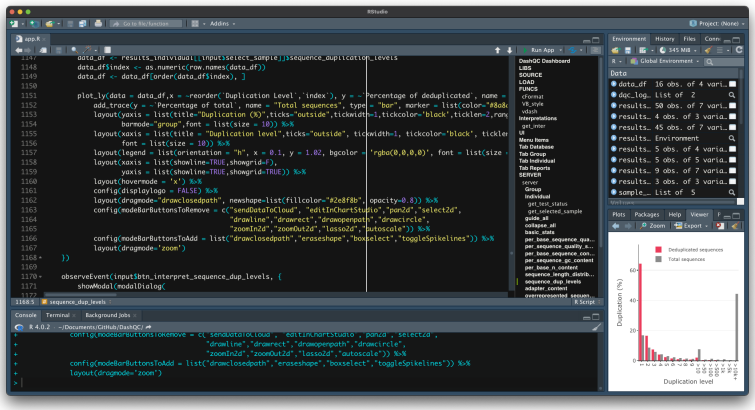


Figura 5: Interfaz gráfica de RStudio con el proyecto *DashQC* cargado.

El uso de RStudio para el análisis de datos y la programación en R proporciona muchas ventajas, entre ellas:

- Interfaz intuitiva para el seguimiento de los objetos, *scripts* y figuras
- Editor de texto con características como la sintaxis codificada por colores
- Funcionalidades de auto-completado y sugerencias de código
- Herramientas para crear documentos que contengan el código, las notas y las imágenes de un proyecto
- Visualización de carpetas de proyecto y ambiente de desarrollo

(Arifin, 2019)

5.5.5. Rmarkdown y LaTeX

RMarkdown permite crear documentos que sirven como un registro de análisis. Son de gran ayuda en la investigación reproducible y funcionan como un *notebook*. RMarkdown permite la presentación de gráficos, tablas, etc. junto a texto convencional para agregar

explicaciones. Esta herramienta utiliza la sintaxis de Markdown, un lenguaje de "marcado" que proporciona métodos para crear documentos con cabeceras, imágenes, enlaces, y otros a partir de archivos de texto plano, manteniendo simple su lectura (Xie y col., 2020). Los documentos generados con RMarkdown se pueden exportar en muchos otros tipos de formatos como HTML, PDF o Word. Cuando se crea un archivo RMarkdown (.Rmd), se utiliza la sintaxis convencional de Markdown junto con trozos de código escritos en R (u otros lenguajes de programación) y luego se evalúa el formato y código para producir un archivo de salida.

LaTeX es un sistema de composición tipográfica de alta calidad, incluye funciones diseñadas para la producción de documentación técnica y científica. RMarkdown y LaTeX, juntos, permiten crear fácilmente documentos HTML y PDF como informes. Por medio de LaTeX es posible incluir notación matemática, ecuaciones, símbolos especiales, operadores, letras del alfabeto griego, funciones matemáticas, matrices o tablas y darle estilos especiales al texto (Mirzaee, 2022).

5.5.6. Shiny dashboards

Dashboards y su utilidad

Un *dashboard* es una presentación visual de datos que se puede utilizar de diferentes maneras, principalmente, para proporcionar información de forma visual. Son un tipo de visualización de datos, y a menudo utilizan herramientas de visualización comunes, como gráficos, diagramas y tablas para mostrar datos de diferentes fuentes. Esto es con el fin de que cualquier persona pueda leerlos e interpretarlos más fácilmente con elementos interactivos. Un *dashboard* suele estar en su propia página y recibe información de una base de datos vinculada. Permiten a todo tipo de profesionales controlar el rendimiento, crear informes y establecer estimaciones y objetivos para la toma de decisiones. Otras ventajas son la representación visual del rendimiento, la capacidad de identificar tendencias, una forma fácil de medir la eficiencia, la visibilidad total de todos los sistemas y la rápida identificación de los valores atípicos o correlaciones de los datos (Vidhya, 2019). Los *dashboards* o tableros son un resumen de conjuntos de datos diferentes presentados de forma que la información relacionada sea más fácil de entender gracias a que son dinámicos, interactivos, muestran datos en tiempo real y ayudan a tener una comprensión *data-driven* (Tableau, 2022b).

Componentes y funcionamiento

Para crear un tablero de la forma más completa, primero se debe definir su público y sus objetivos. Es decir, preguntarse para quién se está construyendo y qué es lo que necesita entender. Luego, hay que elegir los datos que sean relevantes para los objetivos para evitar incluir información de más. También, es necesario comprobar la calidad, completitud y veracidad de los datos. A continuación hay que seleccionar las visualizaciones. Hay muchos tipos diferentes de visualizaciones que se pueden utilizar, como cuadros, gráficos, mapas, etc. Es necesario elegir las que mejor representen los datos. Para iniciar la construcción del tablero se puede utilizar una plantilla y luego construir sobre ella para ir afinando los visuales en base a las metas propuestas. No hay que olvidar en mantener la sencillez, por medio de colores

y estilos similares para que el *dashboard* no se vea desordenado y abrumador. Finalmente, hay que iterar y mejorar sobre periodos de desarrollo tras comentarios y retroalimentación de terceros para mejorar la adopción y la comprensión (Tableau, 2022b).

Shiny dashboards

Debido a la creciente tendencia a presentar análisis a través de *dashboards*, los científicos de datos suelen enfrentarse a problemas a la hora de seleccionar la plataforma para crearlos, ya que las plataformas más destacadas, como Tableau (Tableau, 2022a) y Power Bi (Microsoft, 2022), tienen un coste, no son *open source* y tienen el requisito de que los datos estén en un formato determinado. En este punto sale a la luz la forma más sencilla y de código abierto de crear *dashboards*, es decir, los *Shiny dashboards*.

Es posible crear increíbles tableros interactivos utilizando cualquier tipo de datos importados o creados mediante programación en R. Lo más importante de los *Shiny dashboards* es que son sencillos de crear con la reutilización de las plantillas y el moldeado de los diseños ajustando los códigos y visuales. El primero paso es instalar la librería en R llamada *shinydashboard* y luego hay que dedicar tiempo a comprender las diferentes secciones de un tablero en *Shiny*. El procesamiento de datos y renderizado está controlado por el *server* y en el *UI* se define la estructura y orden de las visualizaciones a incluir. El *UI* está separado en 3 partes principales:

- *Header*: encabezado del tablero con el título.
- *Sidebar*: es un espacio útil para poner filtros, entradas de datos, rangos de fechas, selectores, cajas de texto y campos de búsqueda que luego podrán ser visualizados en el cuerpo del tablero.
- *Body*: incluye la visualización de datos y la creación de gráficos, mapas, tablas, textos de salida, etc. que son la parte principal del tablero.

(Vidhya, 2019)

Toda aplicación de Shiny tiene los dos elementos principales mencionados anteriormente. El primero es la interfaz de usuario (*UI*), que se encarga tanto de los controles del usuario como del diseño y la estructura de los elementos de salida, como los gráficos o las tablas. La interfaz de usuario suele consistir en un esquema de diseño, en el que se especifican los controles del usuario (como *sliders*, selectores, botones, etc.). El segundo elemento es el servidor (*server*), que se encarga de todos los cálculos y la generación de gráficos para el *UI*. El *UI* y *server* están en constante comunicación a la hora de levantar un tablero. Ambos elementos están conectados por objetos *input* y *output*, en donde el *UI* pone los valores de los controles en variables *input*, a los cual el servidor tiene acceso y puede reaccionar y producir variables *output*, las cuales la interfaz muestra. Es una arquitectura bastante simple y puede estructurarse en un solo archivo (*server* y *UI* juntos) o en archivos separados.

Plotly

La librería de gráficos R de *Plotly* hace gráficos interactivos de calidad de publicación. Con esta librería es posible hacer gráficos de líneas, gráficos de dispersión, gráficos de área, gráficos de barras, barras de error, gráficos de caja, histogramas, mapas de calor, subtramas, ejes múltiples y gráficos 3D. Es un paquete de R para crear gráficos interactivos basados en la web a través de la biblioteca de gráficos JavaScript *plotly.js*. Los gráficos creados con este paquete se renderizan localmente a través del marco de trabajo de *htmlwidgets* (Plotly, 2022) y bien pueden ser utilizadas tanto en informes RMarkdown como en tableros interactivos.

5.6. Pipelines

5.6.1. ¿Qué es un pipeline?

Las tecnologías modernas en torno a la bioinformática están cambiando de tal forma que los científicos crean cantidades masivas de datos cada vez que hacen un experimento genómico. La secuenciación de nueva generación ha aumentado la demanda de requisitos de procesamiento de alta gama. Por lo tanto, a medida que la bioinformática se vuelve más compleja, los sistemas deben añadir más etiquetas, parámetros, metadatos y mejores capacidades de análisis para grandes volúmenes de datos. Aquí es donde entra en juego un *pipeline* bioinformático. Un *pipeline* es una serie de elementos de procesamiento, almacenamiento y cálculo conectados en serie que crean una tubería por la que viajan los datos. La salida de un proceso proporciona una entrada directa para el siguiente, y así sucesivamente. Esto es importante por que los científicos de datos crean modelos más complejos a partir de la información genómica, lo que hace necesario contar con una forma de mapear y controlar el análisis paso a paso. Esto ayuda a agilizar la gestión de la información y a minimizar los errores derivados del procesamiento manual (Mazzu, 2022).

5.6.2. Implementación de un pipeline

Los laboratorios que realizan ensayos basados en datos de secuenciación de nueva generación tienen como opción de implementación uno o más *pipelines* bioinformáticos, ya sean desarrollados por el laboratorio, proporcionados por la plataforma de secuenciación o por un tercero. A la hora de realizar la implementación hay que tomar en cuanta varios pasos para el funcionamiento adecuado, siendo el primero de ellos, la validación. El requisito más importante para implementar un *pipeline* es una validación adecuada y sistemática en el contexto de los datos de secuenciación de nueva generación que se estén produciendo y/o utilizando. Luego es necesario llevar un control de versiones ordenado y documentado. Se puede aplicar el control de versiones utilizando marcos de software como *git*, *mercurial* y otros (Roy, 2020). Estas herramientas permiten no sólo la gestión sistemática del código fuente del *pipeline*, sino también el desarrollo colaborativo por parte de un equipo de bioinformáticos e ingenieros de software. Cada despliegue, incluyendo una actualización del *pipeline* de producción, debe ser versionado. Asimismo, se debe documentar las versiones de los componentes individuales. Si se desarrolla y gestiona uno o más componentes en el

pipeline, estos deberán seguir los mismos principios de control de versiones que el resto de componentes del flujo de trabajo.

Una vez realizada la revisión y documentación de cada componente del *pipeline*, las dependencias de los datos y las restricciones de entrada/salida con la ayuda de un profesional o equipo de bioinformática es necesario desarrollar mecanismos para alertar sobre errores inesperados. Este sistema de alertas y de QA aumentan la confianza y fiabilidad de un *pipeline*. De igual forma, siempre es útil llevar un sistema de *logs*, ya que cuando una de estas alertas encuentre algún mal funcionamiento, será más sencillo identificar el punto de falla por medio de una bitácora de procesos.

Existen varias prácticas que se pueden implementar independientemente del marco de trabajo para elaborar una *pipeline* robusta. Aparte de la validación, control de versiones y sistema de alertas mencionados anteriormente, existen otras buenas practicas que se pueden aplicar a la construcción de un *pipeline*, como, la reutilización de los cálculos, ya que la fuerza de cualquier sistema de procesamiento paralelo es su capacidad para procesar datos a través de cálculos idénticos. Mientras más se puedan utilizar esos cálculos exactos, mejor será el resultado deseado. Otra de estas practicas es la utilización de herramientas de modelado. Las herramientas como el software de modelado con lenguaje de descripción de flujos de trabajo pueden ser de gran ayuda para trazar los procesos y las dependencias de los componentes dentro de un *pipeline*. Esta capacidad puede, a su vez, apoyar la optimización de los pasos del flujo de datos.

CAPÍTULO 6

Metodología

6.1. Bases del proyecto

6.1.1. Exploración de ideas

Como paso preliminar se lleva a cabo una lluvia de ideas alrededor del tema principal del proyecto, el mejoramiento del control de calidad de datos de secuenciación. Para resumir esto, se elabora un mapa mental a manera de enlazar los conceptos claves, como bioinformática, visualización de datos, secuenciación de nueva generación, *data science* y *dashboards*.

6.1.2. Estado del arte

Se realiza una investigación de antecedentes y del estado de avances en el tema de QC con respecto a la actualidad a fin de conocer cuáles son las herramientas actuales, que función tienen y cómo, si es posible, se pueden mejorar y/o usar para construir un sistema innovado. De igual forma se hace una selección de publicaciones científicas relacionadas a herramientas de control de calidad de datos de secuenciación con el objetivo de identificar cuáles son las más utilizadas, por qué y cuáles son las opiniones profesionales al respecto de ellas.

6.1.3. Delimitación del problema

Una vez adquiridos todos los conocimientos y antecedentes del dominio del tema, se formula la definición formal del problema, siendo este la necesidad de automatización y asistencia en los resultados de control de calidad de datos de secuenciación para facilitar su interpretación, eliminar la necesidad de dedicar tiempo de estudio al significado de las

pruebas estadísticas y cómo estas funcionan y enfocar los recursos en el entendimiento, análisis y discusión de los resultados.

6.1.4. Delimitación del proyecto

Habiendo mencionado lo anterior, se realiza la delimitación de los componentes del proyecto, que al colocar en conjunto crean la herramienta final propuesta: *DashQC*. Este paso incluye la definición de los objetivos generales y específicos luego de haber realizado todos los pasos de observación e investigación anteriores.

6.2. Control de calidad

6.2.1. FASTQs de estudio

Previo a cualquier actividad de programación, es necesario recolectar la información con la que se estarán haciendo las pruebas y la validación del funcionamiento del proyecto. Se realiza la recolección de 3 estudios de caso (datos de secuenciación biológica) en formato FASTQ (un rango de 5-10 archivos por caso) provenientes del SRA (NCBI, 2022b). Su elección será basada en interés del autor, pero si se buscará obtener conjuntos de muestras distintos para tomar en cuanta estas variaciones en el *pipeline*.

6.2.2. Componente de lectura y QC

Se inicia la programación de la herramienta en R por medio de librerías como *readr*, *fastqcr* y *tidyverse* a manera de elaborar un *script* para la lectura de archivos FASTQ, procesarlos con FastQC para las pruebas de control de calidad y, posteriormente, realizar el manejo de datos (*data wrangling*) para guardarlos de forma agregada en una base de datos personalizada del *pipeline*.

6.2.3. Almacenamiento QC

Una vez desarrollado el componente de lectura y control de calidad se diseña una estructura de datos (en formato .RData) donde es posible guardar los resultados de un conjunto de muestras o muestra individual. Esta estructura es universal para *DashQC* con el fin de guardar resultados ya procesados y permitir que los tiempos de carga sean cortos y la memoria requerida para realizar su lectura sea pequeña. En esta base de datos se debe almacenar también otras estadísticas recopiladas en el control de calidad, como lo son los resultados generales, el resumen de resultados por prueba y por muestras y el resumen de alertas o fallos encontrados.

6.2.4. *Logging* de procesos

Conforme se da la implementación de la sección del *pipeline* encargada de procesamiento de los archivos de entrada y generación del .Rdata de almacenamiento, se implementa un *logger* para ir documentando en un *logfile* los chequeos por módulo. Esto es con el propósito de ir registrando la correcta ejecución del flujo de trabajo y llevar un control de la calidad del dato. Para elaborar este *logger* se utiliza la librería en R de *log4r*.

6.3. *Dashboard*

6.3.1. Diseño de *dashboard*

Previo a la implementación del *dashboard* se elabora un borrador y bosquejo para decidir el orden, presentación, esquema de colores y diseño general del *dashboard*. En esta fase se toma en cuenta factores como visualización de datos, *storytelling*, interpretación de datos y el diseño de una interfaz de usuario amigable e intuitiva.

6.3.2. Implementación de *dashboard*

Con base en el diseño, se elabora el *dashboard* en R con las librerías de *shiny* y *shinydashboard*. La idea es hacer que las gráficas puedan ser visualizadas en diferentes paneles (por prueba estadística) y que estás no sean estáticas, es decir, ofrecer interactividad. Para esto se utiliza las librerías de *Plotly* y *ggplot2*, además de *javascript* y *CSS* para funcionalidades agregadas. Como mencionado anteriormente, las gráficas también incluyen anotaciones y/o indicadores de color para ayudar con la interpretación de estas.

Se da inicio con el desarrollo de la sección de resultados generales. En esta sección se utilizan los datos recopilados del control de calidad de forma grupal. Se diseñan 4 paneles principales, estos deben permitir la visualización de las estadísticas generales, resultados y problemas encontrados de cada muestra, así como, los resultados por cada prueba estándar de control de calidad.

Luego, se desarrolla la sección de resultados individuales. En esta sección se debe utilizar un selector de muestra para indicarle a la página que datos debe mostrar. Esta sección muestra en recuadros informativos el nombre de la muestra seleccionada, sus resultados individuales y sus estadísticas básicas. Debajo de esta información, se desarrollan paneles para mostrar visualmente los gráficos individuales correspondientes a cada una de las pruebas de control de calidad realizadas. Estos paneles deben ser reactivos, interactivos y deben mostrar tanto el resultado obtenido en dicha prueba como un botón que permita acceder a una explicación de los resultados.

Para la generación de la explicación/interpretación de los resultados por prueba se desarrolla un componente basado en la documentación y literatura de FastQC. Este componente se desarrolla de tal forma que reciba el nombre de la prueba, el resultado obtenido y los datos de la muestra y en base a esto pueda compilar un texto interpretativo. El modelo de

diseño para este componente es similar al de un árbol de decisión; contemplando todos los escenarios.

En el *dashboard* se debe desarrollar una sección de descarga de informes, en distintos formatos (PDF, HTML, WORD), donde se permita su generación y descarga automática. En la generación de informes, se habilitan las opciones a elegir entre un informe general, donde se abarquen todas las muestras procesadas, o un informe individual, donde se el habilita al usuario la opción de elegir la muestra sobre la cual desea obtener el informe. En este paso, se debe desarrollar únicamente la interfaz de la página donde se podrá hacer la descarga de dichos informes, puesto que estos no han sido creados aún. Posterior al paso de la creación de los informes, estos serán conectados a esta interfaz para permitir su generación desde el *dashboard*.

6.3.3. Despliegue de *dashboard*

El *dashboard* se visualiza localmente por medio de un enlace local que provee el *pipeline* en cualquier navegador de preferencia del usuario (Google Chrome, Safari, etc.). Se realizan estas pruebas con usuarios externos por medio del portal de *Shinyapps*, donde, a través de un *link*, se les puede compartir la visualización de algunos resultados muestra y recibir su retroalimentación.

6.4. Informes automatizados

6.4.1. Definición de componentes de informes

Esta etapa es de gran importancia, ya que es donde se hace el diseño de los reportes de *DashQC*. En esta etapa, se realiza un bosquejo del orden y secciones que va a llevar tanto el informe general como el informe individual de una muestra. Tomar en cuenta que se busca un diseño de lo general a lo específico, por lo que ambos informes deben iniciar con una introducción a los resultados y estadísticas globales y luego expandir en secciones más detalladas. Dado que en el *dashboard* ya se cuenta con el diseño apropiado de gráficos y textos interpretativos, se debe tomar en cuenta este contenido como el material principal dentro de los reportes.

6.4.2. Implementación de informes

Se utiliza código en R con la librería de *Rmarkdown*, *Knitr*, LaTeX y paquetes como *Plotly* y *ggplot* para crear informes automatizados en formato PDF/WORD (estático/editable) y HTML (interactivo) con el diseño definido en el paso anterior. Además, se iniciarán los preparativos para su incorporación al *dashboard*. Este paso requiere de buena organización de datos en el ambiente local, pues, los informes se generan desde el *dashboard* y deben tomar en cuenta la muestra seleccionada como parámetro para los gráficos, interpretaciones y resultados, sí se da el caso de la generación de un informe para una muestra individual.

6.5. El *pipeline*

6.5.1. *Script* de preparación de equipo y sistema

Para este punto, ya se tiene desarrollado tanto el componente de QC, como el componente visual, por lo que ya se tiene una visión clara de los requerimientos finales del sistema para poder ejecutar estos pasos adecuadamente. Para asegurar que no ocurran problemas de requisitos de sistema, se desarrolla el componente de *setup* del *pipeline*. Este es un *script* encargado de revisar y hacer la instalación de las librerías en R que se utilizan, así como la instalación de *fastqc* en R para poder correr el QC.

6.5.2. Implementación del *pipeline* final

Teniendo los componentes de *setup*, preprocesamiento y *dashboard* creados, se desarrolla en Bash un flujo de trabajo que permite correr estos componentes en serie. Siendo la salida de uno, la entrada de datos del siguiente. Además, se implementa al inicio de este flujo de trabajo una revisión de la existencia del directorio de FASTQC. En caso de existir, se debe limpiar el directorio, y en el caso de que aún no exista, lo debe crear. Luego de consolidar este *pipeline*, se debe agregar el encabezado adecuado al archivo para volverlo un *Unix Executable File* y permitir su funcionamiento desde la línea de comando.

6.6. Pruebas de usuario

6.6.1. *Testing*

Por medio de las herramientas de manejo de acceso que provee *Shinyapps*, se proveerá con un enlace a aquellos usuarios de prueba que podrán acceder al *dashboard* para probar su funcionamiento de forma virtual y se le presentará el *pipeline* localmente a aquellos usuarios que pueden hacer las pruebas de forma presencial. La finalidad de esta fase es revisar posibles *bugs* o componentes del *dashboard* que no están funcionando de la forma esperada previo a su versión final. Para estas pruebas de usuario se solicitará a los *testers* compartir su retroalimentación para los distintos módulos y comentarios. Todo esto es anotado en una rubrica indicando con puntuaciones de 1 a 10 la utilidad que el usuario encontró en dicho módulo. También se anotarán observaciones, dificultades que se les presentaron e ideas de mejoras que le harían al proyecto.

CAPÍTULO 7

Resultados

7.1. Pipeline *DashQC*

El *pipeline* de *DashQC* es un conjunto de procesos que corren en serie, desarrollado en Bash y en R, automatizando la corrida de control de calidad de múltiples muestras y trasladando los resultados a una interfaz gráfica final para su análisis, exploración e interpretación asistida (Figura 6). El inicio del flujo de procesos se da con la preparación de los directorios de trabajo y las especificación de ambiente de R, es decir, la instalación de librerías necesarias para su funcionamiento adecuado (Setup.R). Luego se realiza las configuraciones del *logger* y se prepara el *path* para el directorio de FastQC. A continuación, el *pipeline* ejecuta *fastqc* desde R (Figura 18) y coloca los resultados para cada muestra en dicho directorio. Después, ocurre la lectura, procesamiento y agregación de los datos resultantes para crear una estructura de datos conjunta (DQC.RData). Este archivo actúa como una base de datos que permite desplegar el *dashboard* funcional. Cabe recalcar que, mientras ocurre esta serie de procesos, el *logger* documenta los pasos del *pipeline* con un *timestamp* e indicadores de logro, es decir, chequeos del correcto funcionamiento de los pasos del *pipeline*. Posterior a la creación del archivo RData se cierra el *logger* y se limpia el ambiente de R. Una vez finalizado el preprocesamiento (Preprocess.R), se levanta el *dashboard* para permitir al usuario interactuar con los resultados, representar los resultados de una forma estructurada y optimizar la toma decisiones por medio de un vínculo en el equipo local (Figura 19).

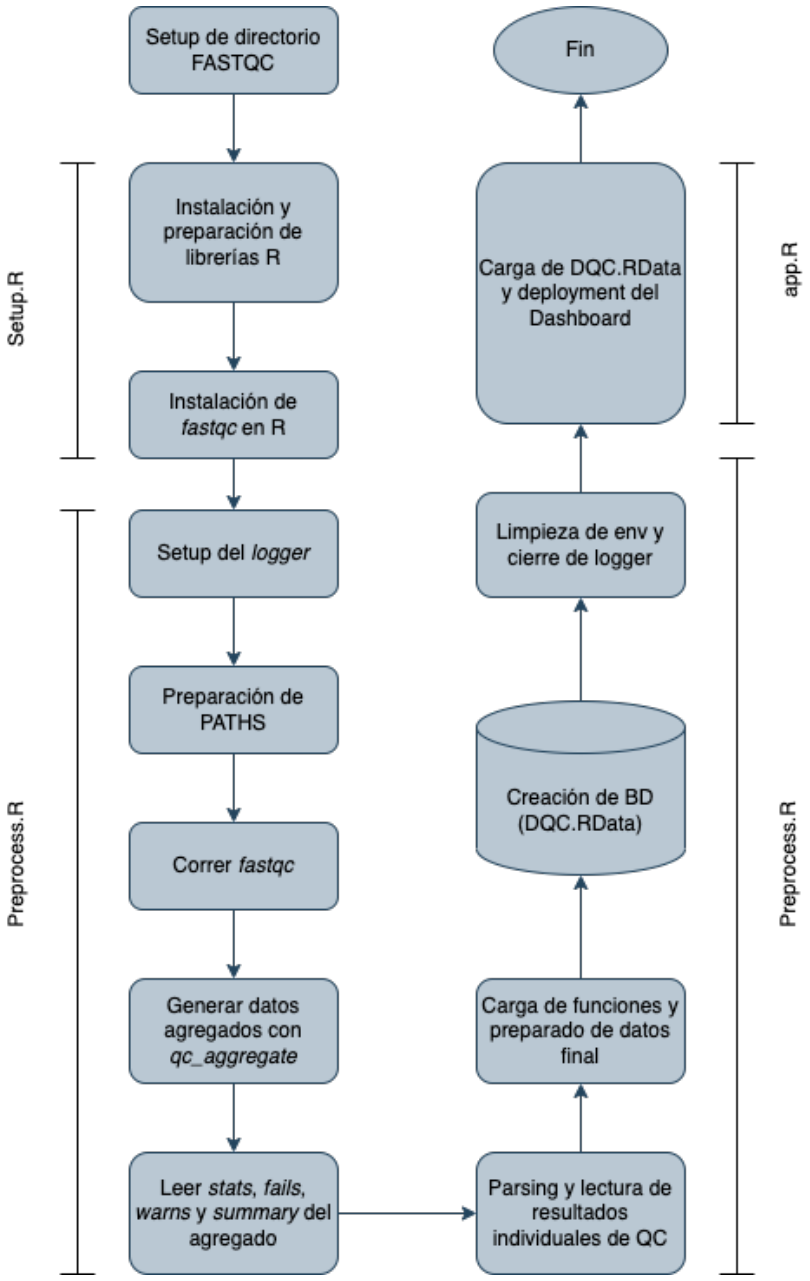


Figura 6: Flujo de procesos en el *pipeline* de *DashQC*.

7.2. De FASTQ a Rdata

Se desarrolló un componente en el lenguaje de programación R para realizar el preprocesamiento de los datos a partir de uno o múltiples archivos FASTQ. Uno de los *scripts* se encarga de instalar y preparar el equipo con los paquetes y librerías necesarias para ejecutarse. El segundo *script* realiza las pruebas de control de calidad estándar de FastQC de forma paralela (con *multithreading*) con éxito de los n archivos FASTQ de entrada.

Posterior a realizar las pruebas de QC, el componente realiza correctamente la agregación de datos y preparación para generar el archivo consolidado DQC.Rdata. El *piepeline* hace uso adecuado del *logger* para generar un *logfile* (Figura 7) con marcas de tiempo (*timestamps*) e indicadores de logro por cada paso ejecutado. Esta sección del *pipeline* es capaz de realizar lectura, procesamiento y preparación de los datos de control de calidad de las muestras del usuario efectivamente.

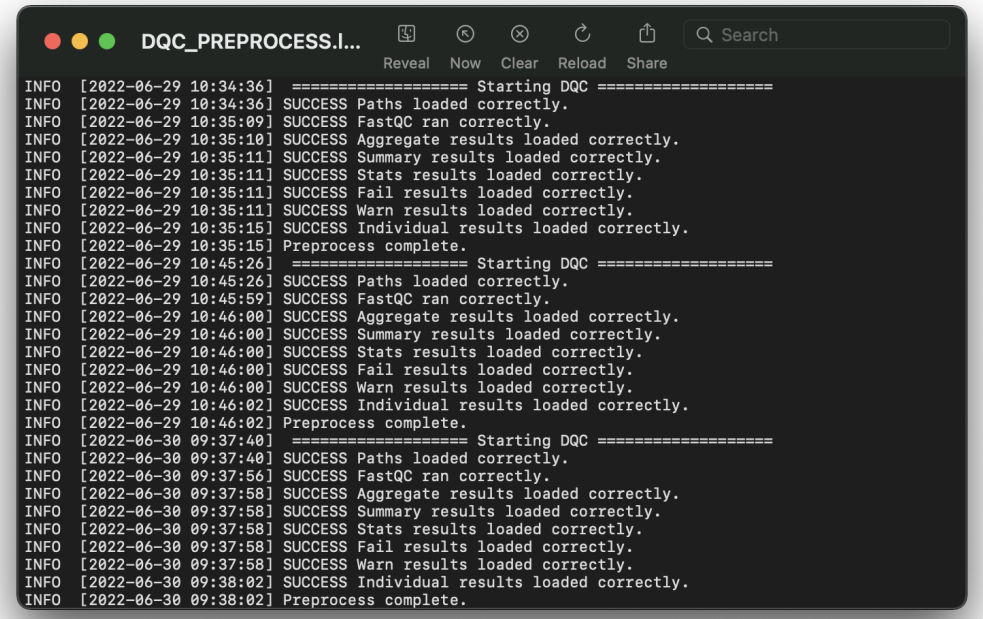


Figura 7: *Logfile* de preprocesamiento.

7.3. *Dashboard DashQC*

Utilizando las tecnologías de *shiny* en R, fue posible elaborar un tablero dinámico dedicado a la visualización automatizada y guiada de todos los resultados preparados por el componente de preprocesamiento. Este tablero es capaz de trabajar con n muestras del usuario de forma eficiente y aporta la posibilidad de ver resultados de forma grupal, como se puede observar en la Figura 8. Esta sección del tablero tiene 3 cajas principales, donde se muestran los conteos de pruebas aprobadas, alertadas o falladas. Luego tiene una sección con 4 módulos, el primero es el gráfico de estadísticas generales por muestra, el segundo el gráfico de resultados de las muestras por prueba, el tercero el gráfico de resumen de resultados por muestra y el último, la tabla de resumen de resultados por prueba.

También aporta la posibilidad de ver resultados de forma individual, por muestra, como se puede observar en la Figura 9. Esta sección cuenta con un selector de muestra en el *sidebar* y los botones de *test guide* y de *collapse tests*. En el *body* del *dashboard* hay 1 recuadro indicando el nombre de la muestra seleccionada y luego tiene 3 cajas principales, donde se muestran los conteos de pruebas aprobadas, alertadas o falladas para dicha muestra. El resto de la sección individual tiene 10 módulos, uno por cada prueba de control de calidad estándar de FastQC. Cada uno de estos módulos, a excepción del de estadísticas básicas, cuenta con un ícono indicador de aprobación, un botón de interpretación, el gráfico o tabla asociada y, en algunos casos, un botón de información. Estas herramientas exitosamente logran desplegar de forma interactiva los resultados de control de calidad y guían al usuario final en la comprensión de las pruebas e introducirle a una serie de interpretaciones y recomendaciones reales.

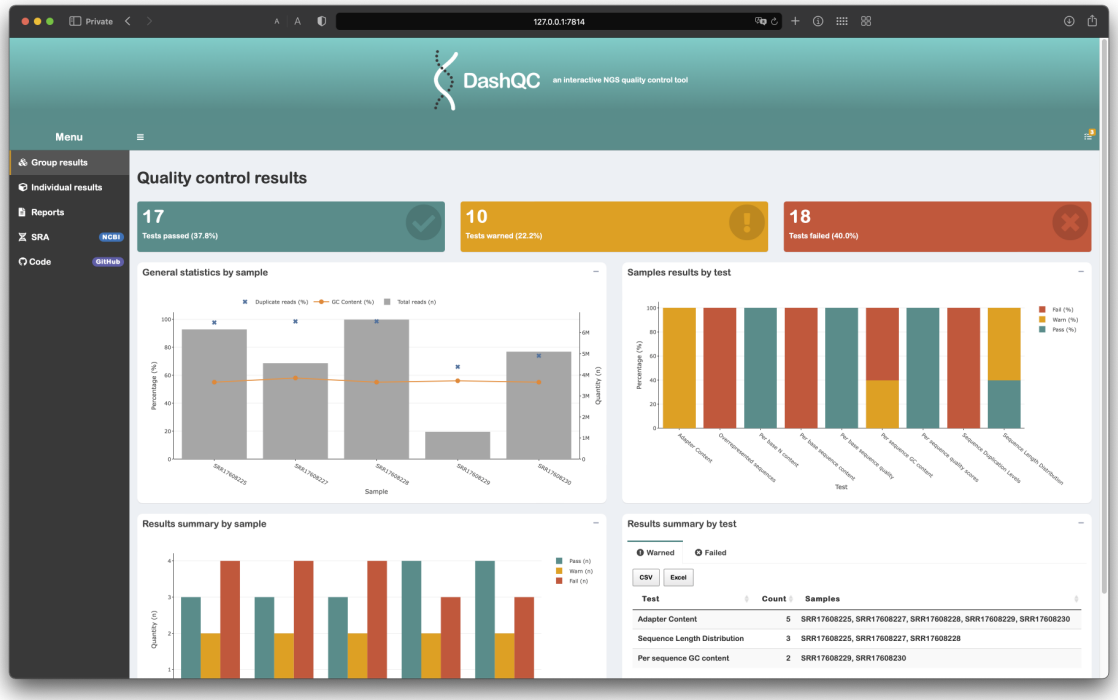


Figura 8: Pantalla de análisis de resultados grupales del *Dashboard*.

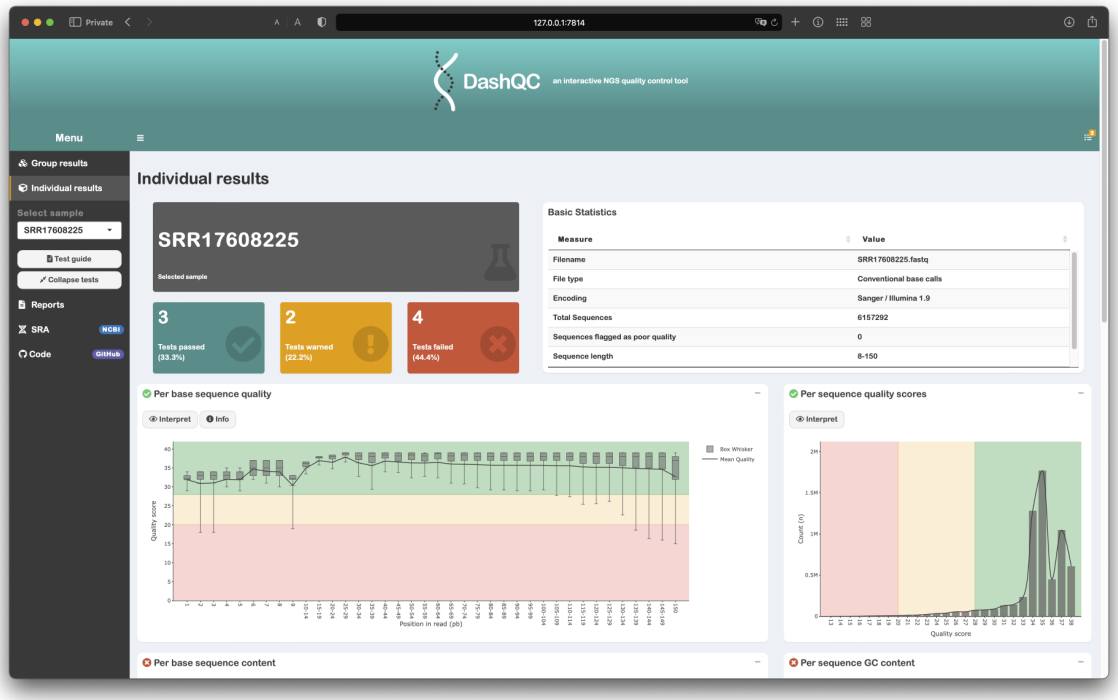


Figura 9: Pantalla de análisis de resultados individuales del *Dashboard*.

El *Test guide* funciona a forma de una guía visual, en donde se recorren todos los módulos de la sección de resultados individuales del *dashboard*, indicando con un recuadro iluminado y una caja de texto explicativo, cómo funciona cada una de las pruebas de control de calidad y qué es lo que se esta verificando en ellas. Esta herramienta es capaz de guiar al usuario final en la comprensión de las pruebas exitosamente, como se puede evidenciar en la Figura 10.

El botón de *Collapse tests* comprime los recuadros donde se encuentran las visualizaciones de la sección de resultados individuales y deja únicamente visible los nombres de las pruebas/módulos y sus iconos indicadores de aprobación, alerta o falla (Figura 11).

7.4. Interpretaciones automáticas y recomendaciones

Se creo un algoritmo en forma de árbol de decisión para la generación de interpretaciones automáticas en texto según los datos, la prueba de control de calidad y el resultado obtenido. Los textos que produce este componente, son creados a partir de investigación de la literatura que hay detrás de cada uno de los módulos de FastQC (Andrews, 2019b) así como otras fuentes oficiales donde se discute la interpretación como tal (Andrews, 2019a). En la sección de los resultados individuales, cada uno de los recuadros de los módulos de control de calidad tiene un botón especificado para desplegar la interpretación de dicha prueba. Esta es desplegada en una pantalla modal que aparece al presionar el botón (Figura 12). El modal va a contener el nombre de la prueba de QC y la interpretación y recomendación generada.

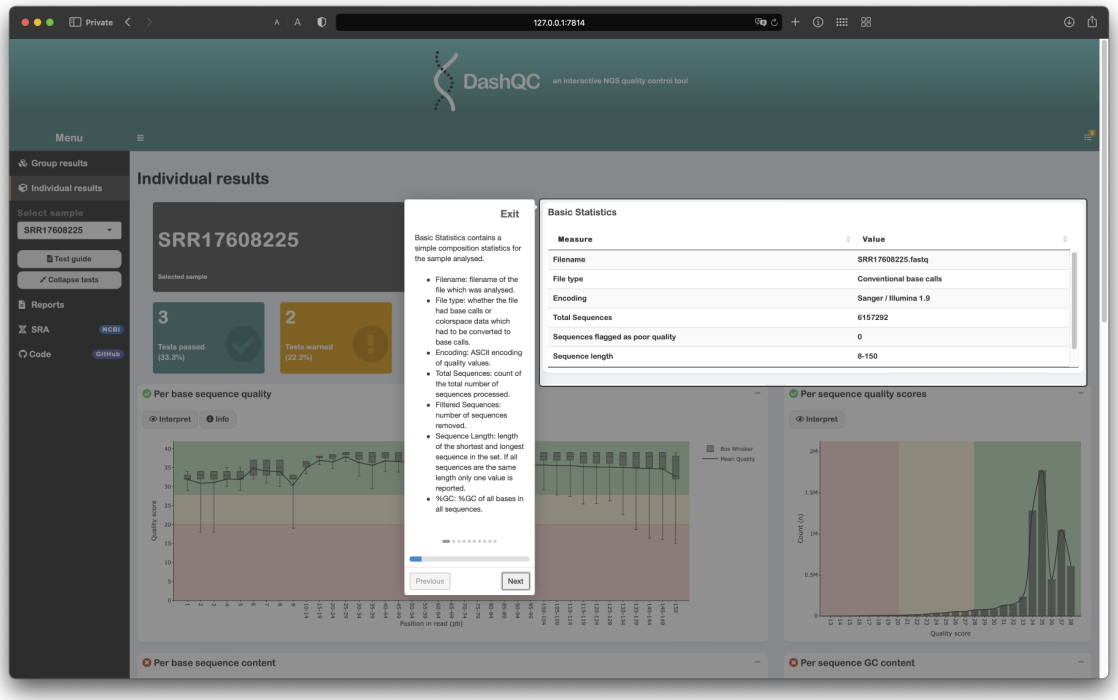


Figura 10: Guía de pruebas para una muestra individual en el *Dashboard*.

7.5. Reportes automatizados

Utilizando el paquete de *RMarkdown* se desarrollaron informes automatizados con el fin de resumir los resultados en un archivo único y descargable. Se diseñaron 2 tipos de reportes, el primero es el reporte general, y este contiene un resumen de los gráficos, tablas y resultados generales para todas las muestras (Figura 13), como se pueden observar en la sección de resultados grupales del *dashboard*. Cabe agregar que este reporte contiene también texto interpretativo para acompañar las figuras. El segundo, es el reporte individual, y este es un informe parametrizado, ya que va a responder a la selección de una muestra en específico para ser generado, como se puede observar en la Figura 35. El reporte individual contiene una recopilación de los resultados generales para la muestra y luego un desglose de los resultados por prueba de control de calidad. En este reporte, por cada prueba de control de calidad se da una explicación de lo que consiste la prueba, luego se presenta el gráfico respectivo, resultado obtenido y se finaliza con la interpretación y recomendaciones generadas (Figura 14).

Es importante mencionar que ambos de estos informes automatizados, generan un *timestamp* en la *metadata* del archivo y el encabezado. Por medio del paquete de *Knitr* y la adecuada programación con *RMarkdown* es posible generar estos reportes en distintos formatos. En el primer formato, PDF, la información dentro del reporte se encuentra fija y estática. En el segundo formato, WORD, permite modificar el texto del reporte, haciendo que este sea modificable y editable, como se puede evidenciar en la Figura 16. El tercero, el formato HTML, permite que los gráficos sean de naturaleza interactiva y que los reportes puedan ser explorados desde cualquier navegador (Figura 15).

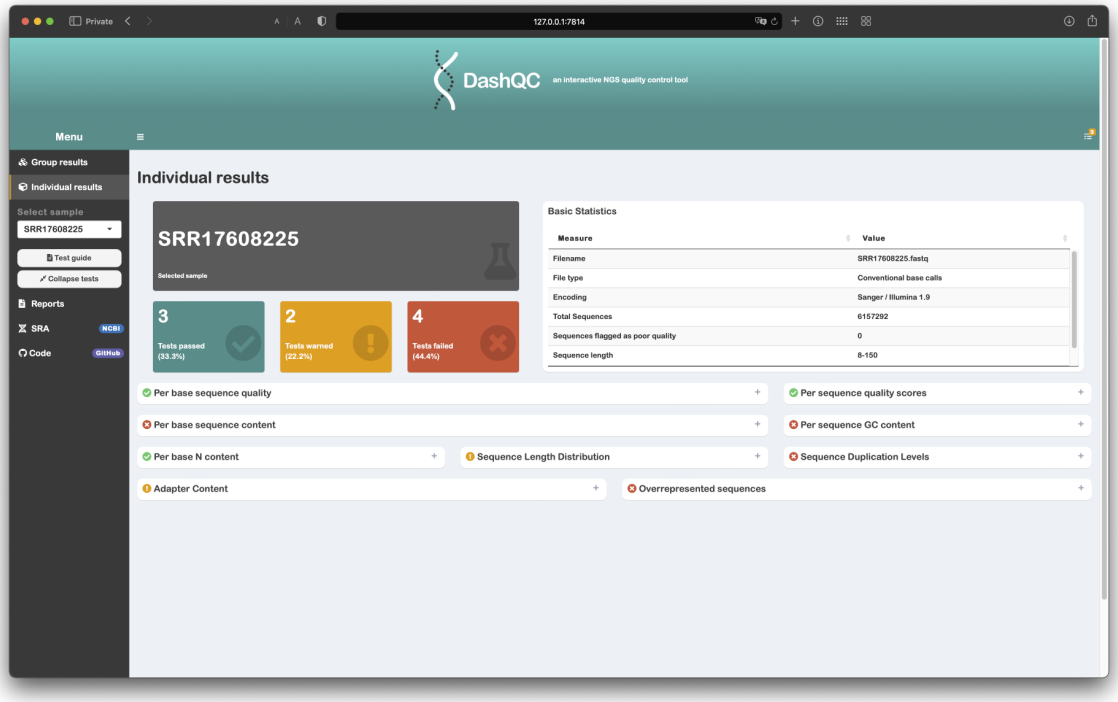


Figura 11: Vista condensada de resultados de una muestra individual en el *Dashboard*.

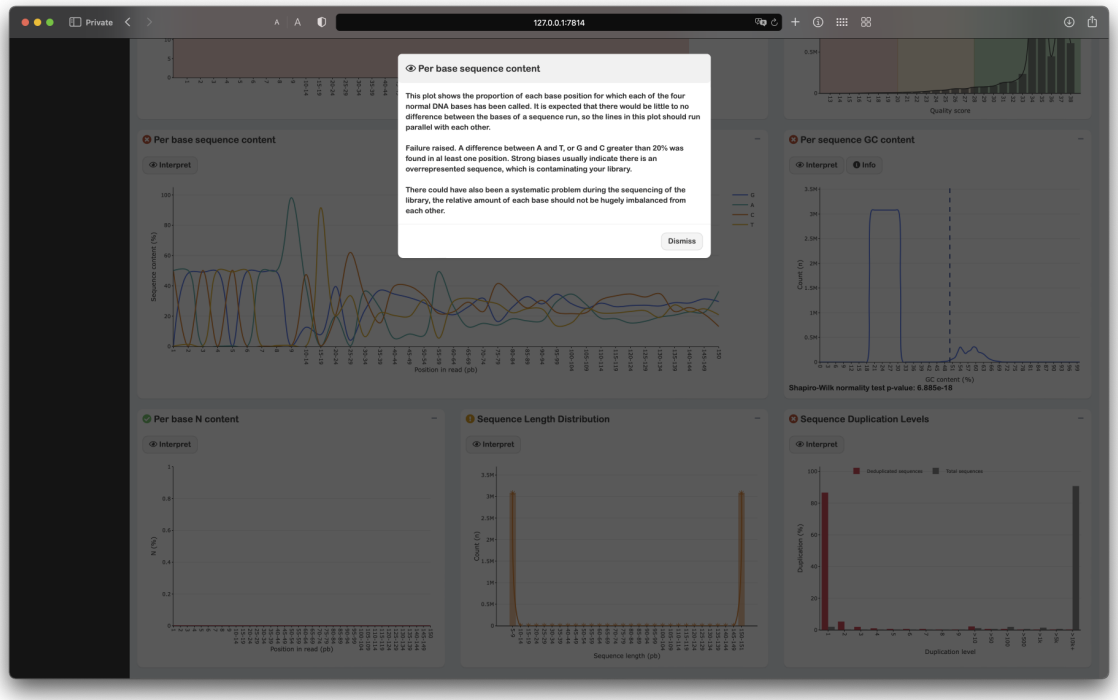


Figura 12: Pantalla modal con la interpretación generada para la prueba de *Per base sequence content* en el *Dashboard*.

General quality control results

DashQC Report - 2022-10-05 11:20:07

General report generated for samples: SRR17608225, SRR17608227, SRR17608228, SRR17608229, SRR17608230.

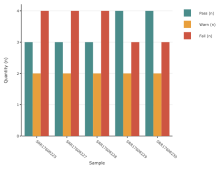
The most warnings were raised by the **Adapter Content** test, for the specific samples: SRR17608225, SRR17608227, SRR17608228, SRR17608229, SRR17608230.

The most failures were raised by the **Overrepresented sequences** test, for the specific samples: SRR17608225, SRR17608227, SRR17608228, SRR17608229, SRR17608230.

Table 1. Tests results summary and distribution

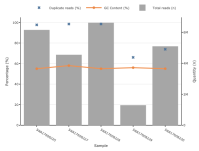
Result	n	%
Tests Passed	17	37.8
Tests Warned	10	22.2
Tests Failed	18	40.0
Total tests performed	45	100.0

Figure 1. Results count summary by sample



The plot above shows the test results count by each sample, making it simple to identify differences among them and spot, if any, have a deficient performance.

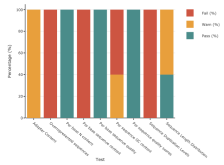
Figure 2. General statistics by sample



Quantitative general statistics contain variables such as **duplicate reads**, **guanine-cytosine content**, and **total reads**. This plot allows the visualization of these variables for comparative analysis between samples.

Duplicate reads should not be a high percentage as it may indicate enrichment bias, like PCR over amplification or a very high level of coverage of the target sequence. GC-Content is expected to vary depending on the subject sample. The total reads represent the total number of reads done for each sample during the sequencing run and the expected amount depends on the proposed goal.

Figure 3. Samples results proportion by test



The plot above shows the results proportion obtained in each standard QC test. A complete red bar indicates that all the samples failed that test. On the other hand, a complete blue bar indicates no problems encountered at all. If a test contains only a small % of fail or warning, then the individual analysis is recommended to find out what went wrong with those samples.

References

- Babraham, B. (2019). FastQC: A quality control tool for high throughput sequence data. https://www.bioinformatics.babraham.ac.uk/projects/fastqc/
- Illumina. (2021). FASTQ files explained. https://support.illumina.com/bulletins/2016/04/fastq-files-explained.html

Figura 13: Reporte general generado en formato PDF.

Quality control results - Sample: SRR17608225

DashQC Report - 2022-10-05 11:20:22

Table 1. Sample SRR17608225 basic statistics summary

Measure	Value
Filename	SRR17608225.fastq
File type	Conventional base calls
Encoding	Sanger / Illumina 1.9
Total Sequences	6157292
Sequences flagged as poor quality	0
Sequence length	8-150
%GC	55

Basic Statistics contains a simple composition statistics for the sample analysed.

- Filename: filename of the file which was analysed.
- File type: whether the file had base calls or colorspace data which had to be converted to base calls.
- Encoding: ASCII encoding of quality values.
- Total Sequences: count of the total number of sequences processed.
- Filtered Sequences: number of sequences removed.
- Sequence Length: length of the shortest and longest sequence in the set. If all sequences are the same length only one value is reported.
- %GC: %GC of all bases in all sequences.

Table 2. Sample SRR17608225 results summary and distribution

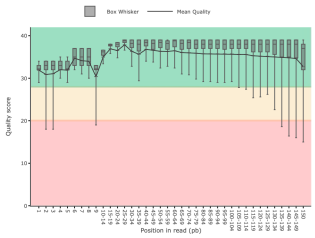
	Pass	Warn	Fail
n	3	2	4
%	33.3	22.2	44.4

The sample results summary indicates the amount of tests passed, warned, and failed of the sample analysed. The % allows the measurement of the proportion for each test indicator from a total of 9 tests performed.

Per base sequence quality

This test evaluates the range of quality values across all bases at each position in the FastQ file. The y-axis on the graph shows the quality scores. The higher the score the better the base call.

Figure 1. SRR17608225 Per base sequence quality: PASS



The y-axis shows the quality scores. The higher the score, the better the base call. The background of the graph divides the y-axis into very good quality calls (green), calls of reasonable quality (yellow), and calls of poor quality (red).

This plot shows an overview of the range of quality values across all bases at each position in the FastQ file. No warning or failure was issued, test passed. For any base, no lower quartile is less than 10 quality score and no median is less than 25 quality score.

Figura 14: Reporte individual generado en formato PDF.

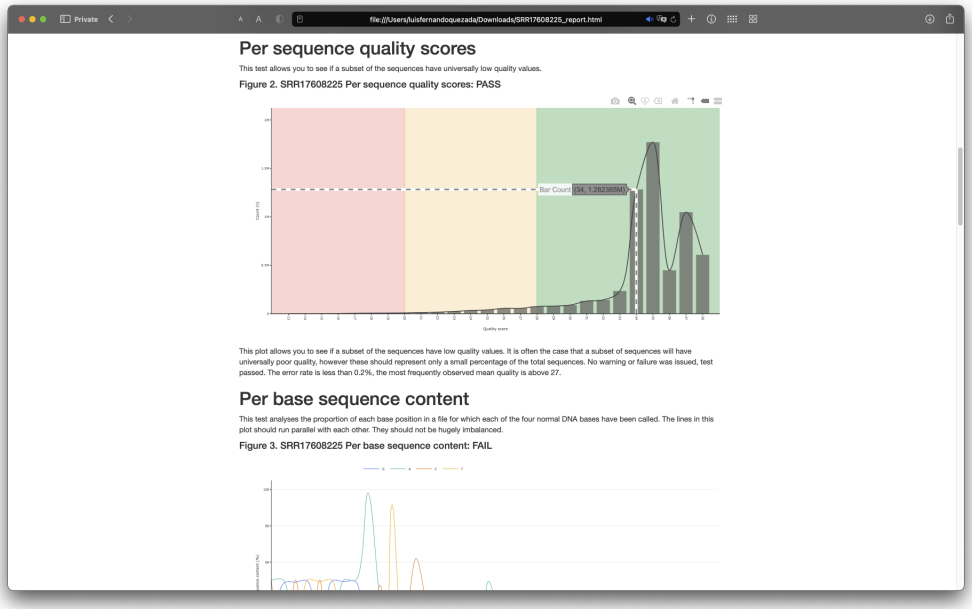


Figura 15: Reporte individual generado en formato HTML.

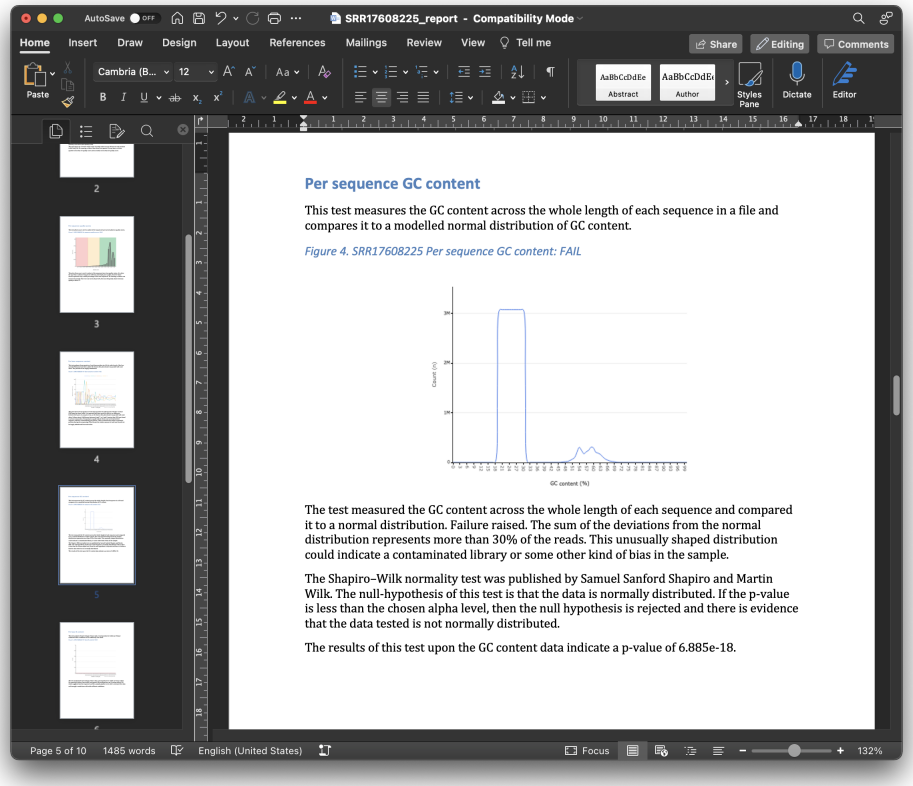


Figura 16: Reporte individual generado en formato WORD.

CAPÍTULO 8

Discusión de resultados

8.1. *Pipeline DashQC*

El valor del *pipeline* realmente proviene del funcionamiento en serie de sus componentes. Esto requiere del diseño adecuado de flujo de datos. Es por esto que a lo largo de todo el *script* de preprocesamiento, se incluyeron validaciones por partes y se comunicó a través del *logger*, si se aprobaron o no. Esto lo que permite es detener, documentar y notificar el proceso donde no se obtuvo lo esperado. Al crear estas validaciones se está hablando no solo de calidad del dato, si no también de forma y estructura esperada. Cualquier fallo en estas características significaría un mal funcionamiento del *pipeline* completo por efecto de cascada, y por eso su importancia. Estas validaciones, incluso, permiten encontrar archivos problemáticos con necesidad de limpieza y depuración o incluso de descarte. Parte de la magia que existe detrás de un *pipeline*, es que puede ser desarrollada por componentes diferentes con propósitos específicos. En el caso de este trabajo, fue ese el acercamiento que se implementó.

El tiempo es un elemento que juega un papel importante en toda investigación o proyecto. Parte de los beneficios más grandes que brindan los *pipelines* es la automatización. No hay razón alguna para no volver automático cualquier proceso que se realiza de la misma manera múltiples veces. En el área científica, no hay tiempo que desperdiciar, y en muchos casos, se puede utilizar el desarrollo de software para crear estas automatizaciones que permiten dedicar más tiempo a otros enfoques, como lo es, el propio análisis de resultados. Por medio del *pipeline DashQC* se logra una automatización completa de todo el proceso que toma correr el control de calidad para un conjunto de muestras. Lo que significa que no se gastó energía ni tiempo adicional en hacer ese proceso de forma individual y tener que buscar información en múltiples archivos para comparar entre ellos. El proceso manual, en especial para grandes números de muestras, puede resultar agotador, repetitivo e incluso llegar a

producir errores accidentales. Cabe resaltar que la principal importancia de implementar el *pipeline DashQC* es por la capacidad que tiene de hacerse cargo del trabajo pesado y habilitar más tiempo para analizar con detenimiento los resultados con el fin de facilitar la comprensión de lo obtenido y robustecer la toma de decisiones.

La manera en la que fue diseñado este *pipeline* abre las puertas a ser utilizado dentro de marcos de trabajo más grandes. Resultaría bastante sencillo conectar los datos de entrada a una base de datos e involucrar a *DashQC* como el componente de QC, previo al *trimming* y *filtering* en un estudio de gran inversión, por ejemplo. También podría conectarse a los datos de salida de un secuenciador, en un laboratorio donde se haga investigación genómica. Incluso, el *pipeline* podría usarse para depurar de forma rutinaria bases de datos de secuenciación de nueva generación y crear *datasets* curados. Su interoperabilidad con otras herramientas es el valor agregado del correcto diseño de un *pipeline* con puntos de entrada y salida estándar.

8.2. *Dashboard DashQC*

La visualización adecuada de la información realmente puede hacer un impacto en cómo se interpreta y las conclusiones que se pueden sacar de ella. La visualización grupal del *dashboard* demuestra ser de gran utilidad por que permite al usuario tener una visión general sobre el desempeño de las muestras utilizadas (Figura 8). Esta sección es de suma importancia cuando se está realizando una investigación con gran cantidad de muestras, dado que permite identificar aquellas que son deficientes en calidad rápidamente sin necesidad de involucrar un proceso manual de recolección de datos a nivel macro. Por otro lado, permite una forma de realizar comparaciones entre las muestras y conocer la frecuencia de aprobación o desaprobación obtenida por cada una. De igual manera, esta sección hace posible conocer si es una prueba de QC específica la que está afectando al conjunto y en que medida lo hace (Figura 21). Los gráficos y tablas pueden ser exportados y ser de gran utilidad para tener registros para conjuntos de corridas o formar parte de un análisis posterior que el investigador desee. Por ejemplo, el gráfico de *General statistics by sample* (Figura 20) se puede utilizar para verificar que no exista sobre-amplificación por PCR o niveles altos de duplicación en las muestras seleccionadas. También puede utilizarse para asegurar que no hay contaminación y que las muestras sí son del organismo secuenciado por medio de la comparación del %GC con el porcentaje teórico del organismo. Con el gráfico de *Results summary by sample* (Figura 22) se simplifica la identificación de las diferencias entre resultados por muestra y se facilita la detección de alguna con rendimiento deficiente.

Parte de la virtud de tener un componente interactivo en el *pipeline* es que es posible navegar los datos de forma específica y detallada. La sección de visualización individual (Figura 9) se desarrolló de tal manera que habilita al usuario recorrer la información específica de una muestra; no solo con el fin de conocer sus resultados por cada prueba estándar de QC, si no también permitir interactuar con los datos puros y recibir una interpretación textual explicando las razones claras de los resultados de aprobación de dichas pruebas. Evidentemente esta es una sección de análisis más específica y no pretende solamente dar "un vistazo rápido", como lo es la sección grupal, por el contrario, está desarrollada para habilitar la profundización sobre una prueba o corrida puntual. Sin embargo, se añadió una

forma de hacer la navegación de esta página lo más amigable y entendible posible por medio de 2 botones.

El botón de *Test guide*, como explicado previamente, habilita el modo guía, en donde aparece un panel de control, donde se puede navegar y avanzar paso a paso por los contenidos de la sección de resultados individuales. Es decir, esta herramienta conduce al usuario, prueba por prueba, indicándole el recuadro sobre el que se debe enfocar (como una lupa) y proporcionando información general de cada prueba de control de calidad como se puede observar en la Figura 10. Esto lo hace de tal forma que se le da al usuario la teoría resumida que hay detrás de cada una de las pruebas. Es información de soporte que está a la disposición del usuario y evita que tenga buscar algún tipo de manual o documentación en internet, ahorrando tiempo en la práctica. La opción de *Test guide* se puede utilizar cuantas veces se desee e incluso apoyarse de ella para una presentación introductoria a los análisis individuales de una muestra, siendo así, un agregado valioso a esta sección de *DashQC*. El segundo botón, el botón de *Collapse all* deja únicamente visible los nombres de las pruebas y sus iconos indicadores de aprobación, alerta o desaprobación, lo cual reduce la cantidad de contenido visual presentado al usuario en esta sección y le habilita la opción de expandir únicamente aquellas pruebas que desee ver (Figura 11). Es una herramienta que simplifica el contenido presentado en la sección y puede utilizarse como un atajo para ver información específica.

Los gráficos de los módulos de la sección de resultados individuales fueron recreados interactivamente y su diseño fue mejorado para volver el proceso de comprensión lo más intuitivo posible. Por ejemplo, la utilización de colores de fondo para los rangos de aprobación en la prueba de *Per sequence quality score* (Figura 27), la adición del %GC en línea punteada a la prueba de *Per sequence GC content* (Figura 29) o la aplicación de *smoothing* al gráfico de *Per base sequence content* (Figura 28). Estos y otros cambios aplicados a la visualización también le dan una modernización a los gráficos estándar de FastQC que se han visto desde siempre. Adicional a esto, se trabajó en crear un componente capaz de construir textualmente interpretaciones basadas en la literatura y naturaleza de cada prueba de QC. Estas son de suma importancia para poder comprender la razón específica de un resultado y acompañan al usuario de una explicación, con la cual, respaldar lo que se está observando en cada módulo. Cabe agregar que no es necesario tomar un *screenshot* o copiar y pegar estas en un documento aparte para guardarlas ya que son las mismas interpretaciones que se adjuntan en los reportes. De tal forma que si se desea archivar múltiples interpretaciones, gráficos o resultados es más conveniente aprovechar de la sección de reportería y tener esto consolidado en un documento científico, debidamente identificado.

La sección de reportería, apreciable en la Figura 35, justamente cumple el propósito de guardar y compartir los resultados de los análisis. Además, como mencionado anteriormente, por medio del sistema de *timestamping* incluso se puede documentar y tener trazabilidad sobre corridas de secuenciación (en caso se estén generando archivos FASTQ propios) o muestro (selección de una base de datos local u *online*). El valor agregado de la selección de la tecnología *RMarkdown* es la posibilidad de programar un documento para poderse generar en múltiples formatos, permitiendo tener estos reportes disponibles de forma estática, interactiva y editable. El formato PDF (estático) esta pensado para ser un documento de almacenaje y registro de lo ocurrido con una muestra (bastante útil en sistemas de trazabilidad) (Figura 36 y 37). Mientras tanto, el formato HTML (interactivo) está pensado como una forma de presentación y exploración de los datos que puede ser utilizado como

un *snapshot* del *dashboard* en vivo (Figura 15). Por otro lado, el formato WORD (.docx) tiene como meta ser utilizado como un punto de partida para elaborar un informe más personalizado por medio de la edición de los contenidos, adición a las interpretaciones y/o explicaciones o, si fuese necesario, eliminación de algún contenido no deseado (Figura 16). La idea del tablero es sobrepasar cualquier impedimento relacionado al análisis y exploración de datos, y proveer al usuario con todos los recursos necesarios para sacar sus propias conclusiones. Esto convierte a *DashQC* en una herramienta con validez científica y de uso profesional.

8.3. Alcance de objetivos

En base a los resultados obtenidos, es posible concluir que se han cumplido los objetivos planteados. El *pipeline DashQC* es capaz de recibir como entrada archivos FASTQ, realizar pruebas de control de calidad estandarizadas (FastQC) y desplegar los resultados de forma interactiva con distintas herramientas para asistir en la interpretación de los resultados. Estas herramientas incluyen gráficas interactivas, secciones de visualización grupal o individual, guías paso a paso, cuadros informativos y reportería automatizada. El presente proyecto se ha llevado a cabo con éxito por medio de la creación de dicho *pipeline*, que presenta una nueva forma de realizar el control de calidad de múltiples muestras de datos de secuenciación de nueva generación.

8.4. Dificultades y retos

Durante el desarrollo del proyecto sin duda se encontraron varias complicaciones que realmente fueron un desafío. Una de ellas fue con la lectura agregada de los archivos ZIP producidos luego de pasar los archivos FASTQ por el control de calidad. El paquete de *fastqcr* fue de gran ayuda para sobrepasar este bloqueo, sin embargo, la documentación era algo confusa sobre cómo solucionar problemas de memoria, *paths* y parámetros de las funciones. Sin embargo con un poco de exploración y *testing*, se comprendió el funcionamiento y posteriormente se logró hacer la lectura adecuada. Otra de las complicaciones principales fue crear la composición de la estructura de datos diseñada (DQC.RData), ya que el código debía contemplar una forma estructurada y escalable para almacenar la información cruda de n muestras y permitir su legibilidad parametrizada luego (para el *dashboard*). Para esto se usó un proceso iterativo, donde por medio de un ID de la muestra se guardaban sus datos en un objeto consolidado (un diccionario de diccionarios de cuadros de datos).

Cabe agregar que a la hora de diseñar los reportes grupales e individuales no hubo mucha problemática por que se trabajó con una versión del DQC.RData precargado. Se encontraron dificultades hasta el momento de anexar la generación de estos reportes al *dashboard*. Esto se debe a que el tablero cuenta con un ambiente de variables al cual los *RMarkdown* debían tener acceso y ser capaces de realizar manipulación sobre esos datos para hacer filtrado (en el caso del reporte individual, que es parametrizado según la muestra seleccionada). Luego de un estudio detallado del funcionamiento de ambientes y jerarquía de datos en *shiny* se logró sobrepasar este reto.

En la sección de resultados individuales, fue un reto lograr que salieran íconos reactivos de aprobación, alerta o fallo al lado del nombre de cada módulo. Fue uno de los desafíos principales por que la comunicación del *server* con el *ui* en el tablero tiene sus limitaciones y los *boxes* de *shiny* realmente no están pensados de esta forma. Después de realizar diferentes intentos se logró codificar estos íconos reactivos con ayuda de funciones reactivas, HTML y CSS para el estilo. Finalmente, el último reto a recalcar fue uno que no fue posible cumplir. Se tenía la idea de poder hacer una versión del *pipeline* que únicamente genere el DQC.RData y luego fuera posible cargar este archivo a un tablero *online*; sin embargo, tras múltiples pruebas y extensa investigación, se abandonó la idea ya que no se encontró una solución robusta para este desafío. No obstante, esta dificultad fue de gran ayuda para definir el límite de la tecnología utilizada en este proyecto.

8.5. Pasos a futuro

Como ha sido mencionado en otras secciones de este trabajo, el crecimiento acelerado de la información biológica disponible y la facilidad de su obtención ha inspirado al desarrollo de herramientas para la investigación de datos de secuenciación, como lo es *DashQC*. La necesidad de herramientas para poder trabajar con estos datos es inminente. Lamentablemente, muchas herramientas existentes no reciben actualizaciones frecuentemente y con el paso de los años se vuelve complicado incluso correrlas, por el simple hecho de que su documentación es escasa y hay que configurar un ambiente con versiones de software antiguo. Los planes de *DashQC* a mediano plazo consisten principalmente en mantener el *pipeline* funcional conforme van saliendo nuevas versiones de los paquetes y librerías que utiliza; madurando el código. Por otra parte, gran cantidad del software existente o servicios de control de calidad y depuración de datos de secuenciación tienen altos costos por cada una de las muestras. *DashQC* utiliza software y librerías *open source* para poner a disponibilidad estos servicios sin ningún costo para fines educativos, y los planes a largo plazo son mantener esta idea, es decir, este es un proyecto *para la academia*.

CAPÍTULO 9

Conclusiones

DashQC presenta un flujo de procesos que automatiza, asiste y presenta el control de calidad de datos de secuenciación de nueva generación de forma visual, interactiva e intuitiva. Se ha creado con éxito un *pipeline* capaz leer múltiples archivos FASTQ, realizar su control de calidad y representar los resultados con una herramienta gráfica (*dashboard*), permitiendo así un acceso interactivo a los datos. Además, se ha logrado implementar exitosamente un proceso guiado para la comprensión de las pruebas y resultados de control de calidad, interpretaciones y recomendaciones por aplicar. De igual forma, se creó un sistema de reportería que permite al usuario generar reportes para almacenar o compartir de forma estática, interactiva o editable los resultados de sus análisis con validez científica y trazabilidad. *DashQC*, al ser una herramienta *open source* y capaz de formar parte de un marco de trabajo más grande, se convierte en una ventaja en cualquier estudio o análisis que se desee llevar a cabo.

CAPÍTULO 10

Recomendaciones

La metodología descrita para el proyecto de *DashQC* es bastante eficiente ya que, los pasos posteriores se basan en la elaboración de los pasos anteriores, es decir, el progreso es acumulativo. Sin embargo, esto puede representar una desventaja, ya que el atraso en uno de los pasos de la metodología puede representar un cuello de botella crítico, por lo que se recomienda compartimentalizar el proyecto por módulos de código que funcionen por individual, acompañados de chequeos de QA y requisitos. Adicionalmente, es recomendable tomar en cuenta el versionando adecuado de librerías ya que existen algunos paquetes que requieren de versiones anteriores de otras librerías para cumplir con sus dependencias.

Con respecto a la capacidad computacional disponible, se sugiere adaptar el número de *threads* utilizadas para correr *fastqc* con el *pipeline DashQC* a las capacidades del equipo disponible, tanto en investigación, como en laboratorio. Por otro lado, es importante mencionar que al trabajar con documentos *Rmarkdown* en formato PDF y WORD desde un *dashboard*, los tiempos de la renderización aumentan en relación a la cantidad de figuras y contenido que estos contengan, por lo que se recomienda trabajar con reportería en formato HTML si se desean tiempos cortos/instantáneos de generación y descarga. Finalmente, se aconseja siempre tomar en cuenta selecciones de paletas de color apropiadas para la correcta visualización de los datos. Para el caso de *DashQC* se contemplaron incluso distintos tipos de cegueras de color, para hacer el *dashboard* apto para todo público, tanto en una presentación formal de resultados, como un componente de una investigación o parte de una documento/publicación científica.

CAPÍTULO 11

Bibliografía

ABM. (2016). *Next Generation Sequencing (NGS) - Data Analysis.* https://old.abmgood.com/marketing/knowledge_base/next_generation_sequencing_data_analysis.php#collapseOne

Alboukadel Kassambara, J. H. & Ahmed, M. (2018). *fastqcr: Quality Control of Sequencing Data.* https://github.com/kassambara/fastqcr

Alva, J. A. V. & Estrada, E. G. (2009). A Generalization of Shapiro–Wilk's Test for Multivariate Normality. *Communications in Statistics - Theory and Methods*, *38*(11), 1870-1883. https://doi.org/10.1080/03610920802474465

Andrews, S. (2019a). *Adapter Content.* https://www.bioinformatics.babraham.ac.uk/projects/fastqc/Help/3%5C%20Analysis%5C%20Modules/10%5C%20Adapter%5C%20Content.html

Andrews, S. (2019b). *FastQC: A quality control tool for high throughput sequence data.* https://www.bioinformatics.babraham.ac.uk/projects/fastqc/

Arifin, W. N. (2019). Introduction to R and RStudio IDE.

AWS. (2022). *What is Data Science?* https://aws.amazon.com/what-is/data-science/

Bayat, A. (2002). Science, medicine, and the future: Bioinformatics. *BMJ*, *324*(7344).

Behjati, S. & Tarpey, P. (2013). *What is next generation sequencing?* https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3841808/

BYU. (2022). *R AND RSTUDIO.* https://statistics.byu.edu/r-and-rstudio

CRAN. (2019). *fastqcr: Quality Control of Sequencing Data.* https://rdrr.io/cran/fastqcr/

Efron, B. & Hastie, T. (2021). *Computer Age Statistical Inference, Student Edition: Algorithms, Evidence, and Data Science* (Vol. 6). Cambridge University Press.

GEN. (2017). *Next-Generation Sequencing Challenges: NGS Growing By Leaps and Bounds, Problems Arise.* https://www.genengnews.com/magazine/286/next-generation-sequencing-challenges/

Hannon, L. (2010). *FASTX-Toolkit: FASTQ/A short-reads pre-processing tools.* http://hannonlab.cshl.edu/fastx_toolkit/

HBS. (2021). *DATA STORYTELLING: HOW TO EFFECTIVELY TELL A STORY WITH DATA.* https://online.hbs.edu/blog/post/data-storytelling#:~:text=What%5C%

20Is%5C%20Data%5C%20Storytelling%5C%3F,inspire%5C%20action%5C%20from%5C%20your%5C%20audience.
Iberdrola. (2020). *What is bioinformatics and what is its impact on health?* https://www.iberdrola.com/innovation/bioinformatics
IBM. (2021). *Data Visualization.* https://www.ibm.com/cloud/learn/data-visualization
Illumina. (2021). *FASTQ files explained.* https://support.illumina.com/bulletins/2016/04/fastq-files-explained.html
JHU. (2021). *Bowtie: An ultrafast memory-efficient short read aligner.* http://bowtie-bio.sourceforge.net/index.shtml
Khetani, R. (2020). *RNA-seq using high-performance computing (HPC).* https://hbctraining.github.io/Intro-to-rnaseq-hpc-salmon/lessons/qc_fastqc_assessment.html
Mazzu, G. (2022). *Bioinformatics Pipeline & Tips For Faster Iterations.* https://www.weka.io/blog/bioinformatics-pipeline/#:~:text=What%5C%20is%5C%20a%5C%20bioinformatics%5C%20pipeline,generate%5C%20interpretations%5C%20from%5C%20this%5C%20data.
Microsoft. (2022). *Visualización de datos | Microsoft | Power BI.* https://powerbi.microsoft.com/es-es/
Mirzaee, A. (2022). *Markdown and LaTeX introduction.* https://ashki23.github.io/markdown-latex.html
NCBI. (2022a). *The NCBI SRA (Sequence Read Archive).* https://github.com/ncbi/sra-tools/wiki/
NCBI. (2022b). *Sequence Read Archive (SRA).* https://www.ncbi.nlm.nih.gov/sra
NCBI. (2022c). *The Sequence Read Archive (SRA): Getting Started.* https://www.ncbi.nlm.nih.gov/sra/docs/
NHGR. (2022a). *BIOINFORMATICS.* https://www.genome.gov/genetics-glossary/Bioinformatics
NHGR. (2022b). *DNA Sequencing Fact Sheet.* https://www.genome.gov/about-genomics/fact-sheets/DNA-Sequencing-%20Fact-Sheet
Nichols, D. (2022). *Coloring for Colorblindness.* https://davidmathlogic.com/colorblind/
NIH. (2021). *The Cost of Sequencing a Human Genome.* https://www.genome.gov/about-genomics/fact-sheets/Sequencing-Human-Genome-cost
Plotly. (2022). *Getting Started with Plotly in R.* https://plotly.com/r/getting-started/
RIDOM. (2022). *Download FASTQ from SRA.* https://www.ridom.de/u/Download_FASTQ_from_SRA.html
Roy, S. (2020). *Next-Generation Sequencing Bioinformatics Pipelines.* https://www.aacc.org/cln/articles/2020/march/next-generation-sequencing-bioinformatics-pipelines
Schmieder, R. & Edwards, R. (2011). Fast Identification and Removal of Sequence Contamination from Genomic and Metagenomic Datasets. *PLOS ONE*, *6*(3), 1-11. https://doi.org/10.1371/journal.pone.0017288
SIB. (2018). *What is bioinformatics.* https://www.sib.swiss/what-is-bioinformatics
Simpson, J. T. (2014). Exploring genome characteristics and sequence quality without a reference. *Bioinformatics*, *30*(9), 1228-1235. https://doi.org/10.1093/bioinformatics/btu023
Tableau. (2022a). *Business Intelligence and Analytics.* https://www.tableau.com/
Tableau. (2022b). *What is a dashboard? A complete overview.* https://www.tableau.com/learn/articles/dashboards/what-is

Trivedi, U. H., Cézard, T., Bridgett, S., Montazam, A., Nichols, J., Blaxter, M. & Gharbi, K. (2014). *Quality control of next-generation sequencing data without a reference*. https://www.frontiersin.org/articles/10.3389/fgene.2014.00111

Vidhya, A. (2019). *Introduction to R Shiny dashboards*. https://medium.com/analytics-vidhya/introduction-to-r-shiny-dashboards-d2ce9451c2c3

Wilke, C. (2019). *Fundamentals of Data Visualization: A Primer on Making Informative and Compelling Figures*. O'Reilly Media. https://books.google.com.gt/books?id=XmmNDwAAQBAJ

Xie, Y., Dervieux, C. & Riederer, E. (2020). *R markdown cookbook*. Chapman; Hall/CRC.

CAPÍTULO 12

Anexos

12.1. Versiones del software utilizado para el desarrollo

12.1.1. Versión de R y sistema

- R: 4.0.2
- Plataforma: x86_64-apple-darwin17.0 (64-bit)
- Desarrollo: OS X 12.1

12.1.2. Paquetes de R base

- stats
- graphics
- grDevices
- utils
- datasets
- methods
- base

12.1.3. Paquetes de R adicionales

rmarkdown_2.10	knitr_1.33	hash_2.2.6.2	shinyjs_2.1.0	rintrojs_0.3.0
fastqcr_0.1.2	forcats_0.5.1	stringr_1.4.0	purrr_0.3.4	readr_2.1.2
tibble_3.1.3	tidyverse_1.3.1	htmltools_0.5.1.1	DT_0.18	tidyr_1.1.3
data.table_1.14.2	dplyr_1.0.7	shinyBS_0.61	plotly_4.10.0	ggplot2_3.3.5
kableExtra_1.3.4	fontawesome_0.2.2	shinydashboard_0.7.1	rstudioapi_0.13	shiny_1.6.0

Cuadro 2: Tabla de paquetes R y versiones utilizadas para el desarrollo del *pipeline*

12.2. *Pipeline DashQC*

12.2.1. Ejecución del *pipeline*

1. Verificar que el equipo tenga R instalado.
2. Disponer de un directorio con el código fuente de *DashQC*.
   - run_dashqc (*Unix Executable File*)
   - Setup.R
   - Preprocess.R
   - app.R
   - group_report.Rmd y indiv_report.Rmd
   - Carpeta *www* del proyecto (CSS y *assets*)
3. Colocar archivos FASTQ de interés en una carpeta dentro de este directorio bajo el nombre de "FASTQ".
4. Correr en un CLI (con disponibilidad de ejecutar *scripts* en Bash) el siguiente comando: ***./run_dashqc***

12.2.2. Documentación del *script* del *pipeline*

Figura 17: Archivo ejecutable para correr el *pipeline DashQC* en Bash y en R.

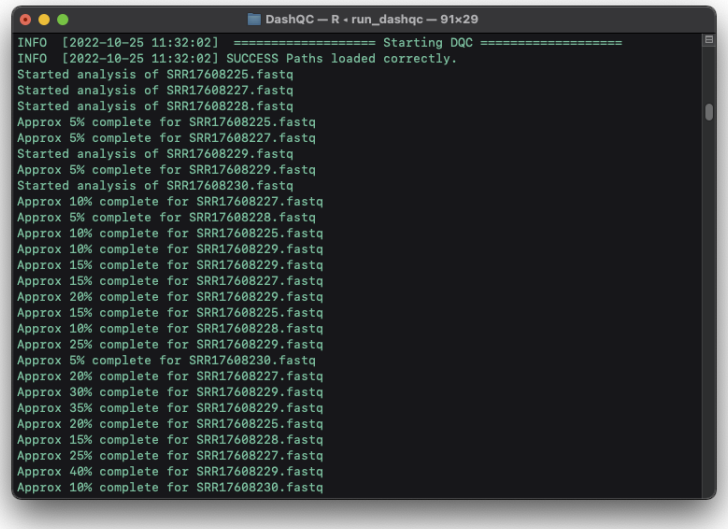


Figura 18: Ejecución en curso del *pipeline* de *DashQC* en la línea de comando.

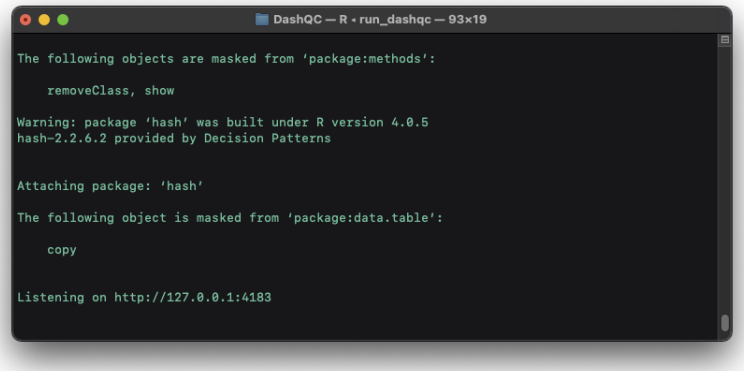


Figura 19: Vínculo local provisto por el *pipeline* para la visualización del *dashboard*.

12.3. Módulos de la sección de resultados grupales del *dashboard*.

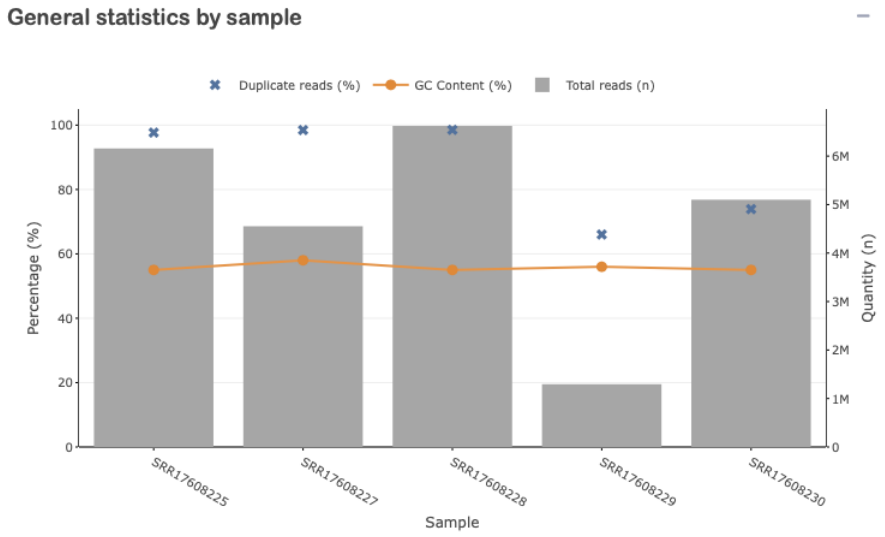


Figura 20: Gráfico del módulo de *General statistics by sample* del *dashboard*.

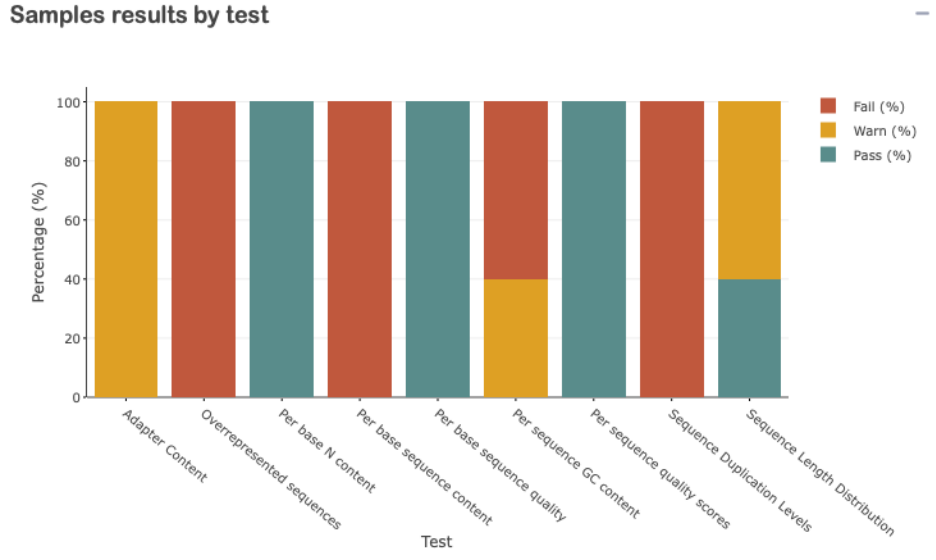


Figura 21: Gráfico del módulo de *Samples results by test* del *dashboard*.

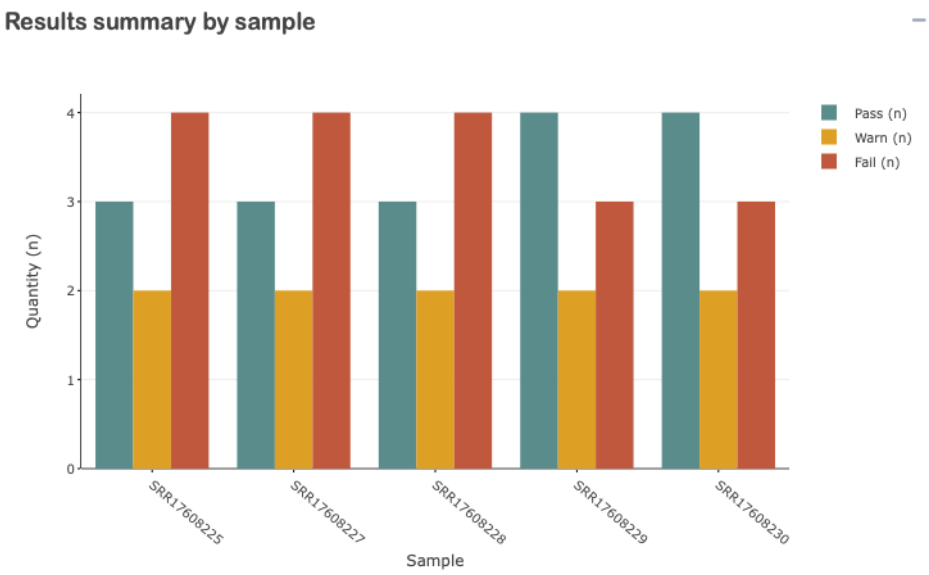


Figura 22: Gráfico del módulo de *Results summary by sample* del *dashboard*.

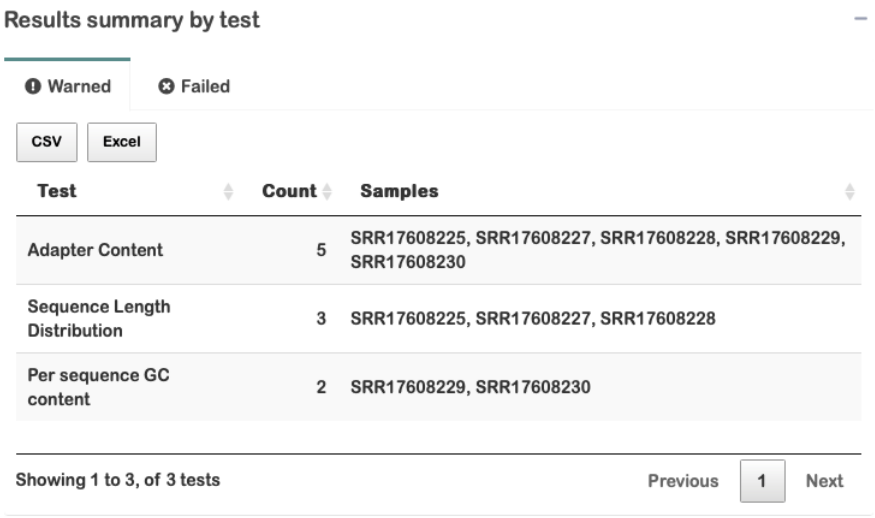


Figura 23: Tabla del módulo de *Results summary by test* del *dashboard*.

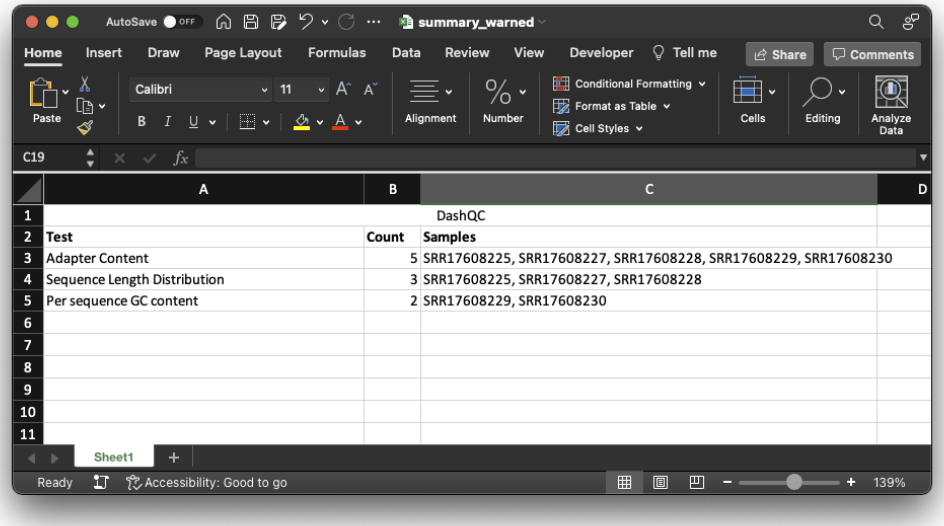
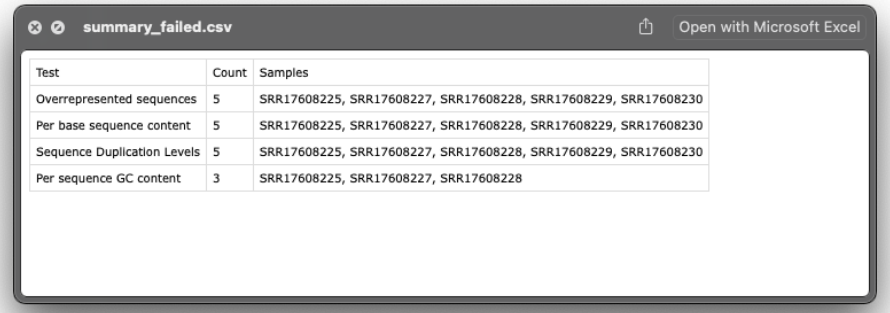


Figura 24: Tabla exportada en excel del módulo de *Results summary by test* del *dashboard*.



Test	Count	Samples
Overrepresented sequences	5	SRR17608225, SRR17608227, SRR17608228, SRR17608229, SRR17608230
Per base sequence content	5	SRR17608225, SRR17608227, SRR17608228, SRR17608229, SRR17608230
Sequence Duplication Levels	5	SRR17608225, SRR17608227, SRR17608228, SRR17608229, SRR17608230
Per sequence GC content	3	SRR17608225, SRR17608227, SRR17608228

Figura 25: Tabla exportada en CSV del módulo de *Results summary by test* del *dashboard*.

12.4. Módulos de la sección de resultados individuales del *dashboard*.

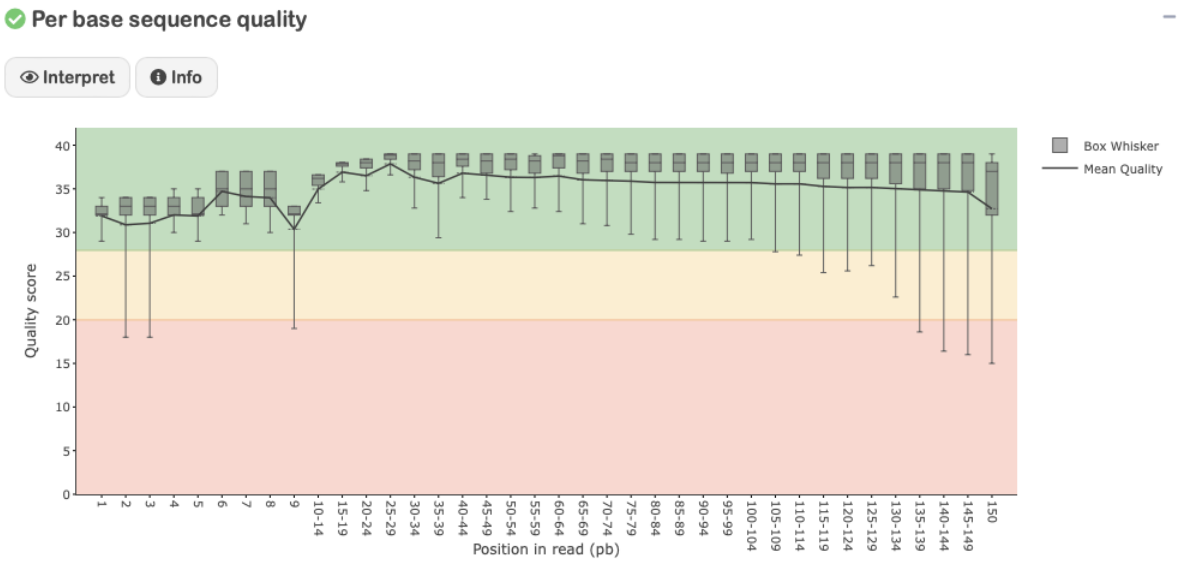


Figura 26: Gráfico del módulo de *Per base sequence quality* del *dashboard*.

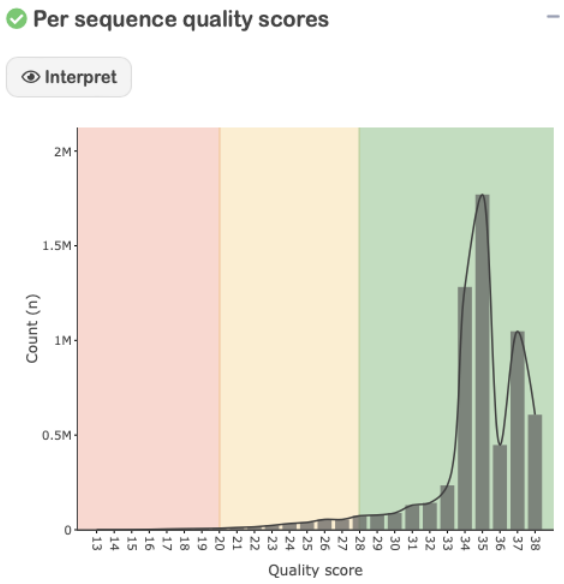


Figura 27: Gráfico del módulo de *Per sequence quality scores* del *dashboard*.

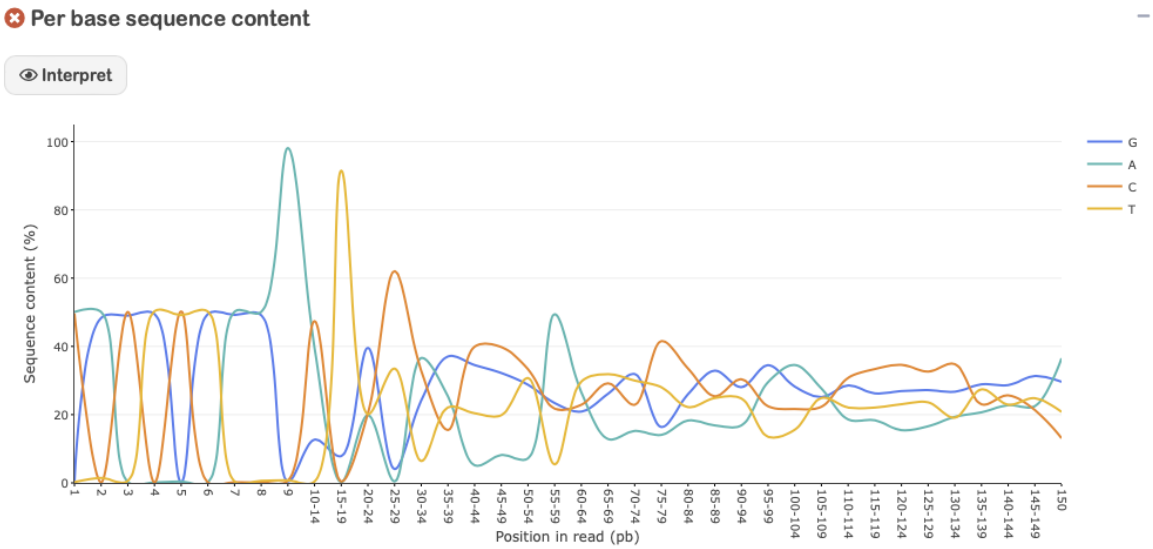


Figura 28: Gráfico del módulo de *Per base sequence content* del *dashboard*.

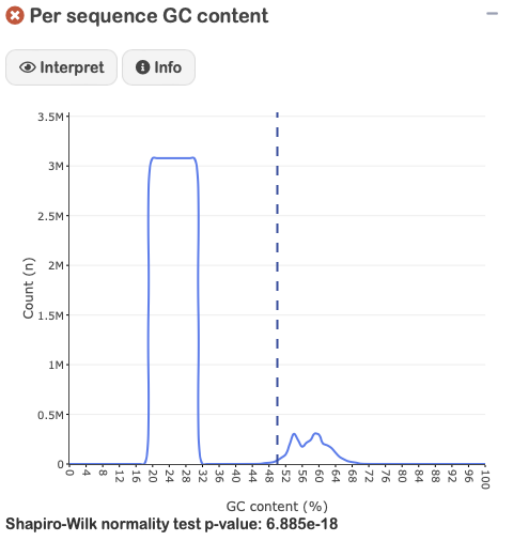


Figura 29: Gráfico del módulo de *Per sequence GC content* del *dashboard*. Asiste la significancia estadística con la prueba de normalidad de Shapiro-Wilk.

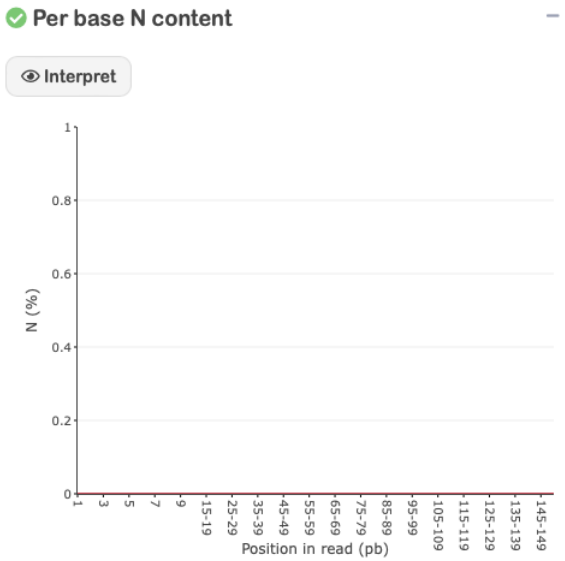


Figura 30: Gráfico del módulo de *Per base N content* del *dashboard*.

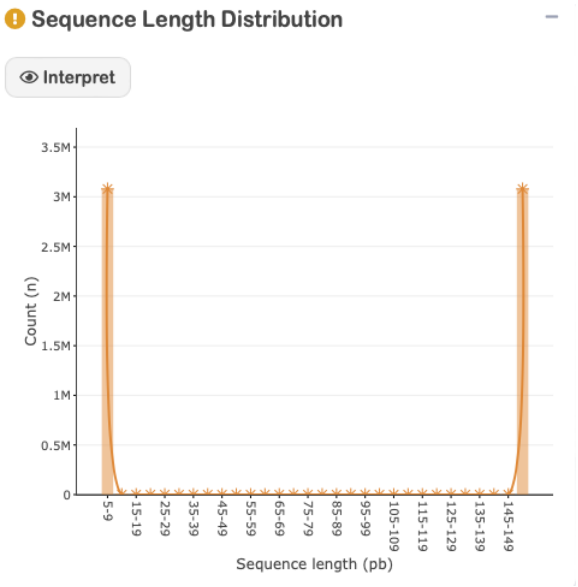


Figura 31: Gráfico del módulo de *Sequence Length Distribution* del *dashboard*.

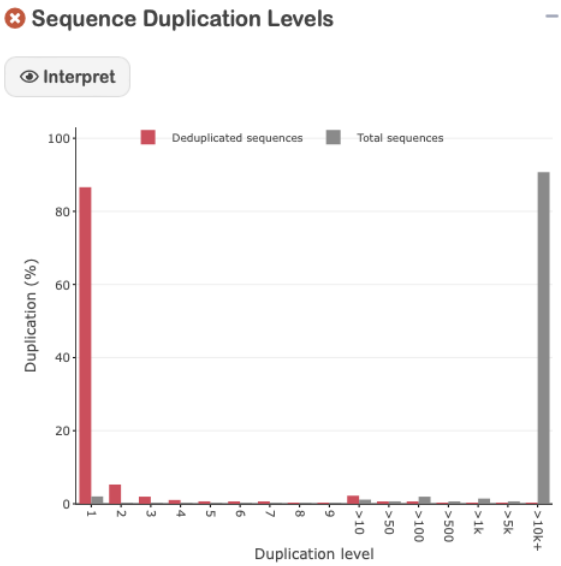


Figura 32: Gráfico del módulo de *Sequence Duplication Levels* del *dashboard*.

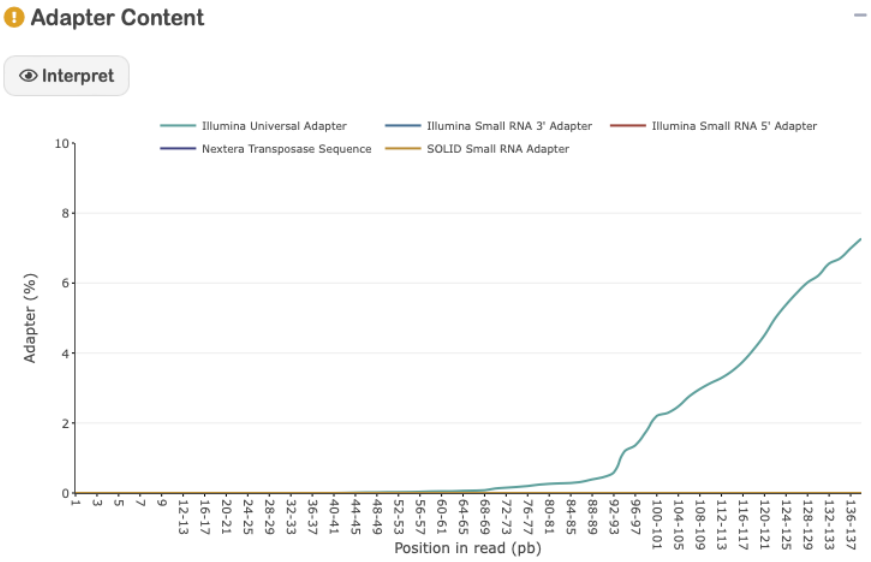


Figura 33: Gráfico del módulo de *Adapter Content* del *dashboard*.

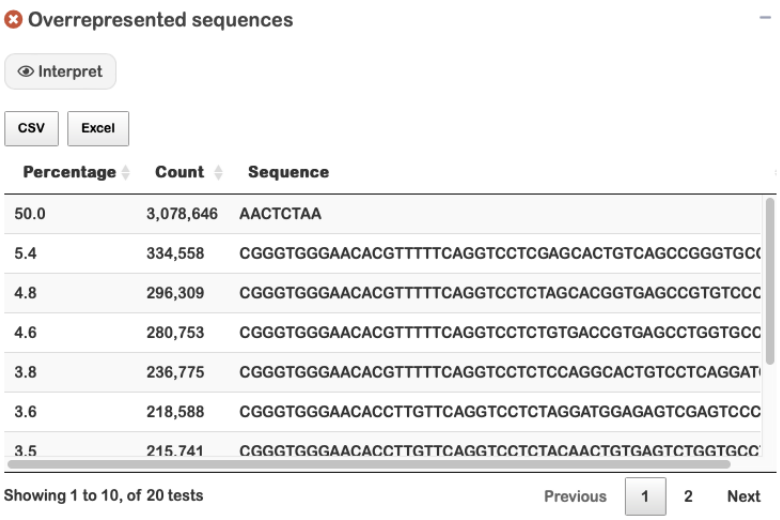


Figura 34: Tabla del módulo de *Overrepresented sequences* del *dashboard*.

12.5. Sección de reportería

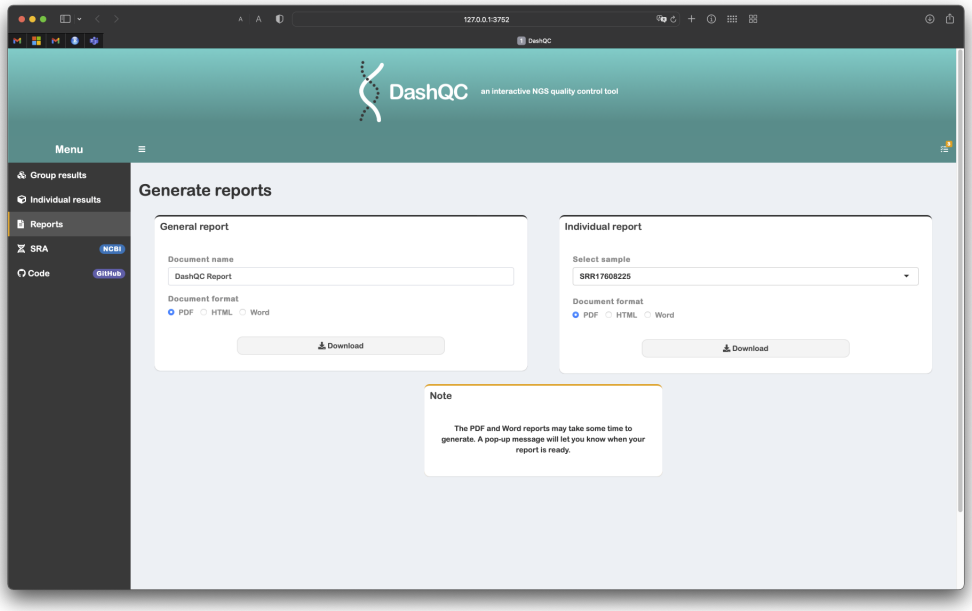
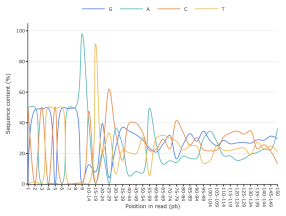


Figura 35: Sección para la generación de reportes general o individual del *dashboard*.

Per base sequence content

This test analyses the proportion of each base position in a file for which each of the four normal DNA bases have been called. The lines in this plot should run parallel with each other. They should not be hugely imbalanced.

Figure 3. SRR17608225 Per base sequence content: FAIL

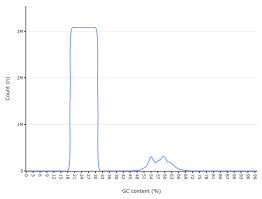


This plot shows the proportion of each base position for which each of the four normal DNA bases has been called. It is expected that there would be little to no difference between the bases of a sequence run, so the lines in this plot should run parallel with each other. Failure raised. A difference between A and T, or G and C greater than 20% was found in al least one position. Strong biases usually indicate there is an overrepresented sequence, which is contaminating your library. There could have also been a systematic problem during the sequencing of the library, the relative amount of each base should not be hugely imbalanced from each other.

Per sequence GC content

This test measures the GC content across the whole length of each sequence in a file and compares it to a modelled normal distribution of GC content.

Figure 4. SRR17608225 Per sequence GC content: FAIL



The test measured the GC content across the whole length of each sequence and compared it to a normal distribution. Failure raised. The sum of the deviations from the normal distribution represents more than 30% of the reads. This unusually shaped distribution could indicate a contaminated library or some other kind of bias in the sample.

The Shapiro–Wilk normality test was published by Samuel Sanford Shapiro and Martin Wilk. The null-hypothesis of this test is that the data is normally distributed. If the p-value is less than the chosen alpha level, then the null hypothesis is rejected and there is evidence that the data tested is not normally distributed.

The results of this test upon the GC content data indicate a p-value of 6.885e-18.

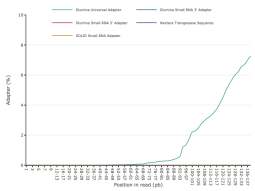
Figura 36: Páginas 4 y 5 del reporte individual.

Adapter Content

This test will do a generic analysis of all of the Kmers in the library to find those which do not have even coverage through the length of your reads. This can find a number of different sources of bias in the library which can include the presence of read-through adapter sequences building up on the end of the sequences.

The plot shows a cumulative percentage count of the proportion of your library which has seen each of the adapter sequences at each position. Once a sequence has been seen in a read it is counted as being present right through to the end of the read so the percentages you see will only increase as the read length goes on.

Figure 8. SRR17608225 Adapter Content: WARN



The test performed an analysis of all of the Kmers in the library to find those which have different sources of bias. The plot shows a cumulative percentage count of the proportion of the library which has seen each of the adapter sequences at each position. Any library where a reasonable proportion of the insert sizes are shorter than the read length will trigger this test. A warning was issued. There was at least one Kmer sequence present in more than 5% of all reads. This doesn't indicate a problem as such, just that the sequences will need to be adapter trimmed before proceeding with any further analysis.

Overrepresented sequences

This test lists all of the sequence which make up more than 0.1% of the total. For each overrepresented sequence the program will look for matches in a database of common contaminants and will report the best hit it finds. Finding a hit doesn't necessarily mean that this is the source of the contamination, but may point you in the right direction.

Table 3. SRR17608225 Overrepresented sequences: FAIL

Percentage	Count	Sequence
50.0	3,078,646	AACTCTAA
5.4	334,558	CGGGTGGGAACACGTTTTTCAGGTCCTCGAGCACTGTCAGCCGGGTGCCT
4.8	296,309	CGGGTGGGAACACGTTTTTCAGGTCCTCTAGCACGGTGAGCCGTGTCCCT
4.6	280,753	CGGGTGGGAACACGTTTTTCAGGTCCTCTGTGACCGTGAGCCTGGTGCCC
3.8	236,775	CGGGTGGGAACACGTTTTTCAGGTCCTCTCCAGGCACTGTCCTCAGGATG
3.6	218,588	CGGGTGGGAACACCTTGTTCCAGGTCCTCTAGGATGGAGAGTCCAGTCCCA
3.5	215,741	CGGGTGGGAACACCTTGTTCAGGTCCTCTACAACTGTGAGTCTGGTGCCT
2.9	176,360	CGGGTGGGAACACCTTGTTCAGGTCCTCTCCAGGCACTGTCCTCAGGATG
2.6	159,128	CGGGTGGGAACACGTTTTTCAGGTCCTCGAGCACCAGGAGCCGCGTGCCT
2.1	127,016	CGGGTGGGAACACGTTTTTCAGGTCCTCCAGTACGGGTCAGCCTAGAGCCT

The test listed all of the sequences which make up more than 0.1% of the total. For each overrepresented sequence the test looked for matches in a database of common contaminants and reported the best hit it found. Failure raised. There was at least one sequence found to represent more than 1% of the total. It could mean that it is a highly biologically significant sequence, or indicates that the library is contaminated or not very diverse.

References

- Babraham, B. (2019). FastQC: A quality control tool for high throughput sequence data. https://www.bioinformatics.babraham.ac.uk/projects/fastqc/
- Illumina. (2021). FASTQ files explained. https://support.illumina.com/bulletins/2016/04/fastq-files-explained.html

Figura 37: Páginas 9 y 10 del reporte individual.

12.6. Pruebas de usuario con el uso del *pipeline*.

12.6.1. Perfil de usuario de prueba

El usuario de prueba objetivo para el *pipeline* idealmente son profesionales y estudiantes de las carreras de Bioinformática, Bioquímica, Biotecnología y Biología. Para realizar las pruebas y la obtención de retroalimentación del funcionamiento de las secciones del *pipeline* se realizó la búsqueda de estudiantes y graduados (de ser posible) en estos campos y también científicos involucrados en el área de investigación genómica.

12.6.2. Retroalimentación

Usuario de Prueba	Puntuación de Utilidad (0-10)					Comentarios del Usuario	Observaciones Propias
	Componente de QC	Informe Automatizado	Dashboard de QC	Sistema de trazabilidad	Producto final		
1	10	10	10	10	10	Piensa que es un sistema necesiado y hubiera querido contar con esta herramienta para varios proyectos, por que le tocó hacerlo manual.	Se vió sorprendida la persona, no creia que fuera posible volver los resultados "interactivos".
2	10	10	10	8	10	Mira dificultad en el sistema de trazabilidad por que cree que algunos laboratorios las personas tienen su propio orden/carpetas etc.	Sería de explorar cual es la forma más comprensible de implementar esto o la más usada.
3	10	10	10	9	10	Cree que el sistema de trazabilidad es bueno pero mejoraria si se pudiera ver en "Drive", o de forma compartida.	Debo investigar este tema de una carpeta compartida, suena interesante el tema de "guardar" en la nube.
4	10	9	10	10	10	Dice que es una herramienta muy útil y da como recomendación asegurarse que el informe se diferencie de los informes viejos que siempre han generado las herramientas de QC.	En efecto, el informe es este el objetivo que busco, va en buen camino.
5	10	10	10	10	10	Me parece una excelente herramienta, sería bueno que hubiera una forma de definir parámetros y/o límites que se puedan definir para avisar si no se cumple una prueba de QC.	Su recomendación es algo que definitivamente hay que considerar en incluir.

Figura 38: Primera recolección de *feedback* y punteo por secciones de *DashQC*.

Usuario de Prueba	Puntuación (0-10)					Retroalimentación importante	Cambios a efectuar
	Componente de QC	Resultados grupales	Resultados individuales	Interpretaciones	Reportes		
1	9	10	10	10	10	*No es muy clara la diferencia de %GC en las muestras (resultados grupales) y para "Per base sequence quality" no se ven muy claros los Box plots.*	Facilitar la visibilidad de diferencias en los %GC de las muestras (lineal). Ajuste de boton auto-scale. Hacer los box-plots más definidos y exactos. Cambio de color? Constraste.
2	10	10	10	8	10	*Las interpretaciones son muy buenas, pero el "Interpret all" solo repite lo mismo que en los botones de "Interpret". Me gustaría comprender también sobre qué tratan las pruebas.*	Implementar algún tipo de guía introductoria sobre cada una de las pruebas.
3	9	10	10	9	10	*No comprendo mucho que es cada prueba, la interpretación ayuda pero algo más falta. Talvés una sección de glosario o algo para "saber más"?*	Con esto, será necesario una página de glosario? o Solo un link a la documentación de FastQC? o La guía introductoria?
4	10	9	10	10	10	*Me gustaría poder indicarle a DashQC el nombre de mi corrida o experimento, para no perder los resultados generales a la hora de guardar un reporte.*	Permitir al usuario definir el nombre del informe grupal. Permitiendo que le apode como desee y lleve su control propio.
5	9	10	10	10	7	*Los reportes automáticos suenan muy bien, los puedo editar y poner en el formato que deseo? Cómo se de que fecha o corrida son?*	Habilitar generación en WORD de los reportes y tomar en cuenta la fecha con un timestamp para los datos dentro del documento.
PUNTEO PROMEDIO	9.4	9.8	10	9.4	9.4		

Figura 39: Segunda recolección de *feedback* y punteo por secciones de *DashQC*.

12.6.3. Pruebas con usuarios

Figura 40: Usuario de prueba utilizando la sección de análisis individual de *DashQC*.

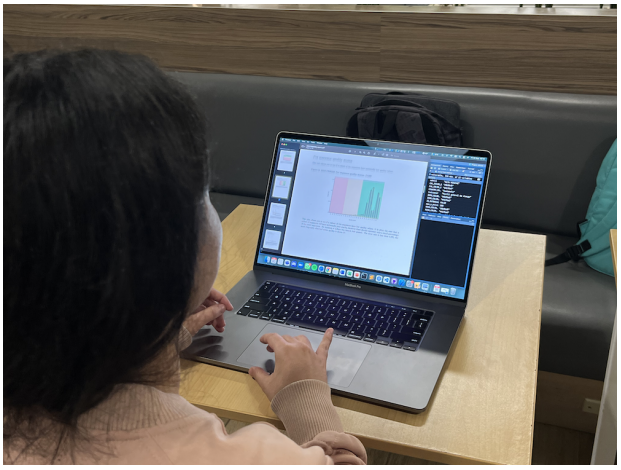


Figura 41: Usuario de prueba generando informes en PDF con el módulo de reportería de *DashQC*.

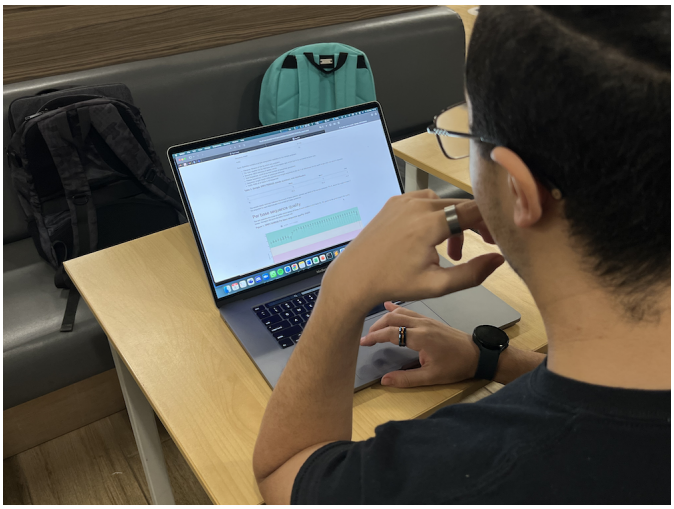


Figura 42: Usuario de prueba analizando el reporte individual en HTML de *DashQC*.

Figura 43: Usuario de prueba comparando los resultados en PDF con una muestra individual en el *dashboard*.

Figura 44: Usuario de prueba utilizando la sección de resultados individual y las interpretaciones de *DashQC*.

12.7. Simulación de visión de colores según tipos principales de ceguera de color.

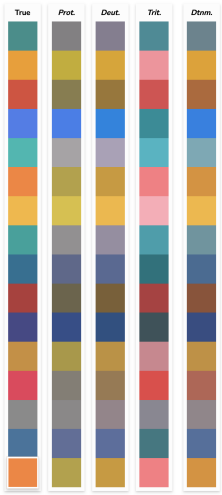


Figura 45: Comparación de paleta de colores del *dashboard* con los tipos de ceguera de color: protanopía, deuteranopía, tritanopía y deuteranomalía.

(Nichols, 2022)

12.8. Prueba de normalidad de Shapiro–Wilk

La prueba de normalidad de Shapiro-Wilk fue publicada por Alva y Estrada, 2009. La hipótesis nula de esta prueba es que los datos se distribuyen normalmente. Si el valor-p es inferior al nivel alfa elegido, se rechaza la hipótesis nula y hay evidencia de que los datos analizados no están distribuidos normalmente.

CAPÍTULO 13

Glosario

ADN: Ácido desoxirribonucleico, contiene las instrucciones genéticas de todos los organismos vivos. 1, 9

ARN: Ácido ribonucleico, molécula presente en todas las células vivas. A diferencia del ADN, está formado por una única cadena. 1, 9

base call: El proceso de inferir una base (A,T,G,C) a partir del cuádruple de intensidad. 11

bcl2fastq: Programa hecho por Illumina para convertir archivos de formato BCL a formato FASTQ. 11

CLI: Interfaz de línea de comandos. Utiliza una interfaz de línea de comandos para recibir comandos de un usuario en forma de líneas de texto. 12, 60

IDE: Entorno de Desarrollo Integrado. 24

NCBI: National Center for Biotechnology Information. 13

pb: Unidad que consta de dos nucleótidos unidos entre sí por enlaces de hidrógeno.. 19

PCR: Reacción en cadena de la polimerasa, técnica de laboratorio que permite la producción (amplificación) rápida de millones a miles de millones de un segmento específico de ADN. 18, 21, 46

QA: Del inglés *Quality Assurance*, es el conjunto de actividades planificadas y sistemáticas aplicadas en un sistema de gestión de la calidad para que los requisitos de calidad de un producto o servicio sean cumplidos. 28

QC: Del inglés *Quality control*, control de calidad. 7, 29, 33, 37, 39, 46, 47

SRA: Archivo de lectura de secuencias. 1, 13, 30