

Ensamblaje y Anotación de Genomas Bacterianos

Libro de Bioinformática

Héctor Fabio Espitia Navarro
Alvaro Jairo Pazos Moncayo



Editorial
Universidad de Nariño

editorial
Universidad de **Nariño**

Ensamblaje y Anotación de Genomas Bacterianos

Libro de Bioinformática

Ensamblaje y Anotación de Genomas Bacterianos

Libro de Bioinformática

Héctor Fabio Espitia Navarro
Alvaro Jairo Pazos Moncayo

editorial
Universidad de **Nariño**

Espitia Navarro, Héctor Fabio

Ensamblaje y anotación de genomas bacterianos : libro de bioinformática / Héctor Fabio Espitia Navarro, Álvaro Jairo Pazos Moncayo—1ª. Ed.-- San Juan de Pasto : Editorial Universidad de Nariño, 2026.

152 páginas : ilustraciones, gráficas, tablas

Incluye Referencias bibliográficas p. 113 – 118, glosario p. 119 - 127 y reseña de los autores p. 150.

ISBN: 978-628-7864-37-5 Impreso

ISBN: 978-628-7864-38-2 Digital

1. Genomas bacterianos—Secuenciación 2. Ensamblaje de Novo 3. Ensamblaje de genomas individuales—Software 4. Ensamblaje y anotación—Plataformas Web 5. Conda—Instalación de paquetes I. Pazos Moncayo, Álvaro Jairo

572.80285 E777 – SCDD-Ed. 23



SECCIÓN DE BIBLIOTECA

Ensamblaje y anotación de genomas bacterianos

Libro de Bioinformática

© Editorial Universidad de Nariño

© Héctor Fabio Espitia Navarro

Álvaro Jairo Pazos Moncayo

ISBN impreso: 978-628-7864-37-5

ISBN digital: 978-628-7864-38-2

DOI: <https://doi.org/10.22267/lib.udn.050>

Primera edición

Corrección de estilo: Luz Elida Vera Hernández

Diseño de cubiertas: María Camila Cerón Cerón

Imagen de portada: Tetiana Lazunova.

Imagen utilizada bajo licencia de iStock

Diagramación: Héctor Fabio Espitia Navarro y Nathaly Rivadeneira

Fecha de publicación: marzo de 2026

San Juan de Pasto - Nariño - Colombia

Prohibida la reproducción total o parcial, por cualquier medio o con cualquier propósito, sin la autorización escrita de sus Autores o de la Editorial Universidad de Nariño

Declaración de responsabilidad

Las opiniones, interpretaciones y conclusiones expresadas en esta obra son exclusivamente responsabilidad de los autores. No reflejan necesariamente la posición oficial, científica o institucional de las organizaciones académicas, gubernamentales, clínicas o privadas con las que los autores estén afiliados o hayan colaborado.

Este libro presenta conceptos, metodologías y herramientas en bioinformática con fines educativos y de divulgación científica. Aunque se ha procurado la mayor precisión posible, los contenidos no deben considerarse como asesoramiento técnico o clínico definitivo, y su aplicación debe ser evaluada en función del contexto específico y bajo supervisión profesional.

Contenido

Introducción	11
Objetivo general.....	11
Metodología.....	13
Disponibilidad de código	14
Prerrequisitos	14
Convenciones del libro	14
Tipografía.....	14
Bloques especiales.....	14
Capítulo 1. Preparación del entorno de trabajo	17
1.1. Estructura de directorios.....	17
1.2. Ambiente Conda e instalación de programas	19
1.2.1. Instalación Conda: Miniforge.....	19
1.2.2. Creación del ambiente Conda	23
1.2.3. Instalación de paquetes	23
1.3. Resumen	25
Capítulo 2. Obtención de datos	26
2.1. Software requerido	26
2.2. Datos.....	26
2.3. Descarga	27
2.4. Resumen	30
Capítulo 3. Control de calidad y limpieza de datos	31
3.1. Software requerido	32
3.2. Determinación de calidad.....	32
3.2.1. Reporte de calidad FastQC	33
3.2.2. Unificación de reportes de calidad.....	35
3.2.3. Interpretación de resultados.....	36
3.3. Limpieza	37
3.4. Verificación de calidad después de la limpieza	43
3.5. Resumen	44

Capítulo 4. Ensamblaje 45

4.1. Software requerido	46
4.2. Estrategias de ensamblaje genómico	46
4.2.1. Ensamblaje de genomas individuales	46
4.2.2. Ensamblaje metagenómico.....	47
4.2.3. Ensamblaje híbrido	47
4.3. Ejecución del ensamblaje con SPAdes	47
4.3.1. Ensamblaje de la muestra SRR22388518	48
4.3.2. Interpretación de los resultados de SPAdes.....	49
4.3.3. Ensamblaje de las muestras restantes.....	53
4.4. Resumen	55

Capítulo 5. Control de calidad de ensamblajes 56

5.1. Software requerido	56
5.2. Determinación de calidad	57
5.2.1. Descarga del genoma de referencia	57
5.2.2. Ejecución de QUAST	58
5.3. Evaluación de calidad con QUAST.....	60
5.4. Evaluación de completitud con BUSCO	64
5.4.1. Ejecución de BUSCO en la muestra SRR22388518	65
5.4.2. Interpretación de la salida de BUSCO	67
5.4.3. Ejecución de BUSCO en las muestras restantes.....	68
5.4.4. Unificación de reportes BUSCO.....	68
5.4.5. Interpretación de los resultados de BUSCO.....	69
5.5. Análisis de k-meros con Jellyfish y KAT.....	71
5.5.1. Análisis de la muestra SRR22388518 y su ensamblaje.....	72
5.5.2. Análisis de k-meros en las muestras restantes.....	81
5.5.3. Interpretación de los resultados de KAT.....	83
5.6. Corrección y mejoramiento de ensamblajes	87
5.6.1. Descontaminación de datos crudos.....	87
5.6.2. Descontaminación del ensamblaje.....	88
5.6.3. Ensamblaje posterior de contigs o scaffolds.....	88
5.6.4. Pulido de ensamblajes.....	89
5.7. Resumen	89

Capítulo 6. Anotación funcional 91

6.1. Software requerido	92
6.2. Ejecución de Prokka	92
6.3. Interpretación de la salida de Prokka	94
6.3.1. Archivo de anotación GFF.....	96
6.4. Visualización de los resultados de anotación	98
6.4.1. Herramientas de visualización.....	98
6.4.2. Comparación de herramientas	102
6.5. Resumen	102

Capítulo 7. Plataformas Web para ensamblaje y anotación 103

7.1. Usuarios objetivo y beneficios.....	103
7.1.1. Perfiles de usuarios que se benefician.....	103
7.1.2. Ventajas principales.....	104
7.1.3. Limitaciones y consideraciones.....	104
7.2. Contextos de uso apropiados.....	104
7.3. Plataformas recomendadas	105
7.3.1. BV-BRC	105
7.3.2. KBase.....	105
7.3.3. Galaxy	106
7.3.4. NCBI Genome Workbench.....	106
7.3.5. RAST	107
7.4. Comparativa de plataformas	107

Capítulo 8. Conclusiones 109

8.1. Conclusiones generales	109
8.2. Limitaciones.....	110
8.3. Trabajos futuros	111

Referencias	113
-------------------	-----

Glosario	119
----------------	-----

Índice de Figuras	128
-------------------------	-----

Índice de Tablas	129
------------------------	-----

Apéndices 130

Apéndice A. Problemas comunes y soluciones sugeridas 131

A.1. Problemas comunes con Conda.....	131
A.1.1. Conflictos de dependencias	131
A.1.2. Entorno no encontrado	132
A.1.3. Problemas de permisos	132
A.1.4. Comandos no encontrados después de la instalación	132
A.1.5. Lentitud en la resolución de dependencias	133
A.2. Problemas comunes durante la descarga de datos	133
A.2.1. Fallas en la descarga desde SRA.....	133
A.2.2. Archivos FASTQ incompletos o corruptos	133
A.2.3. Problemas con identificadores SRA.....	134
A.3. Problemas comunes durante el control de calidad de los datos..	134
A.3.1. FastQC no genera reportes.....	134
A.3.2. Interpretación de resultados FastQC	135
A.3.3. Errores en fastp durante el preprocesamiento	135
A.4. Problemas comunes durante el ensamblaje.....	135
A.4.1. Errores de memoria insuficiente	135
A.4.2. Ensamblaje fragmentado (muchos contigs pequeños).....	136
A.4.3. Archivos de salida vacíos o incompletos.....	136

A.4.4. Errores de formato de archivos.....	137
A.4.5. SPAdes se detiene inesperadamente.....	137
A.4.6. Advertencias sobre cobertura irregular	137
A.4.7. Problemas con paired-end reads	138
A.5. Problemas comunes durante la evaluación de calidad de ensamblajes.....	138
A.5.1. QUAST no encuentra ensamblajes.....	138
A.5.2. BUSCO no encuentra genes marcadores	138
A.5.3. Errores en descarga con datasets (NCBI).....	139
A.5.4. KAT produce gráficos vacíos	139
A.6. Problemas comunes durante la anotación.....	139
A.6.1. Prokka falla en la predicción de genes.....	139
A.6.2. Errores de dependencias en Prokka.....	140
A.6.3. Problemas con archivos de salida de Prokka.....	140
Apéndice B. Ensambladores	141
Apéndice C. Contenido y uso del archivo spades.log	143
C.1. Contenido del archivo	143
C.2. Diagnóstico de problemas	143
Apéndice D. Ensamblaje de datos metagenómicos	145
D.1. Diferencias entre ensamblaje genómico y metagenómico.....	145
D.2. Flujo de trabajo típico	147
D.3. Herramientas especializadas para ensamblaje metagenómico ..	148
D.4. Consideraciones técnicas importantes	149
Acerca de los autores	150

Introducción



La secuenciación de genomas bacterianos completos se ha consolidado como una herramienta fundamental en microbiología moderna y epidemiología molecular. Los datos de secuenciación de nueva generación requieren procesamiento bioinformático especializado para transformar millones de lecturas cortas en genomas ensamblados y funcionalmente anotados. Utilizando muestras de *Helicobacter pylori* como caso de estudio, este libro proporciona las competencias prácticas necesarias para generar genomas de calidad apropiada para análisis comparativos, estudios epidemiológicos y aplicaciones en salud pública.

La obra se organiza en ocho capítulos principales que cubren desde los fundamentos hasta las aplicaciones avanzadas del procesamiento bioinformático: configuración del ambiente computacional (Capítulo 1), obtención de datos desde repositorios públicos (Capítulo 2), control de calidad y limpieza de secuencias (Capítulo 3), ensamblaje *de novo* (Capítulo 4), evaluación integral de la calidad de ensamblajes (Capítulo 5), anotación funcional automatizada (Capítulo 6), plataformas web alternativas para análisis (Capítulo 7), y conclusiones (Capítulo 8). Adicionalmente, se incluyen un glosario de términos especializados y apéndices con guías de resolución de problemas, comparación de ensambladores alternativos y recursos complementarios. Cada capítulo combina fundamentos teóricos con procedimientos paso a paso, comandos específicos, interpretación de resultados y buenas prácticas para facilitar la implementación exitosa del protocolo completo.

Objetivo general

Desarrollar las competencias bioinformáticas necesarias para realizar el ensamblaje *de novo* y la anotación funcional de genomas bacterianos a partir de datos de secuenciación de genoma completo, utilizando herramientas computacionales especializadas y metodologías estandarizadas. Este libro busca capacitar al lector en la implementación de un flujo de trabajo completo que incluye desde la obtención y preparación de datos de secuenciación hasta la generación de genomas anotados de buena calidad para análisis posteriores en microbiología, epidemiología molecular y genómica comparativa.

Metodología

La metodología empleada en este libro sigue un enfoque práctico y secuencial que cubre las etapas principales del procesamiento bioinformático de datos de secuenciación para ensamblar y anotar genomas bacterianos. El flujo de trabajo consta de seis fases principales:

- 1. Configuración del ambiente computacional:** se establece un entorno de trabajo reproducible mediante la instalación y configuración de herramientas bioinformáticas utilizando el gestor de paquetes Conda/Mamba, garantizando la disponibilidad de todas las dependencias necesarias.
- 2. Obtención de datos:** se descarga un conjunto de cuatro muestras de secuenciación de genoma completo de *Helicobacter pylori* desde la base de datos pública SRA (del inglés *Sequence Read Archive*) del NCBI (del inglés *National Center for Biotechnology Information*), utilizando *SRA Tools* para garantizar la integridad de los datos.
- 3. Control de calidad y preprocesamiento:** se evalúa la calidad de las secuencias crudas mediante FastQC y se aplican procedimientos de limpieza y filtrado utilizando fastp, eliminando adaptadores, secuencias de baja calidad y artefactos de secuenciación que puedan comprometer la calidad del ensamblaje.
- 4. Ensamblaje *de novo*:** se implementa la reconstrucción de los genomas utilizando SPAdes, un ensamblador especializado en genomas bacterianos que emplea algoritmos basados en grafos de De Bruijn para ensamblar las lecturas cortas en secuencias contiguas.
- 5. Evaluación de calidad de ensamblajes:** se realiza una evaluación de la calidad de los ensamblajes generados mediante múltiples herramientas: QUAST para métricas estructurales, BUSCO para completitud biológica, y análisis de *k*-meros con Jellyfish y KAT para validación comparativa, utilizando un genoma de referencia de *H. pylori* como estándar.
- 6. Anotación funcional:** se identifican y caracterizan las características genómicas utilizando Prokka, que implementa un flujo de trabajo automatizado para la predicción de genes y la asignación de funciones biológicas mediante búsquedas de homología en bases de datos especializadas.

Cada fase del flujo de trabajo (Figura 1) se documenta con comandos específicos, parámetros detallados y criterios de interpretación de resultados. La metodología enfatiza las buenas prácticas en bioinformática, incluyendo la reproducibilidad, el control de calidad y el análisis de resultados. Adicionalmente, se incluye un apéndice de resolución de problemas comunes para facilitar la implementación práctica del protocolo.

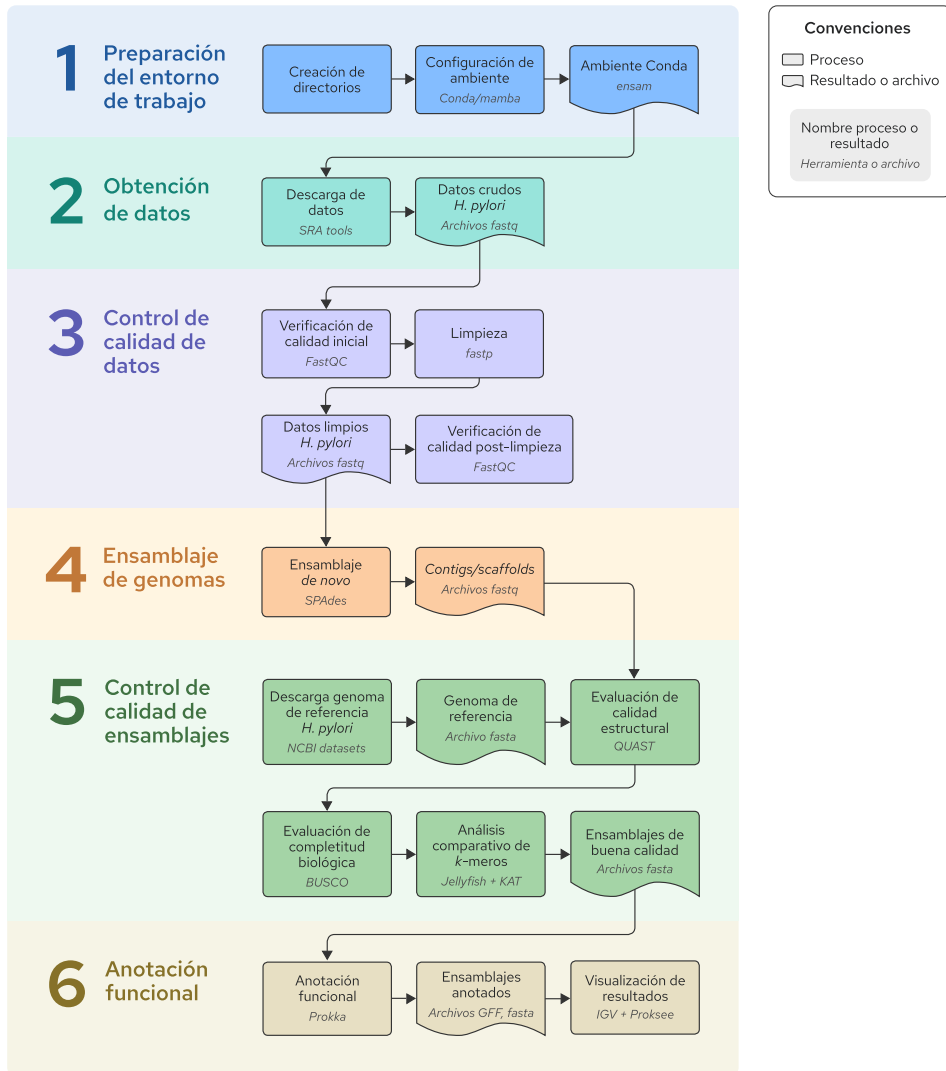


Figura 1. Diagrama de flujo de trabajo para el ensamblaje y anotación de genomas bacterianos. Cada sección identificada con un color distinto representa una fase principal del flujo de trabajo.

Disponibilidad de código

Si bien el libro está diseñado para que el lector ejecute las tareas del flujo de trabajo paso a paso siguiendo las recomendaciones adjuntas, todos los comandos presentes en la obra están disponibles en el repositorio público de GitHub [hspitia/ensamblaje-genomas-bacterianos](https://github.com/hspitia/ensamblaje-genomas-bacterianos)¹. El repositorio incluye un *script* Bash para cada uno de los capítulos con los correspondientes comandos presentados, y un archivo para la creación del ambiente de software con los programas requeridos para el desarrollo del proyecto.

Prerrequisitos

El flujo de trabajo descrito en este libro está diseñado para ser ejecutado en el sistema Operativo (SO) Linux. Para desarrollar esta obra se usó la distribución Ubuntu Linux 20.04. Aunque no se requiere ser un usuario Linux avanzado, será más fácil seguir el flujo de trabajo descrito si está familiarizado con el uso de la línea de comandos en el SO Linux. Puede aprender más sobre Linux y la línea de comandos en el manual “Introducción a Linux para Bioinformática”².

Convenciones del libro

Tipografía

Las siguientes convenciones tipográficas son usadas en este libro:

- *Texto en cursiva*:
 - Palabras escritas en idiomas diferentes del español (p. ej. *prompt*, o *forward*)
 - Expresiones aritméticas (p. ej. $Q \geq 20$).
- **Texto en fuente monoespaciada**:
 - Nombres de archivos (p. ej. `/ruta/al/archivo.txt`)
 - Nombres especiales (p. ej. ambientes conda: `base`)
 - Teclas (p. ej. `ENTER`)
 - Nombre de comandos (p. ej. `ls`).

¹Repositorio de código en GitHub: <https://github.com/hspitia/ensamblaje-genomas-bacterianos>

²Manual en línea: <https://hspitia.github.io/linux-bioinfo/>

- Bloques de órdenes de línea de comandos y su resultado. Por ejemplo:

Orden o comando:

```
# Línea de comentario. Esta línea no se ejecuta  
  
ls -l ~/ejemplo
```

Resultado o salida del comando anterior:

```
total 0  
-rw-rw-r-- 1 hspitia hspitia 0 Dec 17 14:25 archivo_1.txt  
-rw-rw-r-- 1 hspitia hspitia 0 Dec 17 14:25 archivo_2.txt  
-rw-rw-r-- 1 hspitia hspitia 0 Dec 17 14:25 archivo_3.txt  
-rw-rw-r-- 1 hspitia hspitia 0 Dec 17 14:25 archivo_4.txt
```

Nota sobre bloques de órdenes de línea de comandos:

```
# Si un texto en un bloque de órdenes (línea de comandos) o en  
└─ su salida es muy largo (como este texto) y no cabe en el  
└─ bloque, posiblemente verá indicaciones de que el texto  
└─ continúa en la siguiente línea (↔ al inicio), e incluso  
└─ que fue cortado si no había ningún espacio en blanco (| al  
└─ final) como aquí: /ruta/de/archivo/excesivamente/larga/pa  
└─ ra/el/espacio/de/pagina/disponible.txt
```

- **Texto en negrita:** características principales o elementos destacados en enumeraciones y listas descriptivas.
- **Hipervínculos (*links*):** enlaces a partes de este documento (internos) o a sitios Web (externos).

Bloques especiales

Los textos con información de especial atención se presentan en bloques (cajas) especiales que se diferencian por su color. Existen cinco tipos de bloques especiales:

i Nota

Las notas proporcionan información complementaria, contexto adicional o definiciones que amplían el contenido principal sin ser esenciales para el flujo de trabajo.

 Importante

La información importante incluye recordatorios críticos, instrucciones esenciales y prerequisites necesarios para ejecutar correctamente el flujo de trabajo.

 Consejo

Los consejos proporcionan recomendaciones prácticas, opciones de comandos adicionales, y sugerencias que pueden mejorar la eficiencia o comprensión del proceso.

 Precaución

Las precauciones alertan sobre aspectos que requieren atención especial o consideraciones importantes antes de proceder con una tarea específica.

 Advertencia

Las advertencias señalan errores comunes, problemas potenciales o aspectos que pueden causar fallos en la ejecución de comandos si no se consideran adecuadamente.



Capítulo 1.

Preparación del entorno de trabajo

Una buena organización de directorios y una gestión efectiva de software en ambientes virtuales son imprescindibles para facilitar el trabajo fluido y garantizar la reproducibilidad en proyectos bioinformáticos.

Este capítulo establece los requerimientos técnicos de organización y software necesarios para el proyecto de ensamblaje y anotación de genomas bacterianos. Se aborda la creación de una estructura organizacional de directorios, la instalación y configuración de `Conda Miniforge` para la gestión de ambientes virtuales, y la instalación del software bioinformático requerido en un ambiente aislado y reproducible.

1.1. Estructura de directorios

Una práctica recomendable en bioinformática es mantener una estructura organizada de los archivos y directorios que componen un proyecto. Esto facilita la localización de archivos, la reproducibilidad de los análisis y la colaboración con otros investigadores. Siguiendo esta recomendación, se prefiere tener los proyectos centralizados en el sistema, por ejemplo, en el directorio `proyectos` dentro del directorio personal (*home*) del usuario. Aunque centralizados, los proyectos deben ser independientes uno del otro, por lo que se recomienda que cada proyecto se encuentre en su propio directorio único y exclusivo, y en lo posible, con su propia instalación de programas.

Para comenzar, primero se crea el directorio para general para todos los proyectos. Abra una terminal y ejecute las siguientes órdenes:

```
cd $HOME  
mkdir proyectos
```

! Importante

Se asume que usted debe ejecutar en la terminal todos los comandos mostrados a partir de este punto.

Ahora, se creará la estructura para nuestro proyecto de ensamblaje dentro del directorio `proyectos`. En la estructura se contemplan subdirectorios de datos, calidad y resultados:

```
mkdir -p proyectos/ensamblaje/{datos,resultados,scripts}
mkdir -p proyectos/ensamblaje/datos/qc/fastqc
mkdir -p proyectos/ensamblaje/resultados/{00_datos_limpios,01_ensamblaje,
  ↪ e,02_ensamblaje_qc,03_annotacion}
mkdir -p proyectos/ensamblaje/resultados/02_ensamblaje_qc/busco
mkdir -p proyectos/ensamblaje/resultados/02_ensamblaje_qc/kmers
mkdir -p
  ↪ proyectos/ensamblaje/resultados/00_datos_limpios/qc/{fastp,fastqc}
```

La estructura final del directorio del proyecto se puede ver con el comando `tree`:

```
tree -L 4 --charset HTML proyectos/ensamblaje
```

```
proyectos/ensamblaje
|-- datos
|   |-- qc
|       |-- fastqc
|-- resultados
|   |-- 00_datos_limpios
|       |-- qc
|           |-- fastp
|           |-- fastqc
|   |-- 01_ensamblaje
|   |-- 02_ensamblaje_qc
|       |-- busco
|       |-- kmers
|   |-- 03_annotacion
|-- scripts
```

```
14 directories, 0 files
```

1.2. Ambiente Conda e instalación de programas

Una manera eficiente y rápida para instalar todos los programas necesarios para el ensamblaje y anotación de los genomas, consiste en usar Conda¹, un ecosistema de libre distribución para administrar ambientes virtuales de software e instalación de programas o paquetes (Nota 1).

Consejo

Consulte el Apéndice A para resolver problemas y errores comunes que podrían presentarse durante el desarrollo de este proyecto, incluyendo aquellos relacionados con la gestión de ambientes de software con Conda.

1.2.1. Instalación Conda: Miniforge

Se instalará una versión liviana de la comunidad `conda-forge`² llamada Miniforge³, que además cuenta con el gestor de paquetes `mamba`, el cual es más rápido y eficiente que el usado por el ecosistema Conda que lleva el mismo nombre (`conda`).

Nota 1: Ambientes virtuales, `conda`, Anaconda, Miniforge y `mamba`

Un ambiente virtual de software es un espacio en disco en el que se instalan programas de manera aislada del resto de aplicaciones instaladas en el computador. Tiene la ventaja de que allí se instalan todos los programas necesarios para un proyecto, incluidas versiones específicas, sin interferir con otras instalaciones o versiones de otros proyectos o del sistema.

Conda es un ecosistema de libre distribución (*open-source*) de administración de paquetes y bibliotecas multiplataforma. Este ecosistema usa un programa gestor de ambientes virtuales e instalador de paquetes con el mismo nombre, `conda`. Siguiendo el modelo Conda de administración de ambientes y programas, existen diferentes conjuntos de paquetes llamados distribuciones. Entre estas distribuciones, Anaconda⁴, desarrollada por la empresa Anaconda Inc., es tal vez la más conocida. Anaconda incluye un conjunto de herramientas y utilidades para facilitar el trabajo en campos como la ciencia de datos, inteligencia artificial, aprendizaje automático (*machine learning*) y análisis estadístico, y además, usa `conda` como gestor de ambientes y paquetes.

¹Sitio Web de Conda (en inglés): <https://conda.org/>

²Sitio Web de la comunidad conda-forge (en inglés): <https://conda-forge.org/>

³Miniforge (GitHub): <https://github.com/conda-forge/miniforge>.

Dado que el proyecto desarrollado en esta obra no requiere ningún programa incluido por defecto por Anaconda, se prefiere usar una distribución llamada Miniforge, desarrollada y mantenida por la comunidad de desarrolladores conda-forge. Miniforge cuenta con un conjunto mínimo de utilidades básicas sobre el cual se pueden agregar paquetes adicionales según sea necesario. Además de ser más liviana que Anaconda (512 MB vs. ~3 GB), Miniforge usa por defecto la herramienta `mamba` como gestor de ambientes virtuales e instalador de paquetes. En esencia, `mamba` es el equivalente del programa `conda`, pero mucho más rápido, gracias a su diseño y construcción más eficientes, que explota el uso de múltiples núcleos de procesador en paralelo.

Primero, se descarga el instalador:

```
wget "https://github.com/conda-forge/miniforge/releases/latest/download,  
  _/Miniforge3-Linux-x86_64.sh"
```

Consejo

Toda la información acerca de miniforge y otros instaladores están disponibles en <https://github.com/conda-forge/miniforge>

Ahora, se puede dar inicio a la instalación:

```
# Adicionar permiso de ejecución  
chmod +x Miniforge3-Linux-x86_64.sh  
  
# Ejecutar el instalador  
./Miniforge3-Linux-x86_64.sh
```

Al ejecutar el anterior comando, verá un mensaje de los términos de la licencia como este:

```
Welcome to Miniforge3 25.3.1-0
```

```
In order to continue the installation process, please review the license  
agreement.
```

```
Please, press ENTER to continue
```

```
>>>
```

Presione la tecla `ENTER` varias veces hasta que el instalador solicite aceptar los términos de la licencia:

⁴Anaconda: <https://www.anaconda.com/>

```
Do you accept the license terms? [yes|no]
[no] >>>
```

Responda **yes**

A continuación, el instalador solicitará el lugar donde instalará Miniforge. Puede confirmar la ubicación por defecto (en su directorio *home*: `~/miniforge3`) presionando ENTER:

```
Miniforge3 will now be installed into this location:
/home/hector/miniforge3
```

- Press ENTER to confirm the location
- Press CTRL-C to abort the installation
- Or specify a different location below

```
[/home/hector/miniforge3] >>>
```

Una vez el instalador finaliza la copia de archivos, preguntará si desea inicializar Miniforge:

```
...
Transaction finished

installation finished.
Do you wish to update your shell profile to automatically initialize
  ~ conda?
This will activate conda on startup and change the command prompt when
  ~ activated.
If you'd prefer that conda's base environment not be activated on startup,
  run the following command when conda is activated:

conda config --set auto_activate_base false

You can undo this by running `conda init --reverse $SHELL`? [yes|no]
[no] >>>
```

Responda **yes**. Después, verá un mensaje que indica los cambios realizados en los archivos de configuración para inicializar Mambaforge:

```
...
no change    /home/hector/mambaforge/condabin/conda
no change    /home/hector/mambaforge/bin/conda
no change    /home/hector/mambaforge/bin/conda-env
no change    /home/hector/mambaforge/bin/activate
no change    /home/hector/mambaforge/bin/deactivate
no change    /home/hector/mambaforge/etc/profile.d/conda.sh
no change    /home/hector/mambaforge/etc/fish/conf.d/conda.fish
```

```
no change      /home/hector/mambaforge/shell/condabin/Conda.psml
no change      /home/hector/mambaforge/shell/condabin/conda-hook.ps1
no change      /home/hector/mambaforge/lib/python3.10/site-packages/xontr_
└─ ib/conda.xsh
no change      /home/hector/mambaforge/etc/profile.d/conda.csh
modified       /home/hector/.bashrc
```

==> For changes to take effect, close and re-open your current shell. <==

Running `shell init`, which:

- modifies RC file: "/home/hector/.config/nushell/config.nu"
- generates config for root prefix: "/home/hector/miniforge3"
- sets mamba executable to: "/home/hector/miniforge3/bin/mamba"

The following has been added in your

- └─ "/home/hector/.config/nushell/config.nu" file

```
# >>> mamba initialize >>>
```

```
# !! Contents within this block are managed by 'mamba shell init' !!
```

```
...
```

```
# <<< mamba initialize <<<
```

Thank you for installing Miniforge3!

Para que la instalación tome efecto, se debe cerrar y abrir nuevamente la terminal.

Muy seguramente, cuando se abra de nuevo la terminal, notará que el *prompt* (indicador de comandos) luce diferente; algo similar a:

```
(base) hector@Ubuntu:~$
```

El cambio se debe a que Conda usa el *prompt* como medio para informar el ambiente que se encuentra activo (en uso) actualmente, mostrándolo entre paréntesis al inicio de la línea. En el ejemplo anterior aparece la palabra *base*; este es el nombre del ambiente “base” por defecto de Conda.

Ahora, se requiere inicializar *mamba* en el sistema:

```
eval "$(mamba shell hook --shell bash)"
```

1.2.2. Creación del ambiente Conda

El siguiente paso consiste en crear un ambiente Conda en donde se instalarán todas los programas requeridos. Un ambiente es un directorio que contiene una colección específica de paquetes de Conda (con una versión concreta) que se han instalado.

Para crear el ambiente, se debe ejecutar las siguientes órdenes:

```
mamba create -n ensam
```

```
Empty environment created at prefix: /home/hector/miniforge3/envs/ensam
```

Como se pudo observar, **mamba** despliega un mensaje de confirmación de creación del ambiente.

! Importante

Para poder usar el ambiente creado, siempre se debe activarlo primero:

```
mamba activate ambiente
```

donde **ambiente** es el nombre del ambiente creado.

Cuando ya no se necesita el ambiente, se debe desactivar:

```
mamba deactivate
```

1.2.3. Instalación de paquetes

Ahora, se procede a instalar los paquetes requeridos. Primero se debe activar el ambiente **ensam**:

```
mamba activate ensam
```

i Nota 2

Cuando se activa correctamente, se podrá ver el nombre del ambiente recién activado en el *prompt*, en este caso **ensam** en lugar de **base**:

```
(ensam) hector@Ubuntu:~$
```

Luego, deben instalarse los paquetes ejecutando el siguiente comando:

```
mamba install -c bioconda busco=5.5.0 fastp=0.23.2 fastqc=0.12.1  
└─ igv=2.9.0 multiqc=1.1 jellyfish=2.2.10 kat=2.4.2 \  
  4 ncbi-datasetsli=18.4.1 prokka=1.14.6 quast=5.2.0 spades=3.15.5  
└─ sra-tools=3.0.3 pigz csvtk wget java-jdk
```

Posteriormente, se verán algunos mensajes y barras de progreso mientras **mamba** descarga los listados de paquetes y busca los paquetes que se ordenaron instalar. Al final **mamba** mostrará un resumen del total de paquetes a instalar y el tamaño total a descargar, y además, solicitará que confirme la instalación de los paquetes:

```
...  
Summary:  
  
Install: 480 packages  
  
Total d485load: 2GB
```

Confirm changes: [Y/n]

Se debe hacer la confirmación escribiendo **Y**, después se presiona **ENTER**. De nuevo se verá una serie de tareas completadas y barras de progreso mientras la instalación transcurre.

Aunque se ordenó instalar 16 paquetes, cada uno de ellos requiere múltiples dependencias, es decir, paquetes adicionales para funcionar. Aquí es donde se facilitan las cosas: **mamba** (o **conda**) se encarga de revisar las dependencias de cada paquete, instalarlas y configurarlas automáticamente para que cada programa funcione adecuadamente.

Al finalizar la instalación se verá un mensaje semejante a este:

```
...  
  
To be able to use those, please run  
quast-download-gridss  
quast-download-silva  
quast-download-busco
```

```
Linking r-ggplot2-3.5.1-r42hc72bb7e_0
Linking perl-package-deprecationmanager-0.17-pl5321hdfd78af_1
Linking perl-devel-overloadinfo-0.007-pl5321hdfd78af_0
Linking perl-class-load-0.25-pl5321hdfd78af_1
Linking prokka-1.14.6-pl5321hdfd78af_5
Linking perl-class-load-xs-0.10-pl5321h9f5acd7_6
Linking perl-moose-2.2202-pl5321hec16e2b_0
Linking perl-moo-2.005004-pl5321hdfd78af_0
Linking perl-parallel-forkmanager-2.04-pl5321hdfd78af_0
Linking augustus-3.5.0-pl5321hf46c7bb_1
Linking busco-5.5.0-pyhdfd78af_0
```

```
Transaction finished
```

Ahora se cuenta con la instalación del software requerido para el análisis de los datos.

1.3. Resumen

Una preparación adecuada del entorno de trabajo es fundamental para garantizar la reproducibilidad y eficiencia de los análisis bioinformáticos. El uso de herramientas especializadas para la gestión de software y la organización de proyectos constituye la base para el desarrollo exitoso de cualquier proyecto de análisis genómico.

En este capítulo se estableció un entorno de trabajo organizado siguiendo las mejores prácticas de bioinformática. La preparación incluyó la creación de una estructura de directorios sistemática, la instalación de Conda Miniforge como herramienta de gestión de ambientes virtuales y paquetes, y la configuración de un ambiente específico para el proyecto con todo el software requerido. Adicionalmente, se adquirieron conocimientos fundamentales sobre los conceptos relacionados con ambientes virtuales y las distintas herramientas del ecosistema Conda.



Capítulo 2.

Obtención de datos

La obtención de datos es el primer paso en cualquier proyecto de análisis genómico. En este capítulo se describen los pasos necesarios para descargar datos de secuenciación de genoma completo desde la base de datos pública SRA (del inglés *Sequence Read Archive*) usando la herramienta SRA Tools.

2.1. Software requerido

Tabla 1. Software requerido para la obtención de datos.

Software	Versión	Descripción	Referencia
SRA Tools	3.0.3	Colección de herramientas para descargar datos desde la base de datos SRA (Nota 3)	National Center for Biotechnology Information (2024)

2.2. Datos

Se debe descargar cuatro muestras de secuenciación de genoma completo o WGS (del inglés *Whole Genome Sequencing*) del organismo *Helicobacter pylori*, para ser ensambladas. Las muestras se descargarán desde la base de datos SRA, por lo que se requiere sus identificadores (número de acceso o *accession number* en inglés) en el repositorio:

1. SRR22388518
2. SRR22388519
3. SRR18335438
4. SRR18335439

Nota 3: Base de Datos SRA

La base de datos SRA¹ (del inglés *Sequence Read Archive*; Katz et al. (2022)) del NCBI (del inglés *National Center for Biotechnology Information*) es, al momento de escribir este libro, el repositorio público más grande del mundo de datos de secuenciación de alto rendimiento. En esta base de datos se almacenan y comparten datos biológicos crudos generados mediante diferentes tecnologías de secuenciación como Illumina o PacBio, entre otras.

2.3. Descarga

Importante

Recuerde siempre activar el ambiente `ensam` (si no está activado ya) antes de ejecutar los comandos de este libro:

```
mamba activate ensam
```

Para empezar la descarga de las muestras, se debe abrir una `terminal` y ejecutar las órdenes que se dan a continuación.

Advertencia

Siempre que vea comentarios (líneas empezando con `#`) en las cajas de comandos, no los digite en la terminal. Estos comentarios solo proveen información adicional acerca de cada comando.

Consejo

- La descarga de cada muestra toma aproximadamente cinco minutos (o más), dependiendo de la velocidad de su conexión a Internet. No se preocupe si no ve ningún avance aparente después de ejecutar el comando de descarga (`fasterq-dump`).
- Si desea ver el progreso de cada descarga, use la opción `-p` (o `--progress`) con `fasterq-dump`

¹*Sequence Read Archive*: <https://www.ncbi.nlm.nih.gov/sra>

- Puede ver otras opciones útiles de `fasterq-dump` consultando la ayuda: `fasterq-dump --help`

```
# Entrar al directorio para datos en el directorio del proyecto
cd $HOME/proyectos/ensamblaje/datos
```

```
# Descargar la muestra
fasterq-dump -S SRR22388518
```

```
spots read      : 870,940
reads read      : 1,741,880
reads written    : 1,741,880
```

Al finalizar la descarga, `fasterq-dump` imprime en pantalla el resumen de las secuencias descargadas y escritas en archivos (salida anterior) .

Ahora verifique los archivos descargados:

```
ls -lh
```

```
total 962M
-rw-rw-r-- 1 hector hector 481M mar  2 14:40 SRR22388518_1.fastq
-rw-rw-r-- 1 hector hector 481M mar  2 14:40 SRR22388518_2.fastq
```

Como se puede observar, hay dos archivos descargados con la extensión `.fastq`. A continuación se muestra la estructura de nombre de los archivos descargados:

- `SRR22388518`: *accession number*.
- `_1`: conjunto de secuencias, en este caso *forward*.
- `.fastq`: extensión de archivo.

! Importante

Hay dos archivos dado que la muestra se secuenció con tecnología Illumina *paired-end*, en la cual resultan dos lecturas o secuencias (*reads* en inglés), *forward* (`_1`) y *reverse* (`_2`), de secuenciar los extremos de una molécula de ADN. En la Figura 2 se detalla la ubicación relativa de las dos lecturas respecto del fragmento de ADN secuenciado. En el archivo `_1` están guardadas todas las secuencias *forward* y en el archivo `_2`, todas las secuencias *reverse*.

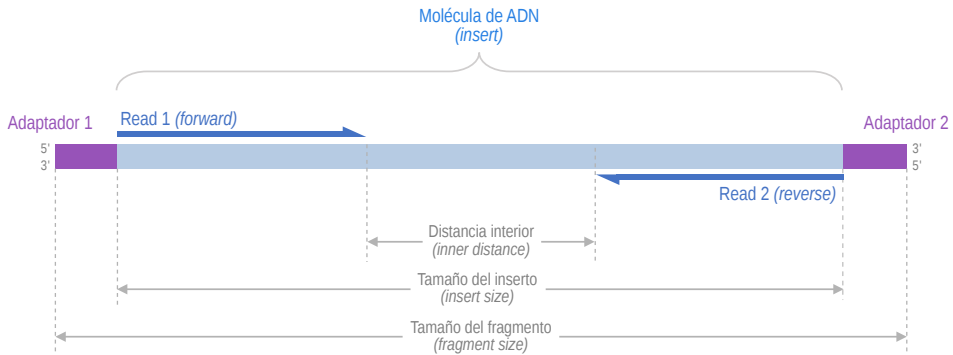


Figura 2. Representación esquemática de la secuenciación *paired-end*.

El formato FASTQ² (Cock et al., 2010) es el estándar *de facto* para datos de secuenciación de segunda generación y es usado, entre otras, por la tecnología Illumina. Este formato almacena tanto las secuencias de ADN como la calidad de cada base secuenciada.

En seguida, se debe descargar el resto de las muestras:

```
fasterq-dump -S SRR22388519
```

```
fasterq-dump -S SRR18335438
```

```
fasterq-dump -S SRR18335439
```

Ahora, aunque no es necesario, es una buena práctica comprimir los archivos descargados para que ocupen menos espacio en disco:

```
gzip *.fastq
```

Como se verá, los archivos que están comprimidos tendrán la extensión `.gz` (`.fastq.gz`):

```
ls
```

```
SRR18335438_1.fastq.gz SRR18335439_2.fastq.gz SRR22388519_1.fastq.gz
SRR18335438_2.fastq.gz SRR22388518_1.fastq.gz SRR22388519_2.fastq.gz
SRR18335439_1.fastq.gz SRR22388518_2.fastq.gz
```

²Wikipedia - FASTQ Format: https://en.wikipedia.org/wiki/FASTQ_format

Finalmente, se debe regresar al directorio raíz del proyecto:

```
cd $HOME/proyectos/ensamblaje
```

2.4. Resumen

El primer paso en un proyecto de análisis genómico es la obtención de datos. Este capítulo estableció esta tarea mediante la descarga de cuatro muestras de *H. pylori* desde la base de datos pública SRA del NCBI usando SRA Tools. Se exploró el formato FASTQ y los principios de la secuenciación Illumina *paired-end*, proporcionando el conjunto de datos necesario para los procesos posteriores de limpieza, ensamblaje y anotación genómica.



Capítulo 3.

Control de calidad y limpieza de datos

El control de calidad y preprocesamiento de datos de secuenciación constituye una etapa fundamental en el flujo de trabajo bioinformático, ya que la calidad de los datos de entrada determina la calidad de los resultados de los análisis posteriores (*downstream*). Los datos crudos de secuenciación de nueva generación contienen errores sistemáticos, adaptadores residuales y secuencias de baja calidad que pueden comprometer la precisión del ensamblaje posterior. El proceso de control de calidad comprende la evaluación de métricas como la distribución de calidades por posición, contenido de bases, presencia de adaptadores, seguido de procedimientos de filtrado y limpieza para mejorar los datos.

Este capítulo presenta la implementación de un protocolo de control de calidad que combina la evaluación inicial con *FastQC*, la unificación de reportes mediante *MultiQC*, y el preprocesamiento utilizando *fastp* para la remoción de adaptadores, filtrado por calidad y eliminación de artefactos. Se describen también los criterios de interpretación de las métricas de calidad, la configuración de parámetros para datos Illumina *paired-end*, y la verificación de los datos limpios. Al finalizar este capítulo, se obtienen datos limpios y de calidad apropiada, listos para el proceso de ensamblaje.

! Importante

Es importante comprender la calidad de los datos de secuenciación. Puede aprender sobre la puntuación Phred y cómo se codifica en el formato FASTQ en los artículos de Wikipedia: “Nivel de calidad Phred”^a y “FASTQ format”^b (en inglés), respectivamente.

^aWikipedia - Nivel de calidad: https://es.wikipedia.org/wiki/Nivel_de_calidad_Phred

^bWikipedia - FASTQ format: https://en.wikipedia.org/wiki/FASTQ_format

3.1. Software requerido

Tabla 2. Software requerido para control de calidad y limpieza de datos.

Software	Versión	Descripción	Referencia
FastQC	0.12.1	Herramienta para el control de calidad en datos de secuenciación de alto rendimiento.	Andrews (2010)
MultiQC ¹	1.14	Herramienta para unificar y resumir múltiples reportes generados por FastQC y muchos otros programas de bioinformática.	Ewels et al. (2016)
fastp ²	0.23.2	Herramienta para el pre-procesamiento rápido de archivos Fastq.	Chen et al. (2018)

3.2. Determinación de calidad

Para revisar la calidad de los datos descargados, se usa **fastqc**:

```
# Entrar al directorio de datos en la carpeta del proyecto ensamblaje
cd $HOME/proyectos/ensamblaje/datos
# Analizar todos los archivos descargados
fastqc -o qc/fastqc *.fastq.gz
```

A continuación, se verán mensajes de progreso de **fastqc**:

```
Started analysis of SRR18335438_1.fastq.gz
Approx 5% complete for SRR18335438_1.fastq.gz
Approx 10% complete for SRR18335438_1.fastq.gz
Approx 15% complete for SRR18335438_1.fastq.gz
Approx 20% complete for SRR18335438_1.fastq.gz
Approx 25% complete for SRR18335438_1.fastq.gz
Approx 30% complete for SRR18335438_1.fastq.gz
Approx 35% complete for SRR18335438_1.fastq.gz
Approx 40% complete for SRR18335438_1.fastq.gz
Approx 45% complete for SRR18335438_1.fastq.gz
Approx 50% complete for SRR18335438_1.fastq.gz
Approx 55% complete for SRR18335438_1.fastq.gz
```

¹MultiQC, sitio Web: <https://multiqc.info/>

²fastp, repositorio GitHub: <https://github.com/OpenGene/fastp>

```
Approx 60% complete for SRR18335438_1.fastq.gz
Approx 65% complete for SRR18335438_1.fastq.gz
Approx 70% complete for SRR18335438_1.fastq.gz
Approx 75% complete for SRR18335438_1.fastq.gz
Approx 80% complete for SRR18335438_1.fastq.gz
Approx 85% complete for SRR18335438_1.fastq.gz
Approx 90% complete for SRR18335438_1.fastq.gz
Approx 95% complete for SRR18335438_1.fastq.gz
Analysis complete for SRR18335438_1.fastq.gz
...
```

Al final, por cada archivo `.fastq` analizado, `fastqc` genera un archivo `.html` con el reporte de calidad y un archivo comprimido (`.zip`):

```
ls qc/fastqc
```

```
SRR18335438_1_fastqc.html  SRR18335439_2_fastqc.html
└─ SRR22388519_1_fastqc.html
SRR18335438_1_fastqc.zip  SRR18335439_2_fastqc.zip
└─ SRR22388519_1_fastqc.zip
SRR18335438_2_fastqc.html  SRR22388518_1_fastqc.html
└─ SRR22388519_2_fastqc.html
SRR18335438_2_fastqc.zip  SRR22388518_1_fastqc.zip
└─ SRR22388519_2_fastqc.zip
SRR18335439_1_fastqc.html  SRR22388518_2_fastqc.html
SRR18335439_1_fastqc.zip  SRR22388518_2_fastqc.zip
```

Consejo

El archivo comprimido `.zip` contiene una copia del reporte `.html`, gráficos, y archivos de estadísticas.

3.2.1. Reporte de calidad FastQC

Para comprender el contenido de los reportes, se tomará como ejemplo el archivo `SRR18335438_1_fastqc.html`. Al abrir el archivo, se podrá notar que el reporte contiene diferentes secciones, llamadas módulos. Los módulos corresponden a diferentes evaluaciones de calidad que la herramienta hace sobre los datos (Figura 3). Puede consultar información detallada (en inglés) acerca de FastQC y sus módulos en el sitio Web del programa³ en la sección de documentación⁴. Por cada módulo,

³Sitio Web de FastQC: <https://www.bioinformatics.babraham.ac.uk/projects/fastqc>

⁴Ayuda de FastQC (en inglés): <https://www.bioinformatics.babraham.ac.uk/projects/fastqc/Help>

el reporte presenta un ícono con tres posibles estados de acuerdo al resultado de la evaluación (Tabla 3).

Tabla 3. Posibles resultados de la evaluación de calidad realizada por la herramienta FastQC

Símbolo	Descripción
	Pasó. La muestra pasó la evaluación de calidad en esta categoría (módulo). No hay problemas de calidad.
	Advertencia. La muestra tiene un posible problema calidad. El usuario debería revisar la muestra y tomar las medidas necesarias para evitar el error de ser posible. Por la naturaleza de la secuenciación de ADN, es normal que se presenten advertencias en algunos módulos sin que signifique que haya un problema real o grave.
	Falló. La muestra falló la evaluación de calidad en este módulo. El usuario debería limpiar los datos para corregir el error.

De acuerdo con los estados, la muestra tiene fallos en las evaluaciones de calidad de los módulos “Per base sequence quality” y “Sequence Duplication Levels”, y tiene una advertencia en el módulo “Sequence Length Distribution” (Figura 3).

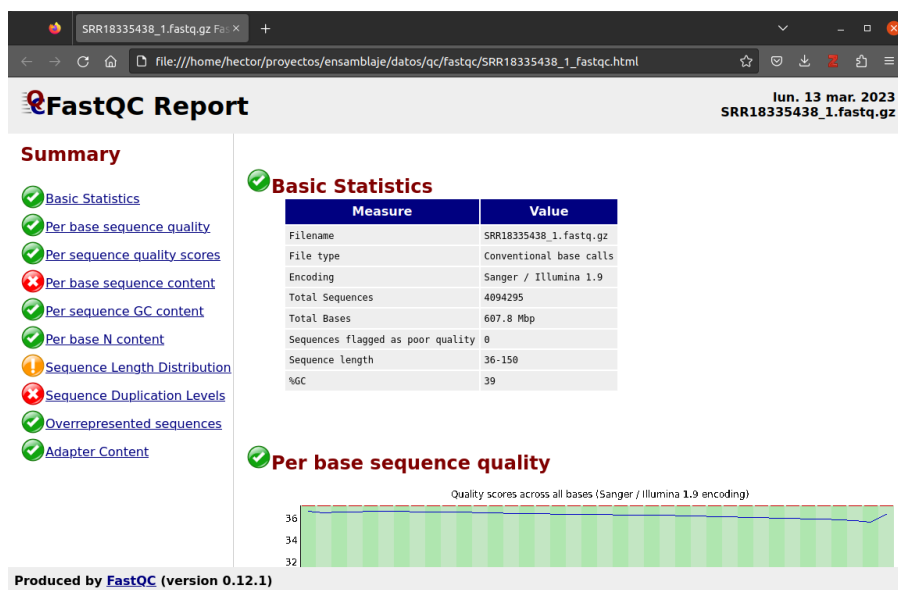


Figura 3. Captura de pantalla de un reporte de calidad generado con la aplicación FastQC.

Tal vez el módulo más importante a revisar es “*Per base sequence quality*”, el cual muestra el valor de calidad de por cada posición a lo largo de las secuencias del archivo procesado. Típicamente, en archivos producidos con tecnología Illumina, la calidad es baja en las primeras siete a 10 bases de las secuencias, incrementando hasta alcanzar su máximo hacia la parte media de la secuencia. Luego, la calidad empieza a decrecer hasta encontrar sus mínimos valores al final de la secuencia. Por lo general, la calidad mínima para lograr ensamblajes decentes es $Q \geq 20$ (eje Y en el gráfico). Así, este gráfico es crucial para que el usuario decida el número de bases (eje X) que se recortarán en los extremos de las secuencias (inicio 5'; final 3'), los cuales contienen las secciones con menor calidad.

3.2.2. Unificación de reportes de calidad

Dado que los reportes de FastQC son independientes por cada archivo de lecturas ([1](#) y [2](#)) y cada muestra, es difícil tener una visión general de la calidad de todos los datos en conjunto. Conviene entonces unificar los datos de calidad en un solo reporte para lograr una mejor interpretación de la información.

Esta unificación se logra con la aplicación MultiQC. Para generar el reporte debe ejecutar la siguiente orden:

```
multiqc --filename multiqc_report.html --outdir qc qc/fastqc
```



```
/// MultiQC | v1.14
```



```
|           multiqc | Search path :  
- /home/hector/proyectos/ensamblaje/datos/qc/fastqc  
|       searching | ████████████████████████████████████████ 100% 16/16  
|         fastqc | Found 8 reports  
|        multiqc | Compressing plot data  
|        multiqc | Report      : qc/multiqc_report.html  
|        multiqc | Data       : qc/multiqc_report_data  
|        multiqc | MultiQC complete
```

En este punto, MultiQC ha generado dos elementos, el archivo [multiqc_report.html](#) y un directorio [multiqc_report_data](#) que contiene archivos de estadísticas. Al abrir el reporte [.html](#) se puede ver la información de calidad de todos los archivos analizados con FastqQC (Figura 4).

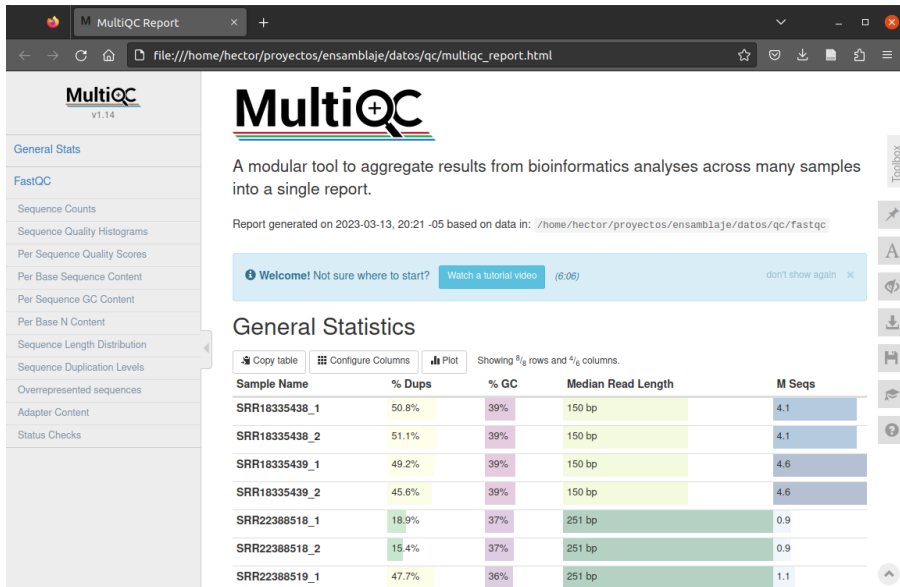


Figura 4. Captura de pantalla del reporte de calidad unificado generado con MultiQC.

Consejo

Puede abrir el reporte [ensamblaje/datos/qc/multiqc_report.html](#) para revisar toda la información de calidad evaluada por FastQC.

3.2.3. Interpretación de resultados

Al revisar la sección “General Statistics” del reporte MultiQC, se puede observar que los tamaños de secuencia varían según la muestra. La longitud (mediana) de secuencias de las muestras [SRR18335438](#) y [SRR18335439](#) es de 150 pb (pares de bases o bp por *base pairs* en inglés), mientras que la de las muestras [SRR22388518](#) y [SRR22388519](#) es de 251 pb.

Ahora, se debe revisar el gráfico de calidad de secuenciación en la sección “Sequence Quality Histograms” (Figura 5). Además de la diferencia de longitud de secuencias, se puede notar que:

- La calidad al inicio de las secuencias en todos los archivos es superior a 30 ($Q \geq 30$), la cual se considera como buena calidad

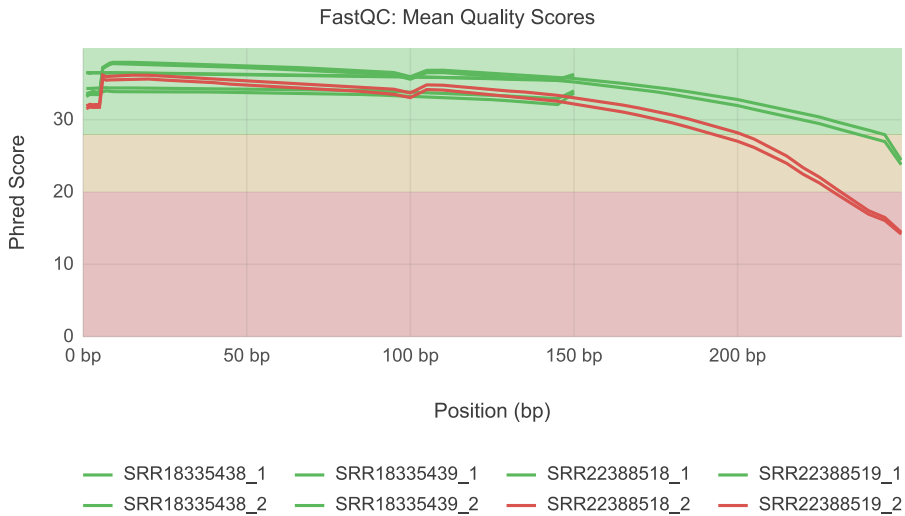


Figura 5. Gráfico de calidad de secuenciación por base

- La calidad al final de las secuencias es superior a 20 ($Q \geq 20$) en todos los archivos, con excepción de las secuencias *reverse* (2) de las muestras [SRR22388518](#) y [SRR22388519](#). Esta calidad se considera como aceptable.

De estas observaciones, se puede concluir que la limpieza de los datos consistirá en recortar las últimas bases (aproximadamente 20) de los archivos [SRR22388518_2.fastq.gz](#) y [SRR22388519_2.fastq.gz](#), pues son las que presentan baja calidad. Además, como buena práctica, se eliminarán aquellas secuencias cuya calidad promedio sea $Q < 20$.

3.3. Limpieza

Se usará la aplicación `fastp` para recortar y filtrar las secuencias por calidad. Se debe empezar procesando la primera muestra:

```
# Entrar al directorio raíz del proyecto (en mi caso,
└─ /home/hector/ensamblaje)
cd ..

# Comando para ejecutar la limpieza de la primera muestra
fastp \
```

```
--verbose \  
--thread 4 \  
--detect_adapter_for_pe \  
--cut_tail \  
--cut_mean_quality 20 \  
--average_qual 20 \  
--report_title 'Reporte fastp: SRR22388518' \  
--in1 datos/SRR22388518_1.fastq.gz \  
--in2 datos/SRR22388518_2.fastq.gz \  
--out1 resultados/00_datos_limpios/SRR22388518_1.clean.fastq.gz \  
--out2 resultados/00_datos_limpios/SRR22388518_2.clean.fastq.gz \  
--unpaired1  
  resultados/00_datos_limpios/SRR22388518_1.unpaired.fastq.gz \  
--unpaired2  
  resultados/00_datos_limpios/SRR22388518_2.unpaired.fastq.gz \  
--html  
  resultados/00_datos_limpios/qc/fastp/SRR22388518.report.fastp.html  
  \  
--json  
  resultados/00_datos_limpios/qc/fastp/SRR22388518.report.fastp.json
```

A continuación se muestran las opciones usadas con el comando y su finalidad:

- **--verbose**: provee información adicional durante el procesamiento de los archivos
- **--thread**: usa cuatro hilos de procesamiento de manera paralela para acelerar el procesamiento de los archivos
- **--detect_adapter_for_pe**: detecta las secuencias de adaptadores de secuenciación y las elimina
- **--cut_tail**: corta la “cola” (*tail*) de las secuencias, la cual se dijo tiene las bases de más baja calidad
- **--cut_mean_quality**: la calidad media mínima aceptada para corte de secuencias. fastp usa una ventana de 4 bases desde el extremo de la secuencia para revisar calidad; si el promedio de estas cuatro bases está por debajo de 20, elimina este extremo.
- **--average_qual**: la calidad media mínima aceptada para mantener una secuencia. Si al calcular la calidad media de toda la secuencia está por debajo de 20, la secuencia es descartada.
- **--report_title**: el título del reporte
- **--in1**: el archivo de secuencias *forward* a ser procesado
- **--in2**: el archivo de secuencias *reverse* a ser procesado
- **--out1**: el archivo de salida con las secuencias *forward* que pasaron el control de calidad

- `--out2`: el archivo de salida con las secuencias *reverse* que pasaron el control de calidad
- `--unpaired1`: el archivo de salida con las secuencias *forward* no pareadas que pasaron el control de calidad (no tienen una secuencia par *reverse*)
- `--unpaired2`: el archivo de salida con las secuencias *reverse* no pareadas que pasaron el control de calidad (no tienen una secuencia par *forward*)
- `--html`: el archivo de reporte en formato html
- `--json`: el archivo de reporte en formato json (estadísticas)

 Consejo

Para conocer todas las opciones disponibles de fastp consulte la ayuda ejecutando `fastp --help`.

En el anterior comando, en lugar de cortar un número determinado de bases al final de las secuencias, se usa la estrategia del programa fastp. Por cada secuencia del archivo, fastp revisa la calidad media de una pequeña porción de 4 bases, llamado ventana, empezando desde el extremo final. Si la calidad media está por debajo del mínimo definido (20 en este caso), fastp recorta esta ventana de la secuencia. Este proceso se repita hasta que no se encuentre una ventana que deba recortarse. Así, solo las secuencias con baja calidad al final son recortadas. Este mismo principio se puede aplicar para el inicio de las secuencias, cuando sea el caso.

Al ejecutar el anterior comando se verá una salida como la siguiente:

```
Detecting adapter sequence for read1...
>Illumina TruSeq Adapter Read 1
AGATCGGAAGAGCACACGTCTGAACTCCAGTCA

Detecting adapter sequence for read2...
No adapter detected for read2

[19:24:32] start to load data of read1
[19:24:32] start to load data of read2
[19:24:36] Read1: loading completed with 871 packs
[19:24:36] Read2: loading completed with 871 packs
[19:24:39] thread 4 data processing completed
[19:24:39] thread 4 finished
[19:24:39] thread 3 data processing completed
[19:24:39] thread 3 finished
[19:24:39] thread 2 data processing completed
[19:24:39] thread 2 finished
[19:24:39] thread 1 data processing completed
[19:24:39] thread 1 finished
```

Ensamblaje y anotación de genomas bacterianos

```
[19:24:39] resultados/00_datos_limpios/SRR22388518_1.unpaired.fastq.gz
↳ writer finished
[19:24:39] resultados/00_datos_limpios/SRR22388518_2.unpaired.fastq.gz
↳ writer finished
[19:24:40] resultados/00_datos_limpios/SRR22388518_2.clean.fastq.gz
↳ writer finished
[19:24:40] resultados/00_datos_limpios/SRR22388518_1.clean.fastq.gz
↳ writer finished
[19:24:40] start to generate reports
```

```
Read1 before filtering:
total reads: 870940
total bases: 218350507
Q20 bases: 198719240(91.0093%)
Q30 bases: 184185641(84.3532%)
```

```
Read2 before filtering:
total reads: 870940
total bases: 218399177
Q20 bases: 175386628(80.3055%)
Q30 bases: 149084629(68.2624%)
```

```
Read1 after filtering:
total reads: 863093
total bases: 209876970
Q20 bases: 193175278(92.0422%)
Q30 bases: 180122777(85.823%)
```

```
Read2 after filtering:
total reads: 863093
total bases: 201834951
Q20 bases: 170111238(84.2823%)
Q30 bases: 146895463(72.78%)
```

```
Filtering result:
reads passed filter: 1726186
reads failed due to low quality: 12556
reads failed due to too many N: 488
reads failed due to too short: 2650
reads with adapter trimmed: 150684
bases trimmed due to adapters: 7399476
```

```
Duplication rate: 0.0137782%
```

```
Insert size peak (evaluated by paired-end reads): 251
```

```
JSON report: resultados/00_datos_limpios/qc/fastp/report.fastp.json
```

HTML report: resultados/00_datos_limpios/qc/fastp/report.fastp.html

```
fastp --verbose --thread 4 --detect_adapter_for_pe --cut_tail
  --cut_mean_quality 20 --average_qual 20 --report_title Reporte fastp:
  SRR22388518 --in1 datos/SRR22388518_1.fastq.gz --in2
  datos/SRR22388518_2.fastq.gz --out1
  resultados/00_datos_limpios/SRR22388518_1.clean.fastq.gz --out2
  resultados/00_datos_limpios/SRR22388518_2.clean.fastq.gz --unpaired1
  resultados/00_datos_limpios/SRR22388518_1.unpaired.fastq.gz
  --unpaired2
  resultados/00_datos_limpios/SRR22388518_2.unpaired.fastq.gz --html
  resultados/00_datos_limpios/qc/fastp/SRR22388518.report.fastp.html
  --json
  resultados/00_datos_limpios/qc/fastp/SRR22388518.report.fastp.json
fastp v0.23.2, time used: 22 seconds
```

Se puede encontrar información importante en los mensajes de la salida, como por ejemplo el número de secuencias y la calidad antes y después de aplicar la limpieza o filtrado (secciones [Read1 before filtering:](#), [Read2 before filtering:](#), [Read1 after filtering:](#) y [Read2 after filtering:](#)), y el resumen del número de lecturas que pasaron y no pasaron el filtro (sección [Filtering result](#)).

Junto con los archivos de secuencias filtradas, fastp genera un archivo de reporte [.html](#) y un archivo de estadísticas.

Ahora se debe procesar el resto de los archivos:

```
fastp \
  --verbose \
  --thread 4 \
  --detect_adapter_for_pe \
  --cut_tail \
  --cut_mean_quality 20 \
  --average_qual 20 \
  --report_title 'Reporte fastp: SRR22388519' \
  --in1 datos/SRR22388519_1.fastq.gz \
  --in2 datos/SRR22388519_2.fastq.gz \
  --out1 resultados/00_datos_limpios/SRR22388519_1.clean.fastq.gz \
  --out2 resultados/00_datos_limpios/SRR22388519_2.clean.fastq.gz \
  --unpaired1
  resultados/00_datos_limpios/SRR22388519_1.unpaired.fastq.gz \
  --unpaired2
  resultados/00_datos_limpios/SRR22388519_2.unpaired.fastq.gz \
  --html
  resultados/00_datos_limpios/qc/fastp/SRR22388519.report.fastp.html
  \
```

```
--json
  ↪ resultados/00_datos_limpios/qc/fastp/SRR22388519.report.fastp.json

fastp \
  --verbose \
  --thread 4 \
  --detect_adapter_for_pe \
  --cut_tail \
  --cut_mean_quality 20 \
  --average_qual 20 \
  --report_title 'Reporte fastp: SRR18335438' \
  --in1 datos/SRR18335438_1.fastq.gz \
  --in2 datos/SRR18335438_2.fastq.gz \
  --out1 resultados/00_datos_limpios/SRR18335438_1.clean.fastq.gz \
  --out2 resultados/00_datos_limpios/SRR18335438_2.clean.fastq.gz \
  --unpaired1
  ↪ resultados/00_datos_limpios/SRR18335438_1.unpaired.fastq.gz \
  --unpaired2
  ↪ resultados/00_datos_limpios/SRR18335438_2.unpaired.fastq.gz \
  --html
  ↪ resultados/00_datos_limpios/qc/fastp/SRR18335438.report.fastp.html
  ↪ \
  --json
  ↪ resultados/00_datos_limpios/qc/fastp/SRR18335438.report.fastp.json

fastp \
  --verbose \
  --thread 4 \
  --detect_adapter_for_pe \
  --cut_tail \
  --cut_mean_quality 20 \
  --average_qual 20 \
  --report_title 'Reporte fastp: SRR18335439' \
  --in1 datos/SRR18335439_1.fastq.gz \
  --in2 datos/SRR18335439_2.fastq.gz \
  --out1 resultados/00_datos_limpios/SRR18335439_1.clean.fastq.gz \
  --out2 resultados/00_datos_limpios/SRR18335439_2.clean.fastq.gz \
  --unpaired1
  ↪ resultados/00_datos_limpios/SRR18335439_1.unpaired.fastq.gz \
  --unpaired2
  ↪ resultados/00_datos_limpios/SRR18335439_2.unpaired.fastq.gz \
  --html
  ↪ resultados/00_datos_limpios/qc/fastp/SRR18335439.report.fastp.html
  ↪ \
```

```
--json
  -o resultados/00_datos_limpios/qc/fastp/SRR18335439.report.fastp.json
```

3.4. Verificación de calidad después de la limpieza

Se puede usar FastQC y MultiQC para revisar la calidad de las muestras después de la limpieza:

```
# Generar reportes con FastQC
fastqc \
  -o resultados/00_datos_limpios/qc/fastqc \
  resultados/00_datos_limpios/*.clean.fastq.gz

# Generar reporte unificado con MultiQC
multiqc \
  --filename multiqc_report.html \
  --outdir resultados/00_datos_limpios/qc \
  resultados/00_datos_limpios/qc/fastp \
  resultados/00_datos_limpios/qc/fastqc
```

Nótese que en el anterior comando de MultiQC, se pasó no solo el directorio de reportes FastQC, sino también el directorio con los reportes de fastp. La aplicación se encarga de generar un solo documento `.html` con dos secciones, una para fastp y otra para FastQC.

Consejo

Explore el reporte `resultados/00_datos_limpios/qc/multiqc_report.html` para ver las diferentes estadísticas de calidad después de haber aplicado los filtros a las muestras.

Al revisar el gráfico de calidad de secuenciación por base, generado sobre los datos limpios (Figura 6), se puede ver que las dos muestras que te tenían baja calidad hacia el final de las secuencias, ahora aparecen con una calidad $Q > 20$ (fondo color naranja).

Finalmente, las muestras están listas para ser ensambladas.

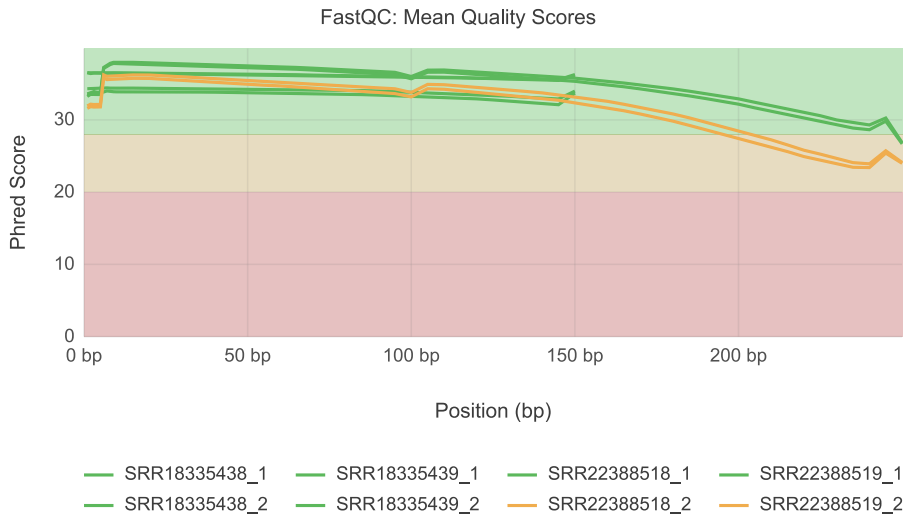


Figura 6. Gráfico de calidad de secuenciación por base generado sobre los datos limpios.

3.5. Resumen

El control de calidad representa un paso fundamental en cualquier proyecto de análisis genómico, ya que la calidad de los datos determina la confiabilidad de los análisis posteriores. Este capítulo estableció un flujo de trabajo para la evaluación y mejora de la calidad de datos de secuenciación Illumina mediante la implementación de herramientas especializadas. El proceso incluyó la evaluación inicial con FastQC para generar reportes de calidad, la visualización de métricas con MultiQC para facilitar el análisis conjunto de múltiples muestras, y la interpretación de las métricas obtenidas para identificar problemas comunes en los datos. Posteriormente, se aplicaron filtros de calidad con fastp para optimizar los conjuntos de datos, eliminando lecturas de baja calidad. Como resultado, se obtuvieron conjuntos de datos limpios y de calidad apropiada, preparados para el proceso de ensamblaje y validados mediante la verificación del impacto positivo de los filtros aplicados.



Capítulo 4.

Ensamblaje

El ensamblaje es el proceso computacional que reconstruye secuencias genómicas (completas o parciales) a partir de millones de lecturas cortas (*reads*) de secuenciación. Existen dos tipos de ensamblaje según la manera en que se realiza el proceso, guiado por referencia y *de novo*. En el ensamblaje guiado por referencia, se usa la información del genoma existente más completo del organismo que se estudia, para guiar el ordenamiento de las lecturas. Sin embargo, existen casos en los que el organismo aún no posee un genoma ensamblado. Este último caso requiere un ensamblaje *de novo*, en el cual el ordenamiento de las lecturas se hace desde cero usando únicamente la información que proveen las lecturas mismas, es decir, sin guía.

Piense en el ensamblaje como el proceso de armar un juego de rompecabezas de millones de piezas. Cada pieza representa una lectura y la imagen final representa el genoma ensamblado. El objetivo es reconstruir la imagen final organizando las piezas en el orden correcto. Armar el rompecabezas usando la información de la imagen guía impresa en la caja del juego (colores, formas), representa el ensamblaje guiado por referencia. Por el contrario, armar el rompecabezas sin observar ni una sola vez la imagen de la caja, representa el ensamblaje *de novo*.

El ensamblaje utiliza algoritmos especializados que identifican solapamiento entre las lecturas para construir secuencias contiguas más largas que representan fragmentos del genoma original del organismo secuenciado.

En este capítulo se presenta una visión integral del ensamblaje *de novo* de genomas bacterianos, comenzando con las diferentes estrategias metodológicas disponibles (ensamblaje de genomas individuales, metagenómico e híbrido) y su posterior aplicación práctica. Se utiliza SPAdes, uno de los ensambladores *de novo* más efectivos para datos de secuenciación Illumina, describiendo la configuración de parámetros apropiados para genomas procarióticos y la ejecución del proceso completo para las muestras de *Helicobacter pylori* previamente procesadas. Al final del capítulo, se obtendrán los archivos FASTA con las secuencias ensambladas listas para la evaluación de calidad.

4.1. Software requerido

Tabla 4. Software requerido para ensamblaje.

Software	Versión	Descripción	Referencia
SPAdes ¹²	3.15.5	Conjunto de herramientas para ensamblaje <i>de novo</i> de genomas con múltiples flujos de trabajo de ensamblaje especializado.	Bankevich et al. (2012); Prjibelski et al. (2020)

Antes de proceder a la ejecución práctica del ensamblaje, es importante comprender los diferentes enfoques metodológicos disponibles según el tipo de datos de secuenciación y la naturaleza de las muestras. El ensamblaje de genomas moderno puede clasificarse en tres estrategias principales, cada una con características, ventajas y aplicaciones específicas.

4.2. Estrategias de ensamblaje genómico

4.2.1. Ensamblaje de genomas individuales

El ensamblaje de genomas individuales constituye el enfoque utilizado en este capítulo para *H. pylori*. Se refiere al proceso de reconstrucción del material genético de un solo organismo a partir de datos de secuenciación obtenidos de cultivos puros o aislamientos clínicos. Este método asume que todas las lecturas de secuenciación provienen del mismo organismo, simplificando considerablemente el análisis computacional.

Las principales características del ensamblaje de genomas individuales incluyen: (1) cobertura uniforme a lo largo del genoma, facilitando la identificación de solapamientos entre lecturas; (2) homogeneidad taxonómica, permitiendo el uso de parámetros optimizados para la especie específica; (3) menor riesgo de quimeras, ya que no existen secuencias conservadas entre diferentes organismos; y (4) interpretación directa, donde cada *contig* y *scaffold* representa una región específica del genoma del organismo estudiado.

¹SPAdes, sitio Web: <http://cab.spbu.ru/software/spades/>

²SPAdes, repositorio GitHub: <https://github.com/ablab/spades>

4.2.2. Ensamblaje metagenómico

El ensamblaje metagenómico representa un desafío computacional significativamente más complejo, ya que debe reconstruir simultáneamente múltiples genomas a partir de muestras ambientales que contienen comunidades microbianas diversas. Este enfoque requiere algoritmos y herramientas especializadas como MetaSPAdes (Nurk et al., 2017), MEGAHIT (Li et al., 2015), o Flye (Kolmogorov et al., 2019).

Los principales desafíos incluyen: (1) separación de especies presentes en abundancias variables; (2) manejo de regiones conservadas que pueden generar secuencias quiméricas; (3) cobertura desigual entre diferentes organismos; y (4) complejidad computacional para procesar millones de lecturas de origen diverso. Para una descripción detallada, consulte el Apéndice D.

4.2.3. Ensamblaje híbrido

El ensamblaje híbrido representa la metodología más avanzada, combinando las fortalezas complementarias de diferentes tecnologías de secuenciación para generar genomas de calidad superior. Esta estrategia integra la alta precisión nucleotídica de las lecturas cortas Illumina (~150-300 pb con precisión >99%) con la capacidad de resolución estructural de las lecturas largas de tecnologías como PacBio y Oxford Nanopore (ONT) (1-100+ kb).

La implementación práctica utiliza herramientas especializadas como Unicycler (Wick et al., 2017), Flye (Kolmogorov et al., 2019), o módulos híbridos de SPAdes. El flujo típico incluye: (1) ensamblaje inicial con lecturas largas para establecer la estructura genómica; (2) pulido iterativo con lecturas cortas para corregir errores puntuales; y (3) validación cruzada entre tecnologías. Este enfoque puede producir genomas bacterianos con continuidad excepcional (frecuentemente un solo *contig* circular) y precisión nucleotídica superior al 99.99%, representando el estado del arte actual cuando se dispone de recursos para ambas tecnologías. Se recomienda la lectura de Bashir et al. (2012) y Koren & Phillippy (2015) para profundizar en ensamblajes microbianos híbridos y con solo lecturas largas, respectivamente. También puede leer a Rice & Green (2019) para conocer más sobre distintos enfoques para el ensamblaje de organismos complejos usando distintas tecnologías de secuenciación.

4.3. Ejecución del ensamblaje con SPAdes

Aunque existen varios ensambladores de genomas (Ver Nota 4), en este libro se usará SPAdes (Prijbelski et al., 2020) por ser un ensamblador *de novo*³ muy popular,

³Wikipedia - *De novo assemblers*: https://en.wikipedia.org/wiki/De_novo_sequence_assemblers

maduro, fácil de usar, versátil, y que produce buenos resultados en el ensamblaje de genomas bacterianos a partir de secuencias cortas Illumina. SPAdes fue desarrollado inicialmente para ensamblar secuencias cortas Illumina obtenidas de aislamientos bacterianos puros y secuenciación de células individuales (*single-cell sequencing*), y que con el tiempo ha sido extendido para usar secuencias de otras tecnologías (p. ej. PacBio u ONT) y ensamblar metagenomas, transcriptomas, y plásmidos.

i Nota 4: Alternativas de ensambladores

Se pueden utilizar otros ensambladores de genomas como SKESA (Suvorov et al., 2018), Velvet (Zerbino & Birney, 2008), Canu (Koren et al., 2017), y Unicycler (Wick et al., 2017). La elección del ensamblador depende de varios factores, entre ellos: el tipo de datos (tecnología de secuenciación), el tipo de muestra (aislamiento puro o metagenoma), la complejidad del genoma (bacteriano, eucariota, con plásmidos), y los recursos computacionales disponibles (memoria RAM, número de procesadores, espacio en disco). Sin embargo, SPAdes suele ofrecer mejores resultados en términos de continuidad y precisión para genomas bacterianos con datos Illumina. SKESA es conocido por su rapidez y bajo consumo de memoria, siendo una buena alternativa cuando se dispone de recursos computacionales limitados. Velvet, aunque fue uno de los primeros ensambladores *de novo* para secuencias cortas, actualmente es menos utilizado debido a la aparición de herramientas más modernas y eficientes.

Para más información acerca de los ensambladores de genomas mencionados, consulte el Apéndice B.

4.3.1. Ensamblaje de la muestra SRR22388518

Para ensamblar la muestra se debe ejecutar la siguiente orden:

```
# Entrar en el directorio raíz del proyecto
cd ensamblaje
# Ejecutar el ensamblaje
spades.py \
  --isolate \
  -1 resultados/00_datos_limpios/SRR22388518_1.clean.fastq.gz \
  -2 resultados/00_datos_limpios/SRR22388518_2.clean.fastq.gz \
  -o resultados/01_ensamblaje/SRR22388518 \
  -t 4
```

Los parámetros usados en el comando anterior son:

- **--isolate**: indica a SPAdes que el *pipeline* a ser ejecutado es el de ensamblaje de aislamientos bacterianos.
- **-1**: archivo con los *reads forward*.
- **-2**: archivo con los *reads reverse*.
- **-o**: directorio donde se almacenarán los archivos de salida.
- **-t**: hilos (*threads*) de procesos en paralelo para incrementar la velocidad del ensamblaje.

El tiempo de ensamblaje depende del tamaño de la muestra a ensamblar, es decir de número de *reads* en los archivos **.fastq.gz** de entrada, y del número de núcleos de procesador (*cores*) del computador y los hilos de procesamiento asignados al comando (opción **-t**). En el caso antepuesto, el proceso tomó 35 minutos para ensamblar un total de 1.726.186 de *reads* (863.093×2) usando 4 hilos en una máquina con ocho *cores*.

4.3.2. Interpretación de los resultados de SPAdes

La salida del comando de ensamblaje con SPAdes es la siguiente:

```
Command line: /home/hector/mambaforge/envs/ensam/bin/spades.py --isolate
↳ -1 /home/hector/proyectos/ensamblaje/resultados/00_datos_limpios/SR_
↳ R22388518_1.clean.fastq.gz -2
↳ /home/hector/proyectos/ensamblaje/resultados/00_datos_limpios/SRR223_
↳ 88518_2.clean.fastq.gz -o
↳ /home/hector/proyectos/ensamblaje/resultados/01_ensamblaje/SRR223885_
↳ 18 -t 4
```

System information:

SPAdes version: 3.15.5

Python version: 3.10.6

OS: Linux-5.15.0-67-generic-x86_64-with-glibc2.31

```
Output dir: /home/hector/proyectos/ensamblaje/resultados/01_ensamblaje/S_
↳ RR22388518
```

Mode: ONLY assembling (without read error correction)

Debug mode is turned OFF

Dataset parameters:

Isolate mode

Reads:

Library number: 1, library type: paired-end
orientation: fr

```
left reads: ['/home/hector/proyectos/ensamblaje/resultados/00_datos_
↳ _limpios/SRR22388518_1.clean.fastq.gz']
right reads: ['/home/hector/proyectos/ensamblaje/resultados/00_datos_
↳ _limpios/SRR22388518_2.clean.fastq.gz']
interlaced reads: not specified
single reads: not specified
merged reads: not specified
Assembly parameters:
  k: automatic selection based on read length
  Repeat resolution is enabled
  Mismatch careful mode is turned OFF
  MismatchCorrector will be SKIPPED
  Coverage cutoff is turned OFF
Other parameters:
  Dir for temp files: /home/hector/proyectos/ensamblaje/resultados/01_ens_
↳ _samblaje/SRR22388518/tmp
  Threads: 4
  Memory limit (in Gb): 7

===== SPAdes pipeline started. Log can be found here: /home/hector/proy_
↳ _ectos/ensamblaje/resultados/01_ensamblaje/SRR22388518/spades.log

...

===== Terminate finished.

* Assembled contigs are in /home/hector/proyectos/ensamblaje/resultados_
↳ _/01_ensamblaje/SRR22388518/contigs.fasta
* Assembled scaffolds are in /home/hector/proyectos/ensamblaje/resultad_
↳ _os/01_ensamblaje/SRR22388518/scaffolds.fasta
* Paths in the assembly graph corresponding to the contigs are in
↳ _/home/hector/proyectos/ensamblaje/resultados/01_ensamblaje/SRR22388_
↳ _518/contigs.paths
* Paths in the assembly graph corresponding to the scaffolds are in
↳ _/home/hector/proyectos/ensamblaje/resultados/01_ensamblaje/SRR22388_
↳ _518/scaffolds.paths
* Assembly graph is in /home/hector/proyectos/ensamblaje/resultados/01_
↳ _ensamblaje/SRR22388518/assembly_graph.fastg
* Assembly graph in GFA format is in
↳ _/home/hector/proyectos/ensamblaje/resultados/01_ensamblaje/SRR22388_
↳ _518/assembly_graph_with_scaffolds.gfa

===== SPAdes pipeline finished.

SPAdes log can be found here: /home/hector/proyectos/ensamblaje/resultad_
↳ _os/01_ensamblaje/SRR22388518/spades.log
```

Thank you for using SPAdes!

En la salida anterior, se puede observar en el inicio el comando ejecutado, información del sistema y la ubicación de los datos de entrada y salida. `spades.py` indica el inicio del ensamblaje mediante el encabezado `===== SPAdes pipeline started.` y presenta la ubicación del archivo `log` (`Log can be found here: ...`). Este archivo `log` contiene toda la información del proceso del ensamblaje, incluido cualquier error que se haya presentado. Esta información es muy importante pues permite tomar acciones para corregir posibles errores. `spades.py` también detalla el proceso de ensamblaje a través de una amplia serie de mensajes ubicados en parte indicada por los tres puntos `...` en la salida anterior. Cuando el proceso de ensamblaje es exitoso, `spades.py` presenta la sección `===== Terminate finished.` que contiene la ubicación de los archivos de salida. Por último, bajo la sección `===== SPAdes pipeline finished.`, el programa presenta de nuevo la ubicación del archivo `log`.

Ahora, se examina el directorio de salida de SPAdes:

```
ls resultados/01_ensamblaje/SRR22388518
```

```
assembly_graph_after_simplification.gfa
assembly_graph.fastg
assembly_graph_with_scaffolds.gfa
before_rr.fasta
contigs.fasta
contigs.paths
dataset.info
input_dataset.yaml
K127
K21
K33
K55
K77
K99
misc
params.txt
pipeline_state
run_spades.sh
run_spades.yaml
scaffolds.fasta
scaffolds.paths
spades.log
tmp
```

Cuando el ensamblaje es exitoso, se deben generar los siguientes archivos de salida:

- **contigs.fasta**: las secuencias contiguas (*contigs*)⁴ (en español, llamado menos frecuentemente como *cóntigo*) en formato FASTA⁵.
- **scaffolds.fasta**: las secuencias de *scaffolds*⁶ en formato FASTA.
- **assembly_graph.gfa**: grafo de ensamblaje y los caminos de los *scaffolds* en formato GFA 1.0⁷.
- **assembly_graph.fastg**: grafo del ensamblaje en formato FASTG.
- **contigs.paths**: caminos de los *contigs* en el grafo de ensamblaje.
- **scaffolds.paths**: caminos de los *scaffolds* en el grafo de ensamblaje.
- **spades.log**: archivo con todos los mensajes acerca del proceso de ensamblaje.

i Nota 5: Archivo **spades.log**

El archivo **spades.log** es un registro detallado del proceso de ensamblaje que realizó SPAdes. La información que contiene este archivo es clave para entender la forma en que se ejecutó el ensamblaje y detectar posibles fallos en el proceso. Por ejemplo, el archivo proporciona información acerca de los parámetros de SPAdes y de los recursos del sistema usados para el ensamblaje (versión, CPU, memoria RAM). También contiene información acerca de cada etapa del ensamblaje (lectura de datos, corrección de errores, construcción del grafo y ensamblaje final), tiempos de ejecución, y más importante, mensajes de advertencia y error. Estos mensajes pueden ser usados para detectar y corregir diversos problemas, incluyendo los relacionados con formato y calidad en los *reads* de entrada, recursos del sistema y parámetros incompatibles.

Para saber más acerca del uso del archivo log de SPAdes para identificación y solución de problemas consulte el Apéndice C.

Dado que SPAdes es un ensamblador de genomas basado en grafos de De Bruijn⁸ (Compeau et al., 2011), algunos de los archivos de salida contienen información de los grafos usados durante el ensamblaje (con extensión **.gfa** y **.fastg**). Si desea visualizar los archivos de grafos puede usar la aplicación Bandage⁹ (Wick et al., 2015).

Los archivos de salida más importantes son aquellos que contienen las secuencias del ensamblaje, es decir los *contigs* y *scaffolds* en formato FASTA (**.fasta**). Los *scaffolds* son las secuencias más largas de ensamblaje, las cuales son creadas a partir de las

⁴Wikipedia - Cóntigo: <https://es.wikipedia.org/wiki/C%C3%B3ntigo>

⁵Wikipedia - Formato FASTA: https://es.wikipedia.org/wiki/Formato_FASTA

⁶Wikipedia - Scaffolding (bioinformatics): [https://en.wikipedia.org/wiki/Scaffolding_\(bioinformatics\)](https://en.wikipedia.org/wiki/Scaffolding_(bioinformatics))

⁷The GFA Format Specification: <https://gfa-spec.github.io/GFA-spec/GFA1.html>

⁸Wikipedia - Grafo de De Bruijn: https://es.wikipedia.org/wiki/Grafo_de_De_Bruijn

⁹Bandage (GitHub): <http://rrwick.github.io/Bandage/>

secuencias de los *contigs*. Este es el archivo que se usará como resultado final del ensamblaje, y por lo tanto, el que será anotado.

A continuación, se muestran las primeras 10 líneas del archivo `scaffolds.fasta`:

```
head resultados/01_ensamblaje/SRR22388518/scaffolds.fasta

>NODE_1_length_407393_cov_61.379170
GCTCTGTTGAATGATTTATCCAAAAATCGTTTAGCAACTAGCAAGGTTGAACTAGGAATA
CAACAAAAATCCTAAATTAAGAGAATTTATAGCTTTTTATAGGATTTATAGGTTTGTA
GTAACGGTATTTTGGGCAAGAAATCAACCTGACCACTTACAACCTTTGCCGCATCAACAT
GTTTTTGTCATGACATCAATTACTCTAAATTCACATTGCGCTAGGGGACACGCTTTTAGA
TCCAAAACATGAAGACGATGAGCCTTTTGATGCGATCGTTTCAACCTCCTTATTCCAC
TAAATGGGTGGGCGATAACAACCTATTTTAATCAACGACGAGCGCTTTAGCCCGGCCGG
TGTGCTAGCGCCCAAAACGCGCCGATCTCGCTTTCACCATGCACATGCTTTCCTATTT
ATCCAATAGCGGCACGGCTGCGATCGTGAATTCCCGGGGTGCTTTATAGGGGAATGC
TGAAGCAAAATCAGAGAATTTTAGTCAAAGAAAATTTTCATTGACTGCGTGATCGCTTT
```

En el identificador (nombre) del primer *scaffold* (`NODE_1_length_407393_cov_61.379170`) se puede observar alguna información de la secuencia: **1** es el número del *contig/scaffold*, **407393** es la longitud de la secuencia en pares de bases (bp), y **61.379170** es el número de *k*-meros que cubren el *scaffold*.

Con un simple comando se puede averiguar el número total de *scaffolds* en el ensamblaje:

```
grep -c '>' resultados/01_ensamblaje/SRR22388518/scaffolds.fasta
```

336

4.3.3. Ensamblaje de las muestras restantes

Ahora, es necesario completar el procesamiento del conjunto de datos ensamblando las muestras restantes:

```
spades.py \
  --isolate \
  -1 resultados/00_datos_limpios/SRR22388519_1.clean.fastq.gz \
  -2 resultados/00_datos_limpios/SRR22388519_2.clean.fastq.gz \
  -o resultados/01_ensamblaje/SRR22388519 \
  -t 4

spades.py \
```

```
--isolate \  
-1 resultados/00_datos_limpios/SRR18335438_1.clean.fastq.gz \  
-2 resultados/00_datos_limpios/SRR18335438_2.clean.fastq.gz \  
-o resultados/01_ensamblaje/SRR18335438 \  
-t 4  
  
spades.py \  
--isolate \  
-1 resultados/00_datos_limpios/SRR18335439_1.clean.fastq.gz \  
-2 resultados/00_datos_limpios/SRR18335439_2.clean.fastq.gz \  
-o resultados/01_ensamblaje/SRR18335439 \  
-t 4
```

Para que cada ensamblaje ([scaffolds.fasta](#)) tenga el nombre de la correspondiente muestra, se crean enlaces simbólicos en el directorio [resultados/01_ensamblaje](#). Por ejemplo, dentro del directorio [resultados/01_ensamblaje](#) el archivo [SRR22388518.fasta](#) apuntará al archivo [SRR22388519/scaffolds.fasta](#). Esto se hace para que los pasos siguientes del manual (control de calidad y anotación) se ejecuten con mayor comodidad. Para crear los enlaces ejecute la siguientes órdenes:

```
ln -sr resultados/01_ensamblaje/SRR22388518/scaffolds.fasta \  
    resultados/01_ensamblaje/SRR22388518.fasta  
  
ln -sr resultados/01_ensamblaje/SRR22388519/scaffolds.fasta \  
    resultados/01_ensamblaje/SRR22388519.fasta  
  
ln -sr resultados/01_ensamblaje/SRR18335438/scaffolds.fasta \  
    resultados/01_ensamblaje/SRR18335438.fasta  
  
ln -sr resultados/01_ensamblaje/SRR18335439/scaffolds.fasta \  
    resultados/01_ensamblaje/SRR18335439.fasta
```

Verifique los enlaces creados:

```
ls -l resultados/01_ensamblaje  
  
total 16K  
drwxrwxr-x  9 hspitia hspitia 4.0K Mar 28  2023 SRR18335438  
lrwxrwxrwx  1 hspitia hspitia  27 Mar 28  2023 SRR18335438.fasta ->  
    SRR18335438/scaffolds.fasta  
drwxrwxr-x  9 hspitia hspitia 4.0K Mar 28  2023 SRR18335439  
lrwxrwxrwx  1 hspitia hspitia  27 Mar 28  2023 SRR18335439.fasta ->  
    SRR18335439/scaffolds.fasta  
drwxrwxr-x 11 hspitia hspitia 4.0K Mar 28  2023 SRR22388518
```

```
lrwxrwxrwx 1 hspitia hspitia 27 Mar 28 2023 SRR22388518.fasta ->
└─ SRR22388518/scaffolds.fasta
drwxrwxr-x 11 hspitia hspitia 4.0K Mar 28 2023 SRR22388519
lrwxrwxrwx 1 hspitia hspitia 27 Mar 28 2023 SRR22388519.fasta ->
└─ SRR22388519/scaffolds.fasta
```

Con todos los ensamblajes listos, se puede analizar su calidad.

4.4. Resumen

El ensamblaje consiste en la reconstrucción de un genoma de un organismo a partir de la organización de las lecturas de secuenciación. Existen dos tipos de ensamblaje según la manera en que se realiza esta tarea, guiado por referencia, y *de novo*. Este capítulo presenta el proceso de ensamblaje *de novo* de genomas bacterianos, enfocándose en la reconstrucción computacional de secuencias genómicas de *H. pylori* a partir de lecturas de secuenciación Illumina usando SPAdes. Se cubrió la ejecución práctica del ensamblador, el análisis de archivos de salida, y las herramientas de verificación del proceso, proporcionando las bases necesarias para obtener ensamblajes de calidad listos para evaluación y anotación funcional.



Capítulo 5.

Control de calidad de ensamblajes

La evaluación de calidad de ensamblajes es un proceso integrado que permite determinar la contigüidad, completitud biológica y fidelidad de secuencia de los genomas ensamblados. Este análisis utiliza métricas estadísticas, análisis de genes marcadores y comparación de espectros de k -meros para evaluar qué tan bien representa el ensamblaje al genoma original del organismo.

Este capítulo presenta un protocolo integral de control de calidad que combina la evaluación estructural con QUAST, el análisis de completitud biológica con BUSCO, y el análisis comparativo de k -meros con Jellyfish y KAT. Además, se abordan las principales estrategias de corrección y mejoramiento de ensamblajes, estableciendo los fundamentos para identificar y validar ensamblajes de calidad apropiada para análisis posteriores.

5.1. Software requerido

Tabla 5. Software requerido para control de calidad de ensamblajes.

Software	Versión	Descripción	Referencia
QUAST ¹²	5.2.0	Herramienta para evaluar la calidad de ensamblajes de genomas (pequeños) y metagenomas, mediante el cálculo de diferentes métricas.	Gurevich et al. (2013)
datasets ³⁴	18.4.1	Herramienta para acceder y descargar información genómica, incluyendo metadatos, desde las bases de datos del NCBI (del inglés <i>National Center for Biotechnology Information</i>)	O'Leary et al. (2024)

¹QUAST, sitio Web: <https://quast.sourceforge.net/quast>

²QUAST, repositorio GitHub: <https://github.com/ablab/quast>

³datasets, sitio Web: <https://www.ncbi.nlm.nih.gov/datasets/>

⁴datasets, repositorio GitHub: <https://github.com/ncbi/datasets>

Software	Versión	Descripción	Referencia
BUSCO ⁵⁶	5.5.0	Herramienta para la evaluación de la completitud biológica de ensamblajes.	Simão et al. (2015)
Jellyfish ⁷⁸	2.3.0	Herramienta rápida y eficiente para el conteo de <i>k</i> -meros en secuencias de ADN utilizando <i>hash tables</i> optimizados.	Marçais & Kingsford (2011)
KAT ⁹¹⁰	2.4.2	Conjunto de herramientas para análisis comparativo de <i>k</i> -meros, especialmente útil para evaluación de calidad de ensamblajes.	Mapleson et al. (2017)

5.2. Determinación de calidad

El cálculo de las métricas se realiza usando el software QUAST (del inglés *Quality Assessment Tool*). Aunque esta herramienta funciona sin un genoma de referencia, la evaluación de calidad será mejor si se provee uno. Este genoma podrá ser el del organismo más cercano o, como en nuestro caso, el del propio organismo (*H. pylori*¹¹). Dado que tenemos acceso al genoma de *H. pylori*, se usará como genoma de referencia.

5.2.1. Descarga del genoma de referencia

Para descargar el genoma de referencia se necesita su identificador en la base de datos pública, en este caso RefSeq¹² (Haft et al., 2018). En el caso del genoma de *H. pylori*, este identificador (*accession number*) es GCF_017821535.1. Para descargar el archivo del genoma se usa la herramienta datasets:

```
# Entrar al directorio datos del proyecto
cd $HOME/proyectos/ensamblaje/datos

# Descargar el paquete de datos con el genoma
```

⁵BUSCO, sitio Web: <https://busco.ezlab.org/>

⁶BUSCO, repositorio GitLab: <https://gitlab.com/ezlab/busco>

⁷Jellyfish, sitio Web: <http://www.genome.umd.edu/jellyfish.html>

⁸Jellyfish, repositorio GitHub: <https://github.com/gmarcais/Jellyfish>

⁹KAT, sitio Web: <https://kat.readthedocs.io/>

¹⁰KAT, repositorio GitHub: <https://github.com/TGAC/KAT>

¹¹NCBI Datasets Genome page of *Helicobacter pylori*: https://www.ncbi.nlm.nih.gov/labs/data-hub/genome/GCF_017821535.1/

¹²NCBI RefSeq - <https://www.ncbi.nlm.nih.gov/refseq/>

```
datasets download genome accession GCF_017821535.1 --filename
↳ GCF_017821535.1_dataset.zip
```

```
Collecting 1 records [=====]
↳ 100% 1/1
Downloading: GCF_017821535.1_dataset.zip 473kB done
```

Ahora se debe descomprimir el paquete de datos descargado que contiene el genoma:

```
unzip GCF_017821535.1_dataset.zip
```

```
Archive: GCF_017821535.1_dataset.zip
  inflating: README.md
  inflating: ncbi_dataset/data/assembly_data_report.jsonl
  inflating: ncbi_dataset/data/GCF_017821535.1/GCF_017821535.1_ASM178215
↳ 3v1_genomic.fna
  inflating: ncbi_dataset/data/dataset_catalog.json
```

Al descomprimir el paquete se generan dos elementos, el archivo `README.md` y el directorio `ncbi_dataset`. El archivo que se necesita es `ncbi_dataset/data/GCF_017821535.1/GCF_017821535.1_ASM1782153v1_genomic.fna`, el cual se renombrará y se moverá a una ubicación más apropiada:

```
mv ncbi_dataset/data/GCF_017821535.1/GCF_017821535.1_ASM1782153v1_genom
↳ ic.fna \
  GCF_017821535.1.fasta
```

En este momento, se pueden eliminar los archivos que no se emplearán:

```
rm -fr README.md ncbi_dataset GCF_017821535.1_dataset.zip
```

5.2.2. Ejecución de QUAST

```
cd $HOME/proyectos/ensamblaje

quast -o resultados/02_ensamblaje_qc/quast \
  -r datos/GCF_017821535.1.fasta \
  -t 4 \
  resultados/01_ensamblaje/*.fasta
```

Capítulo 5. Control de calidad de ensamblajes

```
/home/hector/mambaforge/envs/ensam/bin/quast -o
↳ resultados/02_ensamblaje_qc/quast -r datos/GCF_017821535.1.fasta -t 4
↳ resultados/01_ensamblaje/SRR18335438.fasta
↳ resultados/01_ensamblaje/SRR18335439.fasta
↳ resultados/01_ensamblaje/SRR22388518.fasta
↳ resultados/01_ensamblaje/SRR22388519.fasta
```

Version: 5.2.0

System information:

OS: Linux-5.15.0-67-generic-x86_64-with-glibc2.31 (linux_64)

Python version: 3.10.6

CPUs number: 8

Started: 2023-03-13 21:09:50

Logging to /home/hector/proyectos/ensamblaje/resultados/02_ensamblaje_qc/
↳ /quast/quast.log

NOTICE: Output directory already exists and looks like a QUASt output dir.

↳ Existing results can be reused (e.g. previously generated alignments)!

...

RESULTS:

Text versions of total report are saved to /home/hector/proyectos/ensam/

↳ blaje/resultados/02_ensamblaje_qc/quast/report.txt, report.tsv, and
↳ report.tex

Text versions of transposed total report are saved to

↳ /home/hector/proyectos/ensamblaje/resultados/02_ensamblaje_qc/quas/
↳ t/transposed_report.txt, transposed_report.tsv, and
↳ transposed_report.tex

HTML version (interactive tables and plots) is saved to /home/hector/pr/
↳ oyectos/ensamblaje/resultados/02_ensamblaje_qc/quast/report.html

PDF version (tables and plots) is saved to /home/hector/proyectos/ensam/
↳ blaje/resultados/02_ensamblaje_qc/quast/report.pdf

Icarus (contig browser) is saved to /home/hector/proyectos/ensamblaje/r/
↳ esultados/02_ensamblaje_qc/quast/icarus.html

Log is saved to /home/hector/proyectos/ensamblaje/resultados/02_ensamb/
↳ laje_qc/quast/quast.log

Finished: 2023-03-13 21:10:00

Elapsed time: 0:00:10.278012

NOTICES: 4; WARNINGS: 0; non-fatal ERRORS: 0

Thank you for using QUASt!

QUAST genera un reporte en diferentes formatos: texto (.txt y .tsv), LaTeX (.tex), html (.html) y pdf (.pdf).

5.3. Evaluación de calidad con QUAST

Se dará un vistazo al reporte .html ubicado en `ensamblaje/resultados/02_ensamblaje_qc/quast/report.html` (Figura 7).

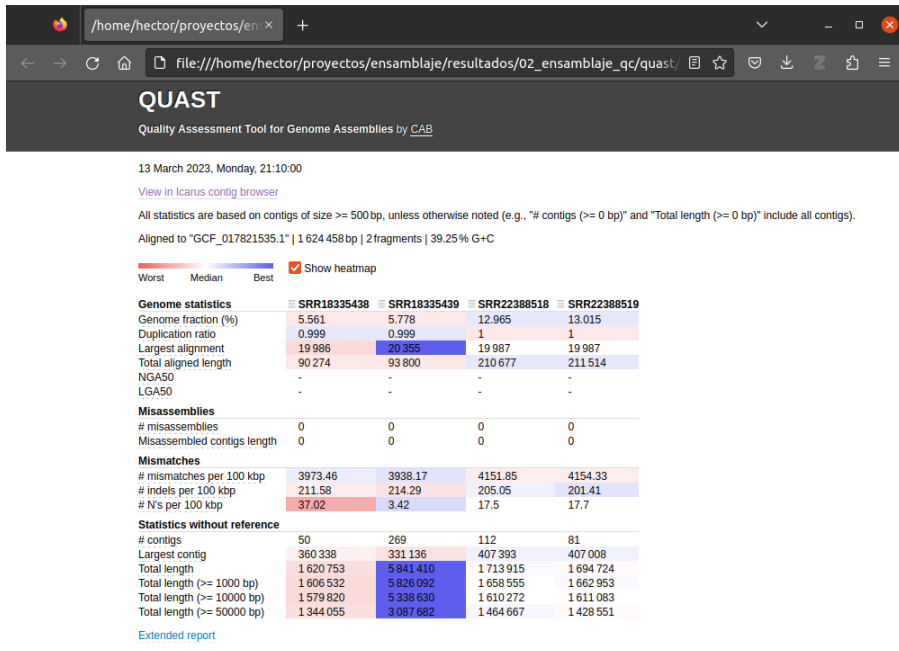


Figura 7. Reporte HTML generado por QUAST.

El reporte .html presenta una tabla con las estadísticas de los cuatro ensamblajes analizados. QUAST además adiciona colores (como mapa de calor) a la tabla para identificar rápidamente el mejor/peor (azul/rojo) ensamblaje con respecto de una estadística determinada.

 Advertencia

La guía del mapa de calor es solo visual y no implica que los ensamblajes sean calificados como buenos o malos, pues la calidad de un ensamblaje está dada por una combinación de métricas.

 Consejo

Puede ver la definición (en inglés) de cada estadística ubicando el puntero sobre el nombre correspondiente.

Si se hace clic en “*Extended report*” se verá aún más estadísticas, algunas de ellas bastante importantes. Entre las estadísticas más importantes para tener en cuenta se encuentran:

- **# contigs**: el número total de *contigs* con longitud mayor a 500 pb. Cuanto menor número de contigs obtenidos, mejor será el ensamblaje.
- **Total length**: la suma de las longitudes de los *contigs* en el ensamblaje. Es importante conocer el tamaño del genoma del organismo que se estudia para evaluar la calidad del ensamblaje. Se requiere que la longitud total obtenida sea cercana a la longitud del genoma del organismo estudiado. En este caso, el genoma de *H. pylori* tiene un tamaño estimado de 1.6 Mpb (Mega pares de bases o Mbp por *Mega base pairs* en inglés).
- **N50**: es la longitud x del *contig* más pequeño tal que el 50% de la longitud del ensamblaje está contenido en *contigs* más largos o iguales que x ¹³. Esta estadística está destinada a medir la contigüidad del ensamblaje. Generalmente se prefiere que esta estadística sea tan grande como sea posible. Teniendo en cuenta que depende de la longitud del ensamblaje, esta estadística puede ser ambigua cuando se desea comparar la calidad entre distintos ensamblajes con diferentes longitudes, pues no hay una referencia común de comparación.
- **NG50**: análoga al N50, esta estadística está calculada con base en un genoma de referencia para solucionar los problemas del N50 al comparar varios ensamblajes (Earl et al., 2011). Es la longitud x del *contig* más pequeño tal que el 50% de la longitud del **genoma de referencia** está contenido en *contigs* más largos o iguales que x ¹⁴.
- **L50**: el número de *contigs* con longitud mayor o igual a N50¹⁵.

¹³Wikipedia - Estadístico N50: https://es.wikipedia.org/wiki/Estad%C3%ADstico_N50.

¹⁴Wikipedia - Estadístico N50, Estadístico NG50: https://es.wikipedia.org/wiki/Estad%C3%ADstico_N50#Estad%C3%ADstico_NG50.

¹⁵Wikipedia - Estadístico N50, Estadístico L50: https://es.wikipedia.org/wiki/Estad%C3%ADstico_N50#Estad%C3%ADstico_L50.

- LG50: análogo al L50 pero con base en el genoma de referencia. El número de *contigs* con longitud mayor o igual a NG50.

i Nota 6: Valores aceptables en el contexto de *H. pylori*

Teniendo en cuenta el tamaño del genoma de *H. pylori* (organismo de referencia) en 1.6 Mpb, un ensamblaje aceptable debería tener valores cercanos a los siguientes:

- Número de *contigs* ≤ 100 : cuanto menor el número *contigs*, menor es la fragmentación.
- N50 ≥ 100 kpb (kilo pares de bases o kbp por *kilo base pairs* en inglés) y NG50 ≥ 100 kpb: *contigs* más largos denotan una mejor continuidad del genoma.
- L50/LG50 ≤ 5 : pocos *contigs* grandes contienen la mayoría del genoma
- Longitud total entre 1.5-1.7 Mpb: la longitud debe ser cercana a la longitud del genoma de referencia.

Adicionalmente, un ensamblaje aceptable debería presentar valores N50 y NG50 similares entre sí cuando la longitud total es cercana a la del genoma de referencia.

La Tabla 6 resume los valores obtenidos de las métricas para cada ensamblaje y su calidad general de acuerdo a ellas:

Tabla 6. Métricas de calidad de los ensamblajes de genoma de *H. pylori*.

Ensamblaje	# <i>contigs</i>	N50 (pb)	NG50 (pb)	L50	LG50	Longitud total (pb)	Calidad general
SRR18335438	50	158.719	158.719	3	3	1.620.753	Excelente
SRR18335439	269	59.322	144.740	30	4	5.841.410	Deficiente
SRR22388518	112	233.975	233.975	3	3	1.713.915	Buena
SRR22388519	81	156.397	156.397	4	4	1.694.724	Buena

Con base en las métricas evaluadas y el tamaño del genoma de *H. pylori*, se puede concluir que:

- [SRR18335438](#) es el ensamblaje de mejor calidad, con solo 50 *contigs*, una longitud total (1.620.753 pb) muy cercana al tamaño esperado del genoma de *H. pylori* (1.6 Mpb), y valores altos de N50 y NG50 (158.719 pb). Los valores bajos de L50 y LG50 (3) indican que se necesitan pocos *contigs* largos para cubrir el 50% del genoma, lo que sugiere una excelente contigüidad.

- [SRR18335439](#) presenta calidad deficiente debido a su alta fragmentación (269 *contigs*) y especialmente por su longitud total excesiva (5.841.410 pb), que es más de tres veces el tamaño esperado del genoma de *H. pylori*. Esta discrepancia sugiere la presencia de contaminantes o secuencias duplicadas que comprometen la utilidad del ensamblaje.
- [SRR22388518](#) presenta una calidad buena, con 112 *contigs* y la longitud total más alta (1.713.915 pb) pero aún dentro de un rango aceptable. Aunque con un número alto de *contigs*, su N50/NG50 de 233.975 pb es el más alto entre todos los ensamblajes, indicando *contigs* de mayor tamaño promedio.
- [SRR22388519](#) también muestra buena calidad con 81 *contigs*, longitud total apropiada (1.694.724 pb) y métricas de contigüidad aceptables (N50/NG50: 156.397 pb).

Un elemento útil para evaluar la longitud total de los ensamblajes respecto del genoma de referencia, es el gráfico de longitud acumulada de *contigs* generado por QUAST (Figura 8). Se puede corroborar en la Figura 8 que la longitud acumulada (total) del ensamblaje [SRR18335439](#) (línea azul) está muy por arriba del tamaño esperado del genoma de referencia (línea negra punteada). Por el contrario, las longitudes acumuladas de las otras muestras (líneas roja, verde, y violeta) son muy cercanas al tamaño del genoma de referencia.

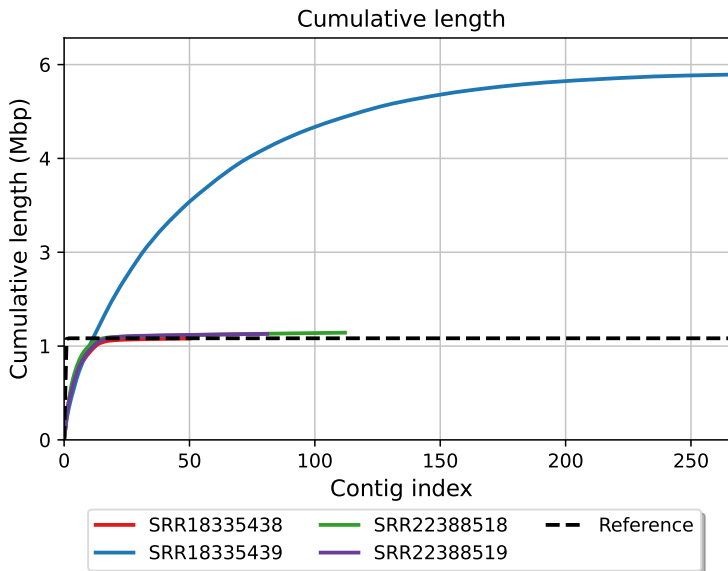


Figura 8. Gráfico generado por QUAST de la longitud acumulada de los ensamblajes.

En general, las métricas sugieren que los ensamblajes [SRR18335438](#), [SRR22388518](#) y [SRR22388519](#) representan genomas completos y de calidad adecuada para análisis posteriores, mientras que [SRR18335439](#) requiere corrección antes de poder ser utilizado confiablemente.

Es importante destacar que la muestra [SRR18335439](#) ilustra claramente cómo, a pesar de aplicar controles de calidad básicos a los datos de secuenciación, no siempre se garantiza la calidad del ensamblaje final. Debido a las deficiencias observadas en este ensamblaje, esta muestra será excluida de los análisis posteriores de anotación funcional.

5.4. Evaluación de completitud con BUSCO

Aunque las métricas básicas proporcionadas por QUAST permiten una evaluación de la contigüidad y estructura de los ensamblajes, no proporcionan información sobre la completitud biológica del ensamblaje en cuanto a sus genes.

La completitud biológica se refiere a qué tan representativo es el ensamblaje respecto al conjunto total de genes esperados para el organismo estudiado, es decir, si el genoma ensamblado contiene la mayoría de los genes esenciales y característicos de su especie.

BUSCO (del inglés *Benchmarking Universal Single-Copy Orthologs*, Simão et al. (2015)) complementa la evaluación de QUAST al evaluar la completitud biológica de los ensamblajes genómicos mediante la identificación de genes ortólogos universales de copia única. Estos genes están altamente conservados dentro de linajes específicos y se espera que estén presentes en la mayoría de las especies del grupo taxonómico. Para bacterias, BUSCO utiliza conjuntos de datos específicos como [bacteria_odb10](#) que contiene genes marcadores universales para procariotas, permitiendo determinar qué tan representativo es el ensamblaje respecto al conjunto total de genes esperados para el organismo estudiado. Además de bacteria, BUSCO tiene linajes específicos para otros grupos taxonómicos, como *Campylobacteriales* ([campylobacteriales_odb10](#)), que es el orden al que pertenece *H. pylori*.

Consejo

Si sabe todo el linaje taxonómico del organismo que está estudiando (dominio, reino, filo o división, clase, orden, familia, género, especie), es recomendable usar el conjunto de datos BUSCO específico para ese linaje, ya que proporcionará una evaluación más precisa de la completitud del ensamblaje. Puede generar la lista de linajes disponibles en BUSCO ejecutando el comando:

```
busco --list-datasets
```

A continuación busque en la lista el elemento que coincida con la categoría más específica de la taxonomía de su organismo de interés.

También puede delegar en BUSCO la selección del conjunto de datos más apropiado para su organismo usando el parámetro `--auto-lineage-prok` (para procariotas) o `--auto-lineage-euk` (para eucariotas) en lugar de usar el parámetro `-l`.

Para consultar la ayuda de BUSCO para más detalles sobre este y otros parámetros, ejecute el siguiente comando:

```
busco --help
```

5.4.1. Ejecución de BUSCO en la muestra SRR22388518

Para ejecutar BUSCO en el primer ensamblaje de *H. pylori*, se deben ejecutar los siguientes comandos:

```
# Entrar al directorio del proyecto
cd $HOME/proyectos/ensamblaje

# Ejecutar BUSCO para el primer ensamblaje
busco -i resultados/01_ensamblaje/SRR22388518.fasta \
-o SRR22388518_busco \
-m genome \
-l campylobacteriales_odb10 \
--out_path resultados/02_ensamblaje_qc/busco \
-c 4
```

Los parámetros utilizados en el comando `busco` son:

- `-i`: archivo de entrada con el ensamblaje a evaluar en formato FASTA
- `-o`: nombre del directorio de salida donde se almacenarán los resultados del análisis
- `-m genome`: modo de análisis, especifica que se está evaluando un genoma ensamblado
- `-l campylobacteriales_odb10`: conjunto de datos de linaje específico para *Campylobacteriales*, que contiene 628 genes marcadores universales

- `--out_path`: directorio padre donde se creará la carpeta de salida especificada en `-o`
- `-c 4`: número de hilos de CPU a utilizar para acelerar el procesamiento

Al ejecutar BUSCO, se obtiene la siguiente salida en la terminal:

```
2025-10-17 12:06:52 INFO:      ***** Start a BUSCO v5.5.0 analysis, current
↳ time: 10/17/2025 12:06:52 *****
2025-10-17 12:06:52 INFO:      Configuring BUSCO with local environment
2025-10-17 12:06:52 INFO:      Mode is genome
2025-10-17 12:06:52 INFO:      Downloading information on latest versions of
↳ BUSCO data...
2025-10-17 12:06:56 INFO:      Input file is /home/hector/proyectos/ensamblaje/
↳ je/resultados/01_ensamblaje/SRR22388518.fasta
2025-10-17 12:06:56 INFO:      Running BUSCO using lineage dataset
↳ campylobacteriales_odb10 (prokaryota, 2024-01-08)
2025-10-17 12:06:56 INFO:      Running 1 job(s) on bbtools, starting at
↳ 10/17/2025 12:06:56
2025-10-17 12:06:57 INFO:      [bbtools] 1 of 1 task(s) completed
2025-10-17 12:06:57 INFO:      ***** Run Prodigal on input to predict and
↳ extract genes *****
2025-10-17 12:06:57 INFO:      Running Prodigal with genetic code 11 in
↳ single mode
2025-10-17 12:06:57 INFO:      Running 1 job(s) on prodigal, starting at
↳ 10/17/2025 12:06:57
2025-10-17 12:06:59 INFO:      [prodigal] 1 of 1 task(s) completed
2025-10-17 12:06:59 INFO:      Genetic code 11 selected as optimal
2025-10-17 12:06:59 INFO:      ***** Run HMMER on gene sequences *****
2025-10-17 12:06:59 INFO:      Running 628 job(s) on hmmsearch, starting at
↳ 10/17/2025 12:06:59
2025-10-17 12:07:00 INFO:      [hmmsearch] 63 of 628 task(s) completed
2025-10-17 12:07:01 INFO:      [hmmsearch] 126 of 628 task(s) completed
2025-10-17 12:07:01 INFO:      [hmmsearch] 189 of 628 task(s) completed
2025-10-17 12:07:01 INFO:      [hmmsearch] 252 of 628 task(s) completed
2025-10-17 12:07:02 INFO:      [hmmsearch] 314 of 628 task(s) completed
2025-10-17 12:07:02 INFO:      [hmmsearch] 377 of 628 task(s) completed
2025-10-17 12:07:03 INFO:      [hmmsearch] 440 of 628 task(s) completed
2025-10-17 12:07:03 INFO:      [hmmsearch] 503 of 628 task(s) completed
2025-10-17 12:07:04 INFO:      [hmmsearch] 566 of 628 task(s) completed
2025-10-17 12:07:05 INFO:      [hmmsearch] 628 of 628 task(s) completed
2025-10-17 12:07:05 INFO:      Results:
↳ C:99.8%,S:99.8%,D:0.0%,F:0.0%,M:0.2%,n:628

2025-10-17 12:07:05 INFO:

-----
|Results from dataset campylobacteriales_odb10      |
```

```

-----
|C:99.8%[S:99.8%,D:0.0%],F:0.0%,M:0.2%,n:628      |
|627   Complete BUSCOs (C)                        |
|627   Complete and single-copy BUSCOs (S)         |
|0     Complete and duplicated BUSCOs (D)          |
|0     Fragmented BUSCOs (F)                      |
|1     Missing BUSCOs (M)                        |
|628   Total BUSCO groups searched                |
-----
2025-10-17 12:07:05 INFO:   BUSCO analysis done. Total running time: 9
    ↳ seconds
2025-10-17 12:07:05 INFO:   Results written in /home/hector/proyectos/ens_
    ↳ amblaje/resultados/02_ensamblaje_qc/busco/SRR22388518_busco
2025-10-17 12:07:05 INFO:   For assistance with interpreting the results,
    ↳ please consult the userguide:
    ↳ https://busco.ezlab.org/busco_userguide.html

2025-10-17 12:07:05 INFO:   Visit this page
    ↳ https://gitlab.com/ezlab/busco#how-to-cite-busco to see how to cite
    ↳ BUSCO

```

5.4.2. Interpretación de la salida de BUSCO

La salida de BUSCO presenta mensajes acerca del progreso del análisis, incluyendo la predicción de genes y la búsqueda de genes ortólogos con los programas Prodigal (Hyatt et al., 2010) y HMMER¹⁶, respectivamente. Al final del análisis, se muestra un resumen de los resultados (empezando en la línea **INFO**). En el resumen de los resultados se observa que BUSCO encontró 627 genes completos (C:99.8%[S:99.8%,D:0.0%],F:0.0%,M:0.2%) de un total de 628 genes esperados para el linaje *Campylobacteriales*. Esto indica que el ensamblaje es altamente completo, con solo un gen faltante (M:0.2%) y ningún gen fragmentado (F:0.0%).

Además de la salida por terminal, BUSCO genera varios archivos y carpetas en el directorio de salida especificado (**resultados/02_ensamblaje_qc/busco/SRR22388518_busco**):

- **logs**: carpeta que contiene los archivos de registro del análisis.
- **prodigal_output**: carpeta que contiene las secuencias de genes predichos por Prodigal.
- **run_campylobacteriales_odb10**: carpeta que contiene archivos intermedios y resultados detallados del análisis.
- **short_summary_specific_campylobacteriales_odb10.SRR22388518_busco.json**: resumen conciso de los resultados en formato JSON.

¹⁶HMMER (GitHub): <https://github.com/EddyRivasLab/hmmer>

- `short_summary.specific.campylobacterales_odb10.SRR22388518_busco.txt`: resumen conciso de los resultados en formato de texto.

Al interior del directorio `run_campylobacterales_odb10`, se encuentran archivos y directorios adicionales que detallan los genes encontrados y otros datos relevantes para el análisis. El archivo más relevante aquí es `full_table.tsv` que contiene toda la información acerca de los genes esperados para el linaje seleccionado para el análisis: identificador (columna `# Busco id`), clasificación (columna `Status` con valores `Complete` para completo, `Fragmented` para fragmentado, y `Missing` para faltante), la ubicación en el *contig* del ensamblaje en donde se encontró, además de la descripción.

5.4.3. Ejecución de BUSCO en las muestras restantes

Ahora, se debe ejecutar BUSCO para los tres ensamblajes restantes. El siguiente bloque de código muestra los comandos necesarios para cada uno de ellos:

```
busco -i resultados/01_ensamblaje/SRR22388519.fasta \
-o SRR22388519_busco \
-m genome \
-l campylobacterales_odb10 \
--out_path resultados/02_ensamblaje_qc/busco \
-c 4

busco -i resultados/01_ensamblaje/SRR18335438.fasta \
-o SRR18335438_busco \
-m genome \
-l campylobacterales_odb10 \
--out_path resultados/02_ensamblaje_qc/busco \
-c 4

busco -i resultados/01_ensamblaje/SRR18335439.fasta \
-o SRR18335439_busco \
-m genome \
-l campylobacterales_odb10 \
--out_path resultados/02_ensamblaje_qc/busco \
-c 4
```

5.4.4. Unificación de reportes BUSCO

Con todos los ensamblajes evaluados con BUSCO, se puede generar un reporte unificado de calidad con MultiQC que contenga los resultados de QUAST y BUSCO:

```
multiqc --filename multiqc_report.html \
--outdir resultados/02_ensamblaje_qc/ \
resultados/02_ensamblaje_qc/
```

MultiQC busca automáticamente los resultados de QUAST y BUSCO en el directorio especificado `resultados/02_ensamblaje_qc/` y genera un reporte consolidado en formato HTML en la ubicación especificada (`resultados/02_ensamblaje_qc/multiqc_report.html`). La Figura 9 muestra una sección del reporte de MultiQC con los resultados de QUAST y BUSCO para los cuatro ensamblajes evaluados.

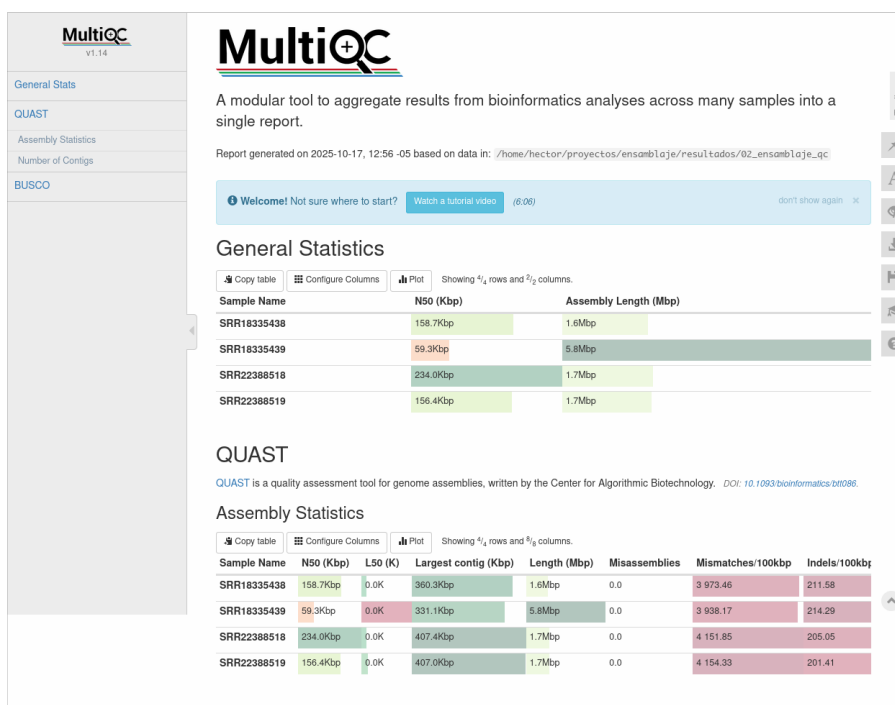


Figura 9. Sección del reporte de MultiQC con los resultados de QUAST y BUSCO.

5.4.5. Interpretación de los resultados de BUSCO

BUSCO clasifica los genes encontrados en cinco categorías: C, S, D, F y M. Las categorías están definidas de la siguiente manera:

- Completos(C, *Complete*): genes encontrados con longitud esperada

- Completos de copia única (S, *Complete and single-copy*): encontrados una sola vez
- Completos duplicados (D, *Complete and duplicated*): encontrados múltiples veces
- Fragmentados (F, *Fragmented*): genes encontrados parcialmente
- Faltantes (M, *Missing*): genes no encontrados

i Nota 7: Valores de referencia para BUSCO en bacterias

Los resultados de BUSCO se presentan como porcentajes. Para genomas bacterianos de alta calidad se esperan valores típicos de la siguiente manera:

- Completitud total (C): >95%
- Genes de copia única (S): >90%
- Genes duplicados (D): <5%
- Genes fragmentados (F): <3%
- Genes faltantes (M): <5%

La Tabla 7 muestra los resultados de BUSCO para cada ensamblaje.

Tabla 7. Resultados de BUSCO para los ensamblajes de genoma de *H. pylori* con respecto del linaje de los *Campylobacteriales* con un total de 682 genes marcadores.

Ensamblaje	Completos	Completos, copia única	Completos, duplicados	Fragmentados	Faltantes	Total
SRR18335438	622 (99,0%)	622 (99,0%)	0 (0,0%)	3 (0,5%)	3 (0,5%)	628 (100%)
SRR18335439	623 (99,2%)	615 (97,9%)	8 (1,3%)	3 (0,5%)	2 (0,3%)	628 (100%)
SRR22388518	627 (99,8%)	627 (99,8%)	0 (0,0%)	0 (0,0%)	1 (0,2%)	628 (100%)
SRR22388519	627 (99,8%)	625 (99,5%)	2 (0,3%)	0 (0,0%)	1 (0,2%)	628 (100%)

i Nota 8: Análisis integrado de calidad con QUAST y BUSCO

La combinación de métricas QUAST y BUSCO proporciona una evaluación más completa. La siguiente tabla resume las métricas más relevantes para la evaluación de calidad:

Tabla 8. Criterios integrados para evaluación de calidad de ensamblajes bacterianos.

Criterio	Excelente	Bueno	Aceptable	Deficiente
N50 (bp)	>100,000	50,000-100,000	20,000-50,000	<20,000

# contigs	<50	50-150	150-300	>300
BUSCO Completo (%)	>98	95-98	90-95	<90
BUSCO Faltante (%)	<2	2-5	5-10	>10

Al revisar los resultados de BUSCO en la Tabla 7 se pueden hacer las siguientes observaciones:

- **SRR18335438** presenta un ensamblaje de excelente calidad con 99.0% de genes completos, 0.5% fragmentados y 0.5% faltantes, complementando sus métricas de QUAST que indican alta contigüidad y longitud adecuada.
- **SRR18335439** muestra un ensamblaje de calidad aceptable con 99.2% de genes completos, 0.5% fragmentados y 0.3% faltantes. Sin embargo, sus métricas de QUAST indican una longitud excesiva y alta fragmentación, sugiriendo posibles problemas en el ensamblaje que requieren corrección antes de análisis posteriores. A pesar de la completitud, su fragmentación y calidad deficiente de métricas QUAST confirma la decisión de excluirlo de los análisis de anotación.
- **SRR22388518** también presenta un ensamblaje de excelente calidad con 99.8% de genes completos y solo 0.2% faltantes, alineándose con sus métricas de QUAST que indican buena contigüidad y longitud cercana al genoma esperado.
- **SRR22388519** muestra un ensamblaje de buena calidad con 99.8% de genes completos y 0.2% faltantes, complementando sus métricas de QUAST que indican buena contigüidad y longitud adecuada.

5.5. Análisis de k -meros con Jellyfish y KAT

Las valoraciones con QUAST y BUSCO son comúnmente las más utilizadas para evaluar la calidad de los ensamblajes. Sin embargo, existen métodos alternativos que aportan información útil para evaluación de calidad desde perspectivas distintas a las métricas clásicas de estructura (p. ej. N50, L50, tamaño del genoma, etc.) y completitud. Un método alternativo sencillo de ejecutar es el análisis de consistencia con los datos de entrada basado en k -meros¹⁷ (también kámeros o k -mers en inglés y mucho más utilizado), que son subsecuencias de longitud k (Figura 10).

i Nota 9: Espectro de k -meros

El espectro de k -meros se puede definir como la distribución de frecuencias de todos los k -meros únicos presentes en un conjunto de lecturas o en un genoma

¹⁷Wikipedia - K-mero: <https://es.wikipedia.org/wiki/K-mero>

(o ensamblaje). El espectro se calcula realizando primero el conteo de k -meros: para cada k -mero único encontrado en las secuencias, se cuenta el número de veces que aparece. Luego, se realiza la distribución de frecuencias: se agrupa los k -meros según su frecuencia de aparición (p. ej. k -meros que aparecen 1 vez, 2 veces, 10 veces, etc.). Por último, se realiza la representación del espectro: se presenta como un histograma donde el eje X representa la frecuencia de aparición y el eje Y representa el número de k -meros únicos con esa frecuencia.

Por ejemplo, si un k -mero específico “ATCGATGATCG” aparece 50 veces en las lecturas, contribuye al conteo de k -meros con frecuencia 50. El espectro completo muestra cuántos k -meros diferentes tienen frecuencia (o multiplicidad) 1, frecuencia 2, frecuencia 50, etc.

El espectro de k -meros refleja características fundamentales del genoma y la calidad de los datos. Por ejemplo, picos de frecuencia alta corresponden a regiones del genoma con cobertura uniforme, frecuencias bajas pueden indicar errores de secuenciación o regiones de baja cobertura, y múltiples picos pueden sugerir contaminación, duplicaciones genómicas o heterocigosidad.

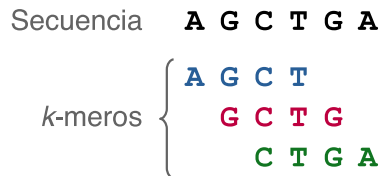


Figura 10. Ejemplo de k -meros. En la secuencia AGCTGA (en negro) de seis nucleótidos (6 pb), se pueden formar en total tres k -meros de tamaño 4 ($k = 4$; 4-meros o 4-mers en inglés): AGCT (azul), GCTG (rojo), y CTGA (verde).

5.5.1. Análisis de la muestra SRR22388518 y su ensamblaje

5.5.1.1. Análisis de consistencia entre espectros

Este análisis requiere dos pasos principales: la caracterización del contenido de k -meros para obtener los espectros de las lecturas y del ensamblaje, y luego su comparación. El espectro de las lecturas sirve como referencia y se compara contra el espectro del ensamblaje. La similitud o consistencia entre los espectros sugiere un ensamblaje de buena calidad, mientras que la discrepancia entre ellos sugiere problemas en el ensamblaje como información faltante o regiones mal ensambladas. A continuación se describe el procedimiento para realizar este análisis utilizando las herramientas Jellyfish y KAT.

Antes de caracterizar (contar) los k -meros de las secuencias, se deben descomprimir las lecturas limpias generadas previamente y almacenadas en el directorio `resultados/00_datos_limpios/` dado que Jellyfish no acepta archivos fastq comprimidos:

```
gunzip resultados/00_datos_limpios/*.clean.fastq.gz
```

! Importante

En el anterior comando se usó el comodín `*` (*wildcard* en inglés) para seleccionar todos los archivos que terminan con `.clean.fastq.gz` en el directorio especificado. El asterisco (`*`) se interpreta en este caso como “cualquier carácter que aparezca cero o más veces” seguido de `.clean.fastq.gz`, lo que resulta en la coincidencia de todos los archivos en el directorio `resultados/00_datos_limpios/` que tengan la terminación `.clean.fastq.gz`.

Usar el comodín permite en este caso descomprimir múltiples archivos de lecturas limpias en una sola línea de comando, facilitando el proceso cuando se tienen varias muestras o archivos de secuenciación.

5.5.1.1.1. Conteo de k -meros con Jellyfish

Jellyfish es una herramienta eficiente para el conteo de k -meros en secuencias de ADN. Su funcionamiento se basa en una tabla hash (*hash table* en inglés) optimizado que permite procesar grandes volúmenes de datos de secuenciación de manera rápida y eficiente.

Para realizar el conteo en las lecturas originales se ejecuta el subcomando `count` de Jellyfish:

```
# Conteo de k-mers (k=21) en las lecturas originales
jellyfish count \
  -m 21 \
  -s 100M \
  -t 4 \
  -C \
  -o resultados/02_ensamblaje_qc/kmers/SRR22388518_reads_kmers.jf \
  resultados/00_datos_limpios/SRR22388518/*.clean.fastq
```

! Importante

La ejecución de este comando solo toma unos segundos y no se genera ningún mensaje de salida de Jellyfish.

Los parámetros usados en `jellyfish count` son:

- `-m 21`: longitud del k -mero (21 nucleótidos). Este valor representa un equilibrio entre especificidad y sensibilidad para genomas bacterianos
- `-s 100M`: tamaño estimado del *hash table* (100 millones de entradas). Debe ser suficientemente grande para almacenar todos los k -meros únicos
- `-t 4`: número de hilos de procesamiento para acelerar el cálculo
- `-C`: considerar ambas cadenas de ADN (*forward* y *reverse complement*), tratando las secuencias como canónicas
- `-o`: archivo de salida para almacenar el conteo de k -meros en formato binario optimizado

A continuación se verifica la generación del archivo especificado listando el contenido en el directorio de salida con `ls`:

```
ls resultados/02_ensamblaje_qc/kmers  
  
SRR22388518_reads_kmers.jf
```

Ahora se debe realizar el conteo de los k -meros para el ensamblaje correspondiente:

```
# Conteo de k-meros en el ensamblaje  
jellyfish count \  
-m 21 \  
-s 100M \  
-t 4 \  
-C \  
-o resultados/02_ensamblaje_qc/kmers/SRR22388518_assembly_kmers.jf \  
resultados/01_ensamblaje/SRR22388518.fasta
```

5.5.1.1.2. Análisis comparativo con KAT

KAT (K-mer Analysis Toolkit) proporciona herramientas especializadas para comparar espectros de k -meros entre diferentes conjuntos de datos. Esta herramienta es útil para evaluar la calidad del ensamblaje mediante la comparación directa de k -meros.

Para realizar el análisis de comparación de los espectros generados anteriormente se debe ejecutar subcomando `comp` de KAT:

```
# Análisis comparativo de k-mers entre lecturas y ensamblaje
kat comp \
  -t 4 \
  -o resultados/02_ensamblaje_qc/kmers/SRR22388518_comparison \
  resultados/02_ensamblaje_qc/kmers/SRR22388518_reads_kmers.jf \
  resultados/02_ensamblaje_qc/kmers/SRR22388518_assembly_kmers.jf
```

Los parámetros usados en `kat comp` son:

- `-t 4`: número de hilos de procesamiento
- `-o`: prefijo para archivos de salida
- El primer argumento es el archivo de conteo de lecturas originales
- El segundo argumento es el archivo de conteo del ensamblaje correspondiente

La salida generada por KAT es la siguiente:

```
Kmer Analysis Toolkit (KAT) V2.4.2

Running KAT in COMP mode
-----

Loading hashes into memory...Loading hashes into memory...Loading hashes
↳ into memory... done. Time taken: 0.6s

done. Time taken: 8.5s

done. Time taken: 8.5s

Comparing hashes ... done. Time taken: 5.1s

Merging results ... done. Time taken: 0.4s

Saving results to disk ... done. Time taken: 0.2s

Summary statistics
-----

K-mer statistics for:
- Hash 1: "resultados/02_ensamblaje_qc/kmers/SRR22388518_reads_kmers.jf"
- Hash 2:
  ↳ "resultados/02_ensamblaje_qc/kmers/SRR22388518_assembly_kmers.jf"
```

Total K-mers in:

- Hash 1: 377153518
- Hash 2: 1798759

Distinct K-mers in:

- Hash 1: 44953126
- Hash 2: 1779709

Total K-mers only found in:

- Hash 1: 52762816
- Hash 2: 0

Distinct K-mers only found in:

- Hash 1: 43173417
- Hash 2: 0

Shared K-mers:

- Total shared found in hash 1: 324390702
- Total shared found in hash 2: 1798759
- Distinct shared K-mers: 1779709

Distance between spectra 1 and 2 (all k-mers):

- Manhattan distance: 4.31734e+07
- Euclidean distance: 3.62224e+07
- Cosine distance: 0.00534122
- Canberra distance: 721.876
- Jaccard distance: 0.96041

Distance between spectra 1 and 2 (shared k-mers):

- Manhattan distance: 3.38119e+06
- Euclidean distance: 1.70015e+06
- Cosine distance: 0.622276
- Canberra distance: 719.367
- Jaccard distance: 0.974321

Creating plot(s) ... done. Time taken: 2.2s

Analysing peaks for spectra copy number matrix

Analysing distributions for:

↳ resultados/02_ensamblaje_qc/kmers/SRR22388518_comparison-main.mx ...

Analysing full spectra

Analysing spectra with copy number 0

Can't optimise peaks because none are defined.

Analysing spectra with copy number 1

Analysing spectra with copy number 2
 Can't optimise peaks because none are defined.
 Analysing spectra with copy number 3
 Can't optimise peaks because none are defined.
 Analysed spectra for all requested copy numbers.
 done. Time taken: 2.3s

Main spectra statistics

 K-value used: 21
 Peaks in analysis: 2
 Global minima @ Frequency=48x (45)
 Global maxima @ Frequency=168x (28577)
 Overall mean k-mer frequency: 173x

Index	Left	Mean	Right	StdDev	Max	Volume	Description
1	123.72	168.59	213.45	22.43	28064	1578151	1X
2	25.85	334	642.15	154.08	162	53018	2X

Calculating genome statistics

 Assuming that homozygous peak is the largest in the spectra with frequency
 ↳ of: 168x
 Homozygous peak index: 1
 CAUTION: the following estimates are based on having a clean spectra and
 ↳ having identified the correct homozygous peak!
 Estimated genome size: 1.68 Mbp
 Estimated assembly completeness: 100.00%

Breakdown of copy number composition for each peak

 ---- Report for f=168.587 (total elements 1575558)----
 0x: No significant content
 1x: 100.00% (1575558 elements at f=168.61)
 2x: No significant content
 3x: No significant content

 ---- Report for f=334.000 (total elements 35843)----
 0x: No significant content
 1x: 100.00% (35843 elements at f=334.00)
 2x: No significant content
 3x: No significant content

Creating plots

```
-----  
Plotting K-mer frequency distributions for general spectra ... done.  
Plotting K-mer frequency distributions for 1x ... done.
```

```
KAT COMP completed.  
Total runtime: 19.4s
```

Además de la salida en la terminal, `kat comp` genera cuatro archivos en la ubicación especificada (`resultados/02_ensamblaje_qc/kmers`):

- `SRR22388518_comparison-main.mx`: archivo binario con la matriz de número de copias principal
- `SRR22388518_comparison.stats`: archivo de texto con estadísticas resumidas (mostrado arriba)
- `SRR22388518_comparison-main.mx.spectra-cn.png`: gráfico PNG del espectro la matriz de número de copias
- `SRR22388518_comparison.dist_analysis.json`: archivo JSON con análisis de distancia detallado

5.5.1.2. Análisis de la salida de `kat comp`

5.5.1.2.1. Resumen impreso en la terminal

La salida de `kat comp` proporciona múltiples secciones de información para evaluar la calidad del ensamblaje, entre las que se pueden resaltar:

- Estadísticas de resumen (*Summary statistics*):
 - Total de *k*-meros (*Total K-mers*): número total de *k*-meros en lecturas (377,153,518) vs. ensamblaje (1,798,759).
 - *K*-meros únicos (*Distinct K-mers*): número de *k*-meros únicos en cada conjunto.
 - Total de *k*-meros únicos solo en (*Total K-mers only found in*): total de *k*-meros presentes solo en lecturas (52,762,816) o solo en ensamblaje (0).
 - *K*-meros únicos solo en (*Distinct K-mers only found in*): *k*-meros únicos presentes en lecturas (43,173,417) o ensamblaje (0).
 - *K*-meros compartidos: *k*-meros presentes en ambos conjuntos (324,390,702 en lecturas; 1,798,759 en ensamblaje; 1,779,709 únicos).

- Distancias entre espectros (*Distance between spectra*): diferentes métricas de distancia que cuantifican la similitud entre los espectros. Generalmente, valores menores de distancia indican mayor similitud entre lecturas y ensamblaje. Cabe resaltar que las distancias no son comparables entre sí (ver Nota 10). La distancia de Jaccard (0.96041) sugiere una diferencia considerable, probablemente debido a k -meros de baja frecuencia en las lecturas.
- Estadísticas principales del espectro (*Main spectra statistics*):
 - *K-value*: longitud de k -mero utilizada (21).
 - Picos identificados (*Peaks*): 2 picos principales en el espectro de frecuencias.
 - Frecuencia promedio (*Overall mean k-mer frequency*): 173x de cobertura promedio de k -meros.
 - Descripción de picos (*Peak description*): pico principal a 168.59x (1X) y pico secundario a 334x (2X).
- Estadísticas del genoma (*Calculating genome statistics*):
 - Tamaño estimado del genoma (*Estimated genome size*): 1.68 Mpb (muy cercano al tamaño esperado de *H. pylori*).
 - Completitud estimada del ensamblaje (*Estimated assembly completeness*): 100.00% (excelente).
 - Estas estimaciones se basan en el pico homocigoto identificado a 168x de cobertura.

Esta salida indica que el ensamblaje SRR22388518 es de buena calidad, con completitud estimada del 100% y un tamaño genómico que coincide con las expectativas para *H. pylori*. La presencia de k -meros únicos en las lecturas puede deberse a errores de secuenciación o regiones de muy baja cobertura no ensambladas.

i Nota 10: Métricas de distancia utilizadas por KAT

KAT calcula diferentes métricas de distancia para cuantificar las diferencias entre los espectros de k -meros. Estas métricas no son directamente comparables entre sí, y en cambio, cada una proporciona una perspectiva diferente sobre la calidad del ensamblaje. Las siguientes son las métricas:

- Distancia de Manhattan: suma de las diferencias absolutas entre las frecuencias de k -meros correspondientes. Es sensible a diferencias en todos los k -meros pero no considera las magnitudes relativas.

- Distancia Euclidiana (*Euclidean*): raíz cuadrada de la suma de las diferencias al cuadrado. Penaliza más las diferencias grandes que las pequeñas, siendo útil para detectar cambios significativos en el espectro.
- Distancia del Coseno (*Cosine*): mide el ángulo entre los vectores de frecuencia de k -meros. Valores cercanos a 0 indican espectros similares, independientemente de sus magnitudes absolutas.
- Distancia de Canberra: suma ponderada de diferencias absolutas normalizadas por las sumas de frecuencias. Es especialmente sensible a diferencias cerca del origen y menos influenciada por valores altos.
- Distancia de Jaccard: basada en la teoría de conjuntos, mide la disimilitud como $1 - (\text{intersección}/\text{unión})$. Para k -meros, refleja qué proporción del espectro total no es compartida entre lecturas y ensamblaje. Es particularmente útil para evaluar completitud.

5.5.1.2.2. Gráfico de comparación de k -meros

Un archivo útil para la interpretación visual es el gráfico de número de copias del espectro ([SRR22388518_comparison-main.mx.spectra-cn.png](#)), que muestra la distribución de frecuencias de k -meros en las lecturas y el ensamblaje (Figura 11). En este gráfico, el eje X representa la frecuencia de los k -meros (multiplicidad) y el eje Y representa el número de k -meros únicos con esa frecuencia. Cada color en el gráfico representa a qué “número de copia” (*copy-number* en inglés) del ensamblaje pertenecen esos k -meros de las lecturas. Por ejemplo, 0x (negro), representa los k -meros presentes en lecturas pero ausentes en el ensamblaje (contenido perdido). A su vez, 1x, 2x, 3x, Nx (paleta), representan los k -meros de lecturas que se encuentran 1, 2, 3, N veces en el ensamblaje (contenido colapsado, correcto o duplicado según el caso).

Al analizar el gráfico se pueden hacer la siguientes observaciones:

- Pico principal en multiplicidad ~150–175: este es el pico de k -meros verdaderos, es decir, los que aparecen consistentemente en las lecturas. La mayoría de estos k -meros están en rojo (1x), lo cual indica que fueron ensamblados una sola vez. Esto es bueno considerando que *H. pylori* es un organismo haploide (que tiene un solo conjunto completo de cromosomas).
- Presencia de negro (0x) en multiplicidades medias (entre 10 y 25): hay una cantidad considerable de k -meros con multiplicidad alta que no están en el ensamblaje. Esto sugiere que existe contenido genuino que fue omitido, lo cual

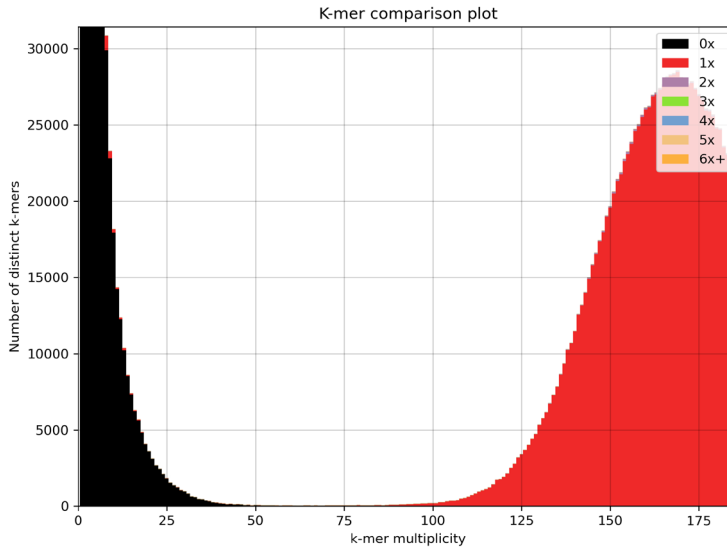


Figura 11. Espectro de número de copias generado por KAT para la muestra SRR22388518.

podría corresponder con regiones repetitivas, variantes alélicas, o simplemente *k*-meros que no fueron ensamblados por errores de secuenciación o filtros.

- Colores de número de copia (*copy-number*) $\geq 2x$ (violeta, verde, azul, amarillo, naranja): hay una fracción de *k*-meros que aparecen más de una vez en el ensamblaje. Esto puede indicar duplicación de *contigs*, o regiones repetidas (p. ej. de RNA ribosomal o transposones) no colapsadas, es decir, que el ensamblador no unificó en una sola copia.

5.5.2. Análisis de *k*-meros en las muestras restantes

Una vez completado el análisis detallado de la primera muestra, se procede a procesar las muestras restantes utilizando el mismo flujo de trabajo:

```
# Procesar muestra SRR22388519
# Conteo en lecturas
jellyfish count \
  -m 21 \
  -s 100M \
  -t 4 \
```

```
-C \  
-o resultados/02_ensamblaje_qc/kmers/SRR22388519_reads_kmers.jf \  
resultados/00_datos_limpios/SRR22388519_*.clean.fastq  
  
# Conteo en ensamblaje  
jellyfish count \  
-m 21 \  
-s 100M \  
-t 4 \  
-C \  
-o resultados/02_ensamblaje_qc/kmers/SRR22388519_assembly_kmers.jf \  
resultados/01_ensamblaje/SRR22388519.fasta  
  
# Análisis comparativo con KAT  
kat comp \  
-t 4 \  
-o resultados/02_ensamblaje_qc/kmers/SRR22388519_comparison \  
resultados/02_ensamblaje_qc/kmers/SRR22388519_reads_kmers.jf \  
resultados/02_ensamblaje_qc/kmers/SRR22388519_assembly_kmers.jf  
  
# Procesar muestra SRR18335438  
# Conteo en lecturas  
jellyfish count \  
-m 21 \  
-s 100M \  
-t 4 \  
-C \  
-o resultados/02_ensamblaje_qc/kmers/SRR18335438_reads_kmers.jf \  
resultados/00_datos_limpios/SRR18335438_*.clean.fastq  
  
# Conteo en ensamblaje  
jellyfish count \  
-m 21 \  
-s 100M \  
-t 4 \  
-C \  
-o resultados/02_ensamblaje_qc/kmers/SRR18335438_assembly_kmers.jf \  
resultados/01_ensamblaje/SRR18335438.fasta  
  
# Análisis comparativo con KAT  
kat comp \  
-t 4 \  
-o resultados/02_ensamblaje_qc/kmers/SRR18335438_comparison \  
resultados/02_ensamblaje_qc/kmers/SRR18335438_reads_kmers.jf \  
resultados/02_ensamblaje_qc/kmers/SRR18335438_assembly_kmers.jf
```

```
# Procesar muestra SRR18335439
# Conteo en lecturas
jellyfish count \
-m 21 \
-s 100M \
-t 4 \
-C \
-o resultados/02_ensamblaje_qc/kmers/SRR18335439_reads_kmers.jf \
resultados/00_datos_limpios/SRR18335439_*.clean.fastq

# Conteo en ensamblaje
jellyfish count \
-m 21 \
-s 100M \
-t 4 \
-C \
-o resultados/02_ensamblaje_qc/kmers/SRR18335439_assembly_kmers.jf \
resultados/01_ensamblaje/SRR18335439.fasta

# Análisis comparativo con KAT
kat comp \
-t 4 \
-o resultados/02_ensamblaje_qc/kmers/SRR18335439_comparison \
resultados/02_ensamblaje_qc/kmers/SRR18335439_reads_kmers.jf \
resultados/02_ensamblaje_qc/kmers/SRR18335439_assembly_kmers.jf
```

5.5.3. Interpretación de los resultados de KAT

La Figura 12 presenta los gráficos de espectro de número de copia generados por KAT para cada uno de las muestras restantes, junto con el espectro de [SRR22388518](#) (Figura 12a) previamente analizado para facilitar la comparación.

La Tabla 9 resume los patrones observados en los análisis de k -meros para todas las muestras:

Tabla 9. Resumen de análisis de k -meros con KAT para los ensamblajes de *H. pylori*.

Muestra	0x (solo lecturas)	1x (correcto)	$\geq 2x$ (duplicado/repetido)	Calidad general
SRR18335438	Reducido, pérdida baja	Pico muy fuerte en (~400–750)	Señal mínima en $\geq 2x$	Excelente

Muestra	0x (solo lecturas)	1x (correcto)	≥2x (duplicado/repetido)	Calidad general
SRR18335439	Elevado en bajas multiplicidades	Pico muy fuerte en multiplicidades bajas (~25-50) y amplia representación en multiplicidades altas (~300-700)	Moderado, repetidos parciales	Buena
SRR22388518	Notable presencia en multiplicidades medias	Pico principal dominado por 1x (~150-175)	Señal visible (duplicación leve)	Aceptable
SRR22388519	Similar a SRR22388518, pérdida moderada	Pico definido (~100-140)	Señal visible, repetidos presentes	Aceptable

Al analizar los espectros obtenidos con KAT se pueden realizar las siguientes observaciones:

- **SRR18335438** exhibe el mejor perfil de *k*-meros con un pico muy fuerte en 1x en multiplicidades altas (~400-750), *k*-meros 0x (ausentes en el ensamblaje) reducidos y señal mínima en ≥2x. Este patrón concuerda con la calidad identificada por QUAST (N50: 158.719 pb, solo 50 *contigs*, longitud total apropiada) y BUSCO (99.0% genes completos), ratificando su clasificación como el ensamblaje de mejor calidad general.
- **SRR18335439** presenta el patrón que más difiere del de las otras muestras. El alto contenido de *k*-meros en multiplicidades bajas sugiere regiones del genoma con poca cobertura. Sin embargo, hay amplia representación en multiplicidades medias y altas (~200-600), lo que sugiere que el núcleo del genoma está bien representado, y por tanto, la calidad desde el punto de vista del análisis de *k*-meros es buena. El contenido 0x elevado en bajas multiplicidades indica pérdida considerable de contenido. Estos tres hechos sugieren que la muestra tiene una cobertura de secuenciación desigual, es decir, se tienen cantidades diferentes de lecturas para diferentes partes del ensamblaje y no una distribución homogénea. Esta discrepancia en cubrimiento podría indicar también una situación de contaminación, en donde una parte de ensamblaje corresponde con el genoma del organismo (bien representado), pero existen regiones poco cubiertas que corresponden con contaminantes o errores de secuenciación. La señal moderada en ≥2x sugiere elementos repetidos parcialmente ensamblados. Notablemente, a pesar de tener una calidad buena respecto del análisis de *k*-meros y BUSCO (alta completitud con 99.2% genes completos), es la muestra que presenta métricas deficientes de QUAST (269 *contigs*, longitud excesiva de 5.841.410 pb). Esta contradicción sugiere que aunque los genes esenciales

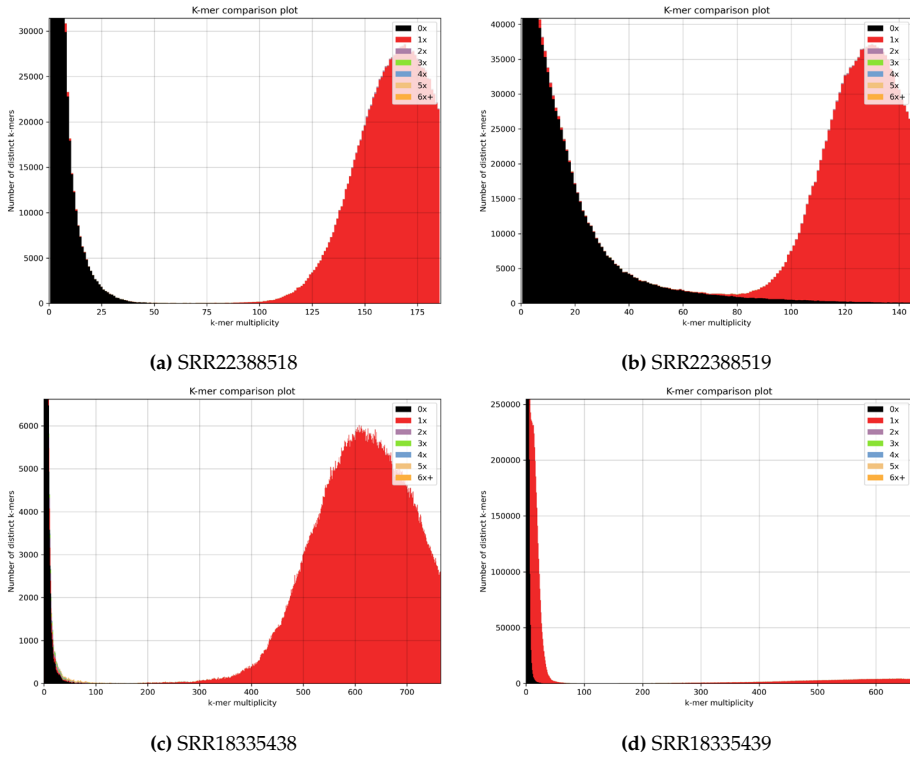


Figura 12. Graficos de espectro de número de copias para las cuatro muestras (lecturas y ensamblajes) analizadas con KAT.

están presentes, el ensamblaje podría contener contaminación significativa y problemas estructurales que comprometen su utilidad práctica.

- **SRR22388518** muestra un pico principal bien definido en multiplicidad ~150-175 con dominancia del color 1x (rojo), indicando que la mayoría de los k -meros fueron ensamblados correctamente una sola vez. Sin embargo, hay presencia notable de k -meros 0x (negro) en multiplicidades medias, sugiriendo pérdida de contenido genuino durante el ensamblaje. La señal visible en $\geq 2x$ indica duplicación leve de algunas regiones. Esto es consistente con las métricas de QUASt que mostraron buena calidad estructural (N50: 233.975 pb, 112 *contigs*) y los resultados de BUSCO que indicaron excelente completitud (99.8% genes completos).
- **SRR22388519** presenta patrones similares a SRR22388518 pero con el pico principal en multiplicidad ~130, también dominado por 1x. La presencia moderada de 0x y señal en $\geq 2x$ es comparable a la muestra anterior. Estos resultados son concordantes con las métricas de QUASt que mostraron buena calidad (N50: 156.397 pb, 81 *contigs*) y BUSCO que confirmó alta completitud (99.8% genes completos).

i Nota 11: Consideraciones técnicas para el análisis de k -meros

- **Selección del valor de k :** el valor de k seleccionado (21 en este caso) representa un equilibrio entre especificidad y sensibilidad. Valores menores de k son más sensibles pero menos específicos, mientras que valores mayores son más específicos pero pueden perder información en regiones de baja complejidad. Para genomas bacterianos, k -meros de 21-31 nucleótidos son generalmente apropiados.
- **Complementariedad con otras métricas:** el análisis de k -meros es complementario a las métricas tradicionales de QUASt y BUSCO, proporcionando una perspectiva adicional sobre la fidelidad del ensamblaje a nivel de secuencia. Mientras QUASt evalúa la estructura del ensamblaje y BUSCO la completitud biológica, el análisis de k -meros evalúa la consistencia entre los datos originales y el resultado del ensamblaje.
- **Limitaciones:** los análisis de k -meros pueden ser sensibles a errores de secuenciación. También, la presencia de secuencias repetitivas puede complicar la interpretación. Por último, el rendimiento depende de la selección apropiada del valor de k para el tipo de genoma analizado.

El análisis integrado de k -meros valida las conclusiones previas y proporciona evidencia adicional de que SRR18335438 es superior, seguido por SRR22388518 y

SRR22388519 con calidad comparable, mientras que SRR18335439, aunque biológicamente completo según BUSCO, presenta posibles problemas estructurales y de contaminación que justifican su exclusión de análisis posteriores.

5.6. Corrección y mejoramiento de ensamblajes

Diversos factores pueden afectar la calidad de un ensamblaje, incluyendo contaminación y problemas técnicos en la biblioteca de ADN para secuenciación, o parámetros de ensamblaje no adecuados. En el caso del ensamblaje [SRR18335439](#), la excesiva longitud y fragmentación sugieren contaminación, la cual podría corresponder a ADN de otros organismos, o incluso, de cepas de *H. pylori* distintas (y más divergentes) de la inicialmente aislada.

Existen varios procedimientos para corregir o mejorar un ensamblaje, entre los cuales se incluye la descontaminación de los datos crudos (es decir, antes del ensamblaje), descontaminación del ensamblaje, ensamblaje posterior de los *contigs* o *scaffolds* obtenidos, y pulido del ensamblaje. Aunque la ejecución de estos procedimientos está fuera del alcance de este documento, se presenta a continuación un breve resumen de cada una de estas operaciones de corrección y mejoramiento y las herramientas de software más comunes para ejecutarlas. Esta información servirá de guía para el lector que quiera profundizar más en el tema de los ensamblajes y su corrección o mejoramiento.

5.6.1. Descontaminación de datos crudos

La descontaminación de datos crudos consiste en identificar y remover secuencias contaminantes antes del proceso de ensamblaje. Este enfoque es preventivo y generalmente más eficiente que corregir el ensamblaje posteriormente. Las herramientas de software principales para descontaminación de datos crudos son:

- Kraken2/Bracken (Lu et al., 2017; Lu et al., 2022; Wood et al., 2019): clasificación taxonómica ultrarrápida para identificar contaminantes.
- Centrifuge (Kim et al., 2016): clasificador metagenómico que requiere menor memoria RAM.
- BBDuk¹⁸ (BBTools¹⁹): filtrado de secuencias basado en *k*-meros de organismos contaminantes.

¹⁸BBDuk Guide - <https://archive.jgi.doe.gov/data-and-tools/software-tools/bbtools/bb-tools-user-guide/bbduk-guide/>

¹⁹BBTools User Guide - <https://archive.jgi.doe.gov/data-and-tools/software-tools/bbtools/bb-tools-user-guide/>

- KneadData²⁰: *pipeline* automatizado que combina control de calidad y descontaminación.
- FastQ Screen (Wingett & Andrews, 2018): mapeo contra múltiples genomas de referencia para detectar contaminación.

5.6.2. Descontaminación del ensamblaje

La descontaminación posterior al ensamblaje es el proceso de identificar y remover *contigs* que no pertenecen al organismo de interés, basándose en criterios como cobertura, composición de nucleótidos o afinidad taxonómica. A continuación se presenta un listado de las principales herramientas para descontaminación del ensamblaje:

- BlobTools (BlobToolKit) (Challis et al., 2020; Laetsch & Blaxter, 2017): visualización y filtrado de *contigs* basado en cobertura y afinidad taxonómica.
- ContEst16S (Lee et al., 2017): detección de contaminación utilizando genes marcadores (ARNr 16S).
- Kraken2: también puede aplicarse directamente a *contigs* ensamblados.
- DIAMOND+MEGAN (Bağcı et al., 2021): análisis taxonómico de *contigs* mediante alineamiento de proteínas.
- CheckM (Parks et al., 2015): evaluación de completitud y contaminación en genomas bacterianos.

5.6.3. Ensamblaje posterior de *contigs* o *scaffolds*

Este proceso, conocido como *scaffolding* o llenado de espacios (*gap filling*), consiste en ordenar y conectar *contigs* para reducir la fragmentación del ensamblaje, utilizando información de lecturas pareadas (*paired-end*), *long reads*, o mapas ópticos. Entre las herramientas comunes para este fin se incluyen:

- SSPACE (Boetzer et al., 2011): *scaffolding* utilizando lecturas pareadas de diferentes tamaños de inserción.
- BESST (Sahlin et al., 2014): algoritmo bayesiano para ordenamiento y orientación de *contigs*.
- MeDuSa (Bosi et al., 2015): *scaffolding* basado en genomas de referencia múltiples.
- Ragout (Kolmogorov et al., 2014): ensamblaje guiado por sintenia de múltiples genomas relacionados.
- TGS-GapCloser (M. Xu et al., 2020): cierre de *gaps* (huecos) utilizando lecturas largas (PacBio/Oxford Nanopore).

²⁰Kneaddata - <https://huttenhower.sph.harvard.edu/kneaddata/>

- LR_Gapcloser (G.-C. Xu et al., 2019): cierre de *gaps* específicamente con lecturas largas de PacBio.

5.6.4. Pulido de ensamblajes

Un proceso de mejoramiento es el pulido (*polishing*) del ensamblaje. El pulido es el proceso de corrección de errores puntuales en la secuencia del ensamblaje, como sustituciones, inserciones o deleciones de nucleótidos individuales. Este proceso mejora la precisión del ensamblaje final mediante el realineamiento de las lecturas originales contra los *contigs* ensamblados. El proceso de pulido es especialmente importante en ensamblajes generados con lecturas largas (p. ej. PacBio u Oxford Nanopore), que aunque proporcionan mejor contigüidad, tienen una tasa de error más elevada que las lecturas cortas de Illumina. El pulido de ensamblajes se puede ejecutar con las siguientes herramientas:

- Pilon (Walker et al., 2014): pulido automático utilizando lecturas Illumina para corregir SNPs, indels y *gaps* pequeños.
- Racon²¹: corrección rápida de consenso para ensamblajes de lecturas largas.
- Medaka²²: herramienta de consenso específica para datos de Oxford Nanopore Technologies.
- Unicycler (Wick et al., 2017): ensamblador híbrido que incluye pasos de pulido automático.
- NextPolish (Hu et al., 2020): pulido de alta eficiencia para ensamblajes de genomas grandes.

5.7. Resumen

El control de calidad de ensamblajes es crucial para asegurar la precisión y utilidad de los genomas ensamblados en análisis posteriores. No solo es necesaria la evaluación de la contigüidad y estructura de los ensamblajes, sino también su completitud biológica y fidelidad a nivel de secuencia.

En este capítulo se estableció un protocolo integral para el control de calidad de ensamblajes genómicos, combinando múltiples enfoques complementarios. El proceso incluyó la descarga del genoma de referencia de *H. pylori* desde RefSeq, la evaluación de métricas estructurales mediante QUAST para calcular índices de contigüidad (N50, NG50, L50, LG50, número de contigs y longitud total), el análisis de completitud biológica a través de BUSCO mediante la identificación de genes

²¹Racon (GitHub): <https://github.com/isovic/racon>

²²Medaka (GitHub): <https://github.com/nanoporetech/medaka>

ortólogos universales de copia única específicos del linaje *Campylobacterales*, y el análisis comparativo de k -meros usando Jellyfish y KAT para evaluar la fidelidad del ensamblaje a nivel de secuencia. El análisis de k -meros proporcionó una evaluación adicional mediante la comparación de espectros entre lecturas originales y ensamblajes, permitiendo identificar regiones mal ensambladas y evaluar la representatividad del ensamblaje. Luego de analizar las métricas obtenidas, se concluyó que tres de los cuatro ensamblajes presentaban calidad adecuada para análisis posteriores, mientras que uno requería corrección debido a su alta fragmentación y longitud excesiva. Finalmente, se proporcionó un resumen de las principales estrategias y herramientas disponibles para corregir o mejorar ensamblajes, incluyendo la descontaminación de datos crudos, descontaminación del ensamblaje, ensamblaje posterior de *contigs* o *scaffolds*, y pulido de ensamblajes.



Capítulo 6.

Anotación funcional

La anotación funcional constituye el paso final del análisis de genomas ensamblados, donde se identifican y caracterizan las características genómicas para facilitar su interpretación biológica. Brevemente, el proceso consiste en identificar características en las secuencias de *contigs/scaffolds* (p. ej. genes), recuperar información útil acerca de ellas, y finalmente, etiquetar o anotar las características con la información recuperada. Entre la información recuperada, la función biológica de los genes es una de las más importantes. La función biológica se encuentra por medio de búsquedas basadas en similitud, en bases de datos públicas especializadas. Haciendo uso de la homología de secuencia, si la secuencia A , de la cual se desconoce su función, es lo suficientemente similar (usualmente $sim \geq 80\%$) a la secuencia B , previamente caracterizada una base de datos, entonces, se puede asociar la función de B a la secuencia A .

Este capítulo presenta la implementación de un flujo de trabajo completo que incluye: la ejecución de *Prokka* para la anotación automatizada de genomas bacterianos, el análisis detallado de los archivos de salida generados y sus aplicaciones específicas, la introducción a conceptos avanzados como el análisis de pangenoma y su relevancia en genómica comparativa, y la visualización de resultados mediante herramientas especializadas (IGV y Proksee) para la validación y exploración de las anotaciones obtenidas. El enfoque metodológico combina la identificación de características genómicas basada en homología de secuencias con técnicas modernas de visualización, proporcionando las bases para análisis posteriores en epidemiología molecular y genómica comparativa.

6.1. Software requerido

Tabla 10. Software requerido para anotación.

Software	Versión	Descripción	Referencia
Prokka ¹	1.14.6	Herramienta de software para la anotación rápida de genomas de bacterias, virus y arqueas.	Seemann (2014)
IGV ²	2.9.0	Visor genómico interactivo multiplataforma para visualización detallada de genomas anotados.	Robinson et al. (2011)
Proksee ³	Web	Plataforma web especializada para visualización circular de genomas procarióticos anotados.	Grant et al. (2023)

6.2. Ejecución de Prokka

Se anotarán los ensamblajes usando la herramienta Prokka, la cual consiste en un *pipeline* que involucra varias herramientas bioinformáticas para identificar las características, hacer las búsquedas en las bases de datos biológicas y, por último, etiquetar las secuencias de las características identificadas.

Para anotar el ensamblaje [SRR22388518](#), se debe ejecutar las siguientes órdenes:

```
# Entrar en el directorio raíz del proyecto
cd $HOME/proyectos/ensamblaje

# Ejecutar la anotación
prokka \
  --cpus 4 \
  --genus "Helicobacter" \
  --species "pylori" \
  --locustag "SRR22388518" \
  --outdir resultados/03_anotacion/SRR22388518 \
  --prefix "SRR22388518" \
  resultados/01_ensamblaje/SRR22388518.fasta
```

Los argumentos usados para la anotación con Prokka son:

¹Prokka, repositorio GitHub: <https://github.com/tseemann/prokka>

²IGV, sitio Web: <https://igv.org/>

³Proksee, sitio Web: <https://proksee.ca>

- **cpus**: el número de *cores* (cpus) a ser usados para agilizar el proceso.
- **genus**: el género del organismo del ensamblaje que queremos anotar.
- **species**: la especie del organismo del ensamblaje que queremos anotar.
- **locustag**: la etiqueta que se agrega a todas las características encontradas y anotadas por Prokka para este ensamblaje. En este caso se usó el *accession number* de la muestra.
- **outdir**: el directorio en donde se guardarán todos los archivos de salida de Prokka.
- **prefix**: el prefijo (nombre) para los diferentes archivos de salida de Prokka. El prefijo siempre será el mismo pero cambia la extensión de cada archivo. En este caso se usó igualmente el *accession number*.
- El archivo de ensamblaje a ser anotado (**resultados/01_ensamblaje/SRR22388518.fasta**).

Al ejecutar el comando se verá una salida parecida a esta:

```
[19:13:29] This is prokka 1.14.6
[19:13:29] Written by Torsten Seemann <torsten.seemann@gmail.com>
[19:13:29] Homepage is https://github.com/tseemann/prokka
[19:13:29] Local time is Fri Mar 24 19:13:29 2023
[19:13:29] You are hector
[19:13:29] Operating system is linux
[19:13:29] You have BioPerl 1.7.8
Argument 1.7.8 isn't numeric in numeric lt (<) at
  ~ /home/hector/mambaforge/envs/ensam/bin/prokka line 259.
[19:13:29] System has 8 cores.
[19:13:29] Will use maximum of 4 cores.
[19:13:29] Annotating as >>> Bacteria <<<
[19:13:29] Creating new output folder: resultados/03_annotacion/SRR22388518
[19:13:29] Running: mkdir -p resultados\03_annotacion\SRR22388518
[19:13:29] Using filename prefix: SRR22388518.XXX
[19:13:29] Setting HMMER_NCPU=1
[19:13:29] Writing log to:
  ~ resultados/03_annotacion/SRR22388518/SRR22388518.log
[19:13:29] Command: /home/hector/mambaforge/envs/ensam/bin/prokka --cpus
  ~ 4 --genus Helicobacter --species pylori --locustag SRR22388518
  ~ --outdir resultados/03_annotacion/SRR22388518 --prefix SRR22388518
  ~ resultados/01_ensamblaje/SRR22388518.fasta

...

[19:14:30] Output files:
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.fsa
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.fna
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.gff
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.sqn
```

```
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.tbl
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.gbk
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.ffn
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.err
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.faa
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.tsv
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.txt
[19:14:30] resultados/03_annotacion/SRR22388518/SRR22388518.log
[19:14:30] Annotation finished successfully.
[19:14:30] Walltime used: 1.02 minutes
[19:14:30] If you use this result please cite the Prokka paper:
[19:14:30] Seemann T (2014) Prokka: rapid prokaryotic genome annotation.
    ~ Bioinformatics. 30(14):2068-9.
[19:14:30] Type 'prokka --citation' for more details.
[19:14:30] Share and enjoy!
```

6.3. Interpretación de la salida de Prokka

La salida de Prokka es bastante detallada y extensa. La primera parte muestra un resumen de las opciones pasadas al programa, incluido el comando ejecutado y la ubicación del archivo *log* (*Writing log to: ...*) con toda la información de salida, muy útil para depuración de errores en caso de que se presenten problemas al ejecutar Prokka. En la parte final se encuentra la lista de archivos de salida, los cuales están ubicados en el directorio *resultados/03_annotacion*. En la Tabla 11 se presenta la descripción de cada archivo de salida.

Tabla 11. Descripción de los archivos de salida generados por Prokka y sus usos principales.

Extensión	Descripción	Usos principales
<i>.gff</i>	El archivo de anotación principal en formato GFF3 que contiene las secuencias y las anotaciones.	Visualización en IGV, Artemis; genómica comparativa; análisis de pangenoma (con Roary)
<i>.gbk</i>	Archivo en formato GenBank derivado del archivo principal <i>.gff</i> .	Visualización en Proksee, y software comercial como Geneious, y SnapGene; conversión de formatos; análisis detallado de secuencias
<i>.fna</i>	Archivo de nucleótidos en formato FASTA de las secuencias de los <i>contigs/scaffolds</i> de entrada.	Búsquedas BLAST, análisis filogenéticos, entrada para otros ensambladores y anotadores
<i>.faa</i>	Archivo de proteínas en formato FASTA de las traducciones de las secuencias codificantes CDS (del inglés <i>coding sequences</i>).	Análisis funcional (InterProScan, eggNOG-mapper), búsquedas de homología de proteínas

Extensión	Descripción	Usos principales
<code>.ffn</code>	Archivo de nucleótidos en formato FASTA de todas los transcritos predichos (CDS, rRNA, tRNA, tmRNA, misc_RNA).	Análisis de expresión génica, estudios de ARN, análisis de contenido GC por gen
<code>.sqn</code>	Un archivo "Seqin" en formato ASN1 para envío a la base de datos GenBank, en caso de ser requerido.	Envío directo a GenBank (NCBI), validación con tbl2asn
<code>.fsa</code>	Archivo de nucleótidos en formato FASTA de las secuencias de los <i>contigs/scaffolds</i> de entrada, usado por el programa tbl2asn para crear el archivo <code>.sqn</code> . Es casi el mismo archivo <code>.fna</code> pero con etiquetas adicionales Seqin en los identificadores de las secuencias.	Preparación para envío a GenBank, procesamiento con tbl2asn
<code>.tbl</code>	Archivo de tabla de características usado por tbl2asn para crear el archivo <code>.sqn</code> .	Envío a GenBank, creación de archivos de secuencia anotados para NCBI
<code>.err</code>	Archivo de reporte de discrepancia para NCBI, en caso de ser requerido, con las anotaciones no aceptables.	Control de calidad pre-envío a GenBank, identificación de problemas de anotación
<code>.log</code>	Archivo con toda la salida generada por Prokka durante la ejecución del comando.	Depuración de errores, análisis de rendimiento, reproducibilidad
<code>.txt</code>	Estadísticas de las características encontradas y anotadas.	Reportes rápidos, control de calidad, resúmenes estadísticos
<code>.tsv</code>	Archivo de texto separado por tabulaciones (TSV, del inglés <i>text separated values</i>) con todas las características: <code>locus_tag</code> , <code>ftype</code> , <code>len_bp</code> , <code>gene</code> , <code>EC_number</code> , <code>COG</code> , <code>product</code> .	Análisis estadísticos, filtrado de genes, <i>scripts</i> personalizados, análisis de anotación funcional

Acto seguido, se procede a anotar el resto de los ensamblajes:

```
prokka \
  --cpus 4 \
  --genus "Helicobacter" \
  --species "pylori" \
  --locustag "SRR22388519" \
  --outdir resultados/03_anotacion/SRR22388519 \
  --prefix "SRR22388519"
```

```
resultados/01_ensamblaje/SRR22388519.fasta

prokka \
  --cpus 4 \
  --genus "Helicobacter" \
  --species "pylori" \
  --locustag "SRR18335438" \
  --outdir resultados/03_anotacion/SRR18335438 \
  --prefix "SRR18335438"\
  resultados/01_ensamblaje/SRR18335438.fasta
```

6.3.1. Archivo de anotación GFF

El elemento más importante generado por Prokka es el archivo GFF (del inglés *General Feature Format*⁴) de anotación principal. Este archivo de texto plano con extensión **.gff** contiene información detallada sobre las características genómicas identificadas (p. ej. genes, ARN ribosomales, ARN de transferencia, etc.). El archivo GFF generado por Prokka utiliza el estándar GFF3⁵, que es un formato de texto estructurado para describir características genómicas. Este formato organiza la información en columnas (campos, Tabla 12) separadas por tabulaciones, donde cada línea representa una característica genómica específica. La comprensión de la estructura del formato GFF es fundamental para la interpretación correcta de los resultados de anotación y para el procesamiento posterior de los datos.

Tabla 12. Descripción de los campos del formato GFF3 utilizado en el archivo de anotación principal de Prokka.

Campo	Nombre	Descripción	Ejemplo
1	seqid	Identificador de la secuencia (cromosoma, <i>contig</i> , <i>scaffold</i>)	SRR22388518_1
2	source	Herramienta o base de datos que generó la anotación	Prokka:1.14.6
3	type	Tipo de característica genómica	gene , CDS , tRNA , rRNA
4	start	Posición de inicio de la característica (base 1)	1
5	end	Posición final de la característica (base 1)	1500
6	score	Puntuación de confianza (opcional, "." si no aplica)	.

⁴Wikipedia - *General feature format*: https://en.wikipedia.org/wiki/General_feature_format

⁵The Sequence Ontology (GitHub) - *Generic Feature Format Version 3 (GFF3)*: <https://github.com/The-Sequence-Ontology/Specifications/blob/master/gff3.md>

Campo	Nombre	Descripción	Ejemplo
7	<code>strand</code>	Hebra del ADN (" <code>+</code> " <i>forward</i> , " <code>-</code> " <i>reverse</i> , " <code>.</code> " desconocida)	<code>+</code>
8	<code>phase</code>	Fase de lectura para CDS (0, 1, 2, o " <code>.</code> " si no aplica)	<code>0</code>
9	<code>attributes</code>	Atributos adicionales en formato clave=valor separados por " <code>;</code> "	<code>ID=gene_001;Name=dnaA;product=chromosomal replication initiator protein DnaA</code>

El archivo GFF es requerido para posteriores análisis de genómica comparativa, como por ejemplo, la identificación del pangenoma⁶ de un grupo de genomas (ver Nota 12). Estos análisis son usados en epidemiología basada en secuenciación, la cual es fundamental en salud pública para la vigilancia y seguimiento de brotes infecciosos.

i Nota 12: Análisis de pangenoma

El análisis de pangenoma consiste en la comparación y clasificación de genes entre múltiples genomas de una misma especie o grupo taxonómico relacionado. Este análisis permite identificar tres grupos principales de genes: el genoma *core* (o núcleo, son los genes presentes en todos los genomas analizados), el genoma accesorio (genes presentes en algunos pero no todos los genomas), y genes únicos (presentes en un solo genoma). Entre los usos del análisis de pangenoma se incluyen, la identificación de genes marcadores (de virulencia, de resistencia a antibióticos, específicos de una especie, serotipo⁷ o cepa), epidemiología molecular (seguimiento de brotes, vigilancia en salud pública, tipificación molecular), evolución y diversidad genómica (transferencia horizontal, presión selectiva, conservación de genes), y aplicaciones biotecnológicas (diagnóstico molecular, desarrollo de vacunas, ingeniería metabólica).

Entre la herramientas más importantes para realizar análisis de pangenoma se pueden mencionar, Roary (Page et al., 2015), Panaroo (Tonkin-Hill et al., 2020), PPanGGOLiN (Gautreau et al., 2020), panX (Ding et al., 2018), y RIBAP (Lamkiewicz et al., 2024).

⁶Wikipedia - Pangenoma: <https://es.wikipedia.org/wiki/Pangenoma>.

⁷Wikipedia - Serotipo: <https://es.wikipedia.org/wiki/Serotipo>.

6.4. Visualización de los resultados de anotación

La visualización de los resultados de anotación es un proceso que permite examinar, interpretar y validar las características genómicas identificadas durante el proceso de anotación. Su objetivo principal es proporcionar una representación gráfica, en lo posible interactiva, de las secuencias de ADN y sus elementos funcionales anotados. La visualización permite realizar, entre otras tareas, la validación manual de las anotaciones generadas por las herramientas como Prokka, la identificación de patrones, la comparación visual entre diferentes genomas, y la detección de errores en el ensamblaje o anotación. En general, la visualización facilita la interpretación biológica y la toma de decisiones en análisis genómicos.

6.4.1. Herramientas de visualización

Aunque existen múltiples programas para visualización de genomas anotados, cabe resaltar las herramientas IGV (Robinson et al. (2011); del inglés *Integrative Genomics Viewer*) y Proksee (Grant et al. (2023)), por ser herramientas versátiles, populares y gratuitas.

6.4.1.1. IGV

IGV es uno de los visualizadores genómicos más utilizados y es útil para la visualización detallada de genomas bacterianos anotados. Esta herramienta multiplataforma (Windows, Mac, Linux) cuenta con una interfaz intuitiva, soporta formatos GFF3, GenBank, y FASTA, permite cargar múltiples pistas (*tracks*) de datos simultáneamente, y ofrece generación de imágenes de alta calidad para publicaciones.

Para visualizar el genoma anotado de [SRR22388518](#) en IGV, se debe convertir el archivo `.gff` a formato `.gtf` (del inglés *Gene Transfer Format*⁸), ya que IGV no soporta directamente archivos GFF3. Esto se puede hacer usando la herramienta `gffread`. Si no se tiene instalado `gffread`, se puede instalar usando `mamba`:

```
# Entrar en el directorio raíz del proyecto
cd proyectors/ensamblaje

# Activar el entorno ensam
mamba activate ensam
```

⁸Wikipedia - *Gene transfer format*: https://en.wikipedia.org/wiki/Gene_transfer_format

```
# Instalar gffread (requisito para IGV)
mamba install -c bioconda gffread=0.12.7
```

Acto seguido, se convierte el archivo `.gff` a `.gtf` y se abre IGV:

```
# Convertir GFF a GTF
gffread \
  resultados/03_annotacion/SRR22388518/SRR22388518.gff \
  -T \
  -o resultados/03_annotacion/SRR22388518/SRR22388518.gtf

# Abrir IGV
igv
```

En IGV, se debe cargar el genoma y la anotación:

1. Cargar el genoma: vaya al Menú > *Genome* > *Load Genome from File...* > seleccionar el archivo `resultados/01_ensamblaje/SRR22388518.fasta`
2. Cargar la anotación: vaya al Menú > *File* > *Load from File...* > seleccionar el archivo `resultados/03_annotacion/SRR22388518/SRR22388518.gtf`

A continuación, se puede explorar el genoma anotado de `SRR22388518` en IGV, como se muestra en la Figura 13.

6.4.1.2. Proksee

Proksee es una herramienta Web moderna específicamente diseñada para la visualización de genomas procarióticos. Desarrollada por el mismo grupo que creó Prokka, la herramienta está optimizada para mostrar características típicas de genomas bacterianos. Proksee no requiere instalación, y además, permite la visualización circular del genoma completo, genera representación automática de operones y *clusters* génicos, cuenta con herramientas de análisis comparativo integradas, y ofrece generación de mapas genómicos para publicaciones.

Para visualizar el genoma anotado de `SRR22388518` en Proksee, se debe cargar el archivo `.gbk` generado por Prokka. Sin embargo, debido a un defecto en la generación del archivo `.gbk` por Prokka, es necesario hacer una pequeña corrección en el archivo antes de cargarlo en Proksee. Esto se puede hacer usando un sencillo *script* de Bash. Para crear el *script*, ejecute el siguiente comando:

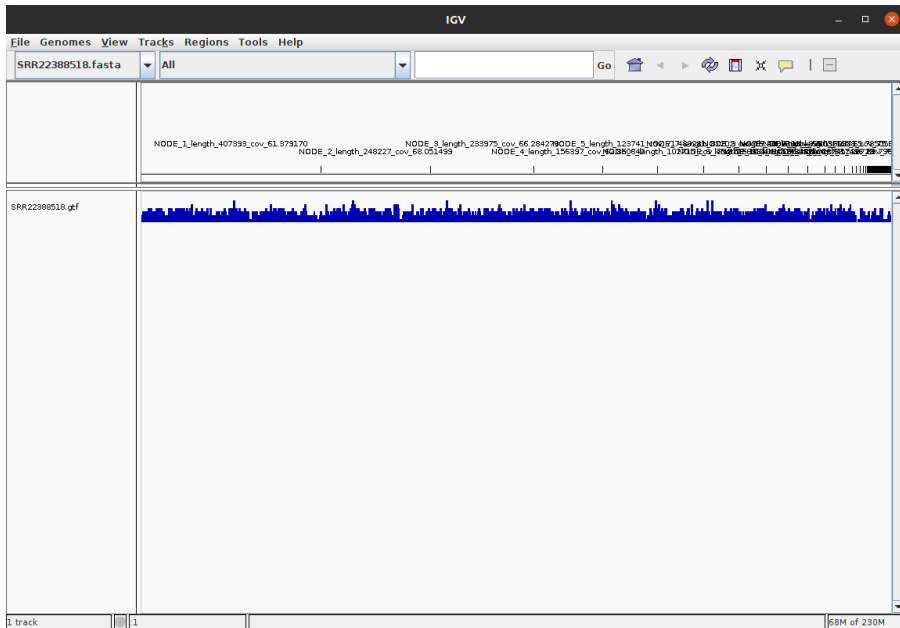


Figura 13. Visualización del genoma anotado de *H. pylori*, muestra SRR22388518, en IGV

```
# Entrar en el directorio raíz del proyecto
cd $HOME/proyectos/ensamblaje

# Crear el script corregir_gbk.sh
echo "#!/usr/bin/env bash
# Script para corregir archivos .gbk generados por Prokka

gbk_file=$1

awk '/^LOCUS/ {
    split($2, arr, "\\_\\");
    pat=arr[4]\\_ bp\\;
    gsub(pat, "\\_ \\\"pat\\");
}
{print}' $gbk_file" > scripts/corregir_gbk.sh

# Dar permisos de ejecución al script
chmod +x scripts/corregir_gbk.sh
```

Ahora, se puede corregir el archivo `.gbk` de la muestra `SRR22388518` ejecutando el siguiente comando:

```
./scripts/corregir_gbk.sh \
  resultados/03_anotacion/SRR22388518/SRR22388518.gbk \
  > resultados/03_anotacion/SRR22388518/SRR22388518.fixed.gbk
```

El archivo corregido **SRR22388518.fixed.gbk** ya está listo para ser cargado en Proksee.

Para cargar el archivo se debe ir a la página web de Proksee: <https://proksee.ca>, y seguir los pasos para cargar el archivo **.gbk**:

1. Hacer clic en el Menú > *New Project*
2. En *Create a New Map*, hacer clic en “Browse” y seleccionar el archivo [resultados/03_anotacion/SRR22388518/SRR22388518.fixed.gbk](#). El archivo se cargará automáticamente.
3. Hacer clic en el botón “Create Map” para generar el mapa genómico.

A continuación, se puede explorar el genoma anotado de [SRR22388518](#) en Proksee, como se muestra en la Figura 14.

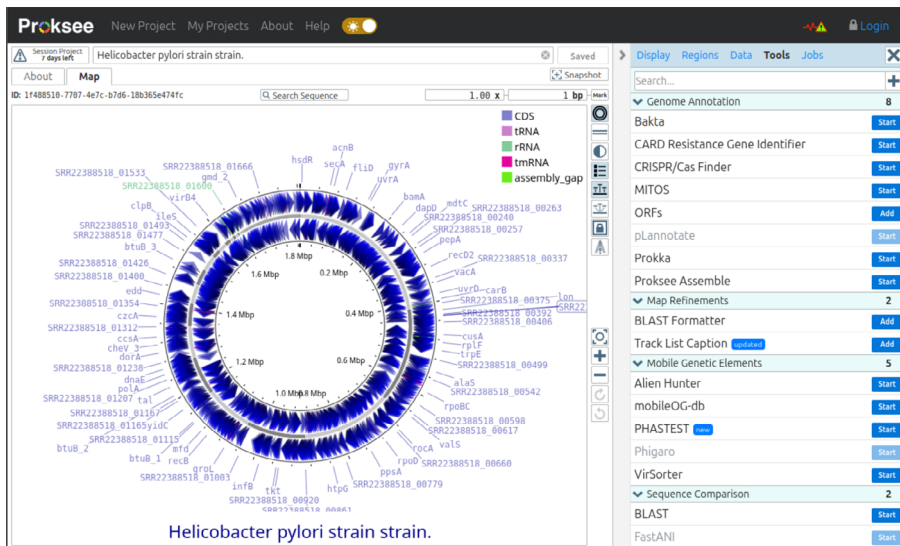


Figura 14. Visualización del genoma anotado de *H. pylori*, muestra SRR22388518, en Proksee.

 Consejo

Si prefiere las interfaces gráficas o tiene limitaciones de recursos computacionales, puede utilizar Proksee para ensamblar muestras (via SPAdes), evaluar la calidad ensamblajes, realizar anotación (via Prokka), y ejecutar otros análisis como comparación de secuencias (con BLAST y FastANI), y composición de secuencias (análisis de contenido GC)

6.4.2. Comparación de herramientas

Tabla 13. Comparación entre IGV y Proksee para visualización de resultados de anotación.

Característica	IGV	Proksee
Tipo de instalación	Programa local	Herramienta web
Especialización	Genomas eucariotas y procariotas	Específico para procariotas
Visualización	Lineal, detallada	Circular, panorámica
Curva de aprendizaje	Moderada	Baja
Capacidades de análisis	Extensas (con <i>plugins</i>)	Básicas pero específicas
Exportación	Múltiples formatos	PNG, SVG

6.5. Resumen

La anotación funcional es un paso crucial en el análisis de genomas, ya que permite identificar y caracterizar las características genómicas, facilitando la interpretación biológica de los datos. En este capítulo se presentó un flujo de trabajo para la anotación automatizada de los ensamblajes y su visualización. Se utilizó la herramienta Prokka para anotar los ensamblajes bacterianos y se describieron los archivos de salida generados y sus usos principales en análisis genómicos. Se presentó el análisis de pangenoma como una aplicación importante de la anotación funcional en genómica comparativa. Finalmente, se utilizaron dos herramientas populares para visualización de los resultados de la anotación funcional: IGV, recomendada para análisis detallados, y Proksee, recomendada para visualizaciones panorámicas y generación de figuras para publicaciones.



Capítulo 7.

Plataformas Web para ensamblaje y anotación

Los análisis de ensamblaje y anotación de genomas bacterianos presentados en los capítulos anteriores requieren la instalación y configuración de múltiples herramientas bioinformáticas, así como conocimientos técnicos en el uso de la línea de comandos. Sin embargo, existen alternativas basadas en plataformas web que democratizan el acceso a estos análisis, permitiendo a investigadores sin experiencia en bioinformática realizar estudios genómicos de alta calidad.

Este capítulo presenta las principales plataformas web disponibles para el ensamblaje y anotación de genomas bacterianos, sus características, ventajas, limitaciones y los contextos en los que resultan más apropiadas.

7.1. Usuarios objetivo y beneficios

7.1.1. Perfiles de usuarios que se benefician

Las plataformas web de análisis genómico están diseñadas para tres grupos principales. Primero, investigadores biomédicos sin formación bioinformática, como microbiólogos clínicos, epidemiólogos y estudiantes de posgrado, que pueden acceder a herramientas sofisticadas sin dominar la línea de comandos. Segundo, laboratorios con recursos computacionales limitados, incluyendo instituciones sin acceso a clusters de alto rendimiento o con presupuestos restringidos, democratizando el acceso a análisis de calidad. Tercero, profesionales que requieren análisis rápidos y estandarizados, como laboratorios de salud pública, consultores en microbiología e investigadores que necesitan resultados reproducibles.

7.1.2. Ventajas principales

Las plataformas web ofrecen ventajas significativas en tres áreas clave. En accesibilidad, proporcionan interfaces intuitivas sin requerir programación, flujos de trabajo optimizados y soporte técnico especializado. En infraestructura computacional, democratizan el acceso a recursos de alto rendimiento gratuitos, implementan escalabilidad automática y garantizan mantenimiento automatizado con almacenamiento seguro. En reproducibilidad, aseguran flujos de trabajo versionados y documentados, parámetros basados en mejores prácticas, trazabilidad completa y facilitan compartir resultados estandarizados.

7.1.3. Limitaciones y consideraciones

Las plataformas web presentan limitaciones importantes en tres aspectos. Restricciones técnicas incluyen menor flexibilidad para personalización, dependencia de versiones disponibles, limitaciones en tamaño de archivos y tiempos de espera durante alta demanda. Consideraciones de privacidad involucran la necesidad de cargar datos sensibles externamente, políticas variables de retención y restricciones para información confidencial. Limitaciones científicas abarcan menor control sobre algoritmos, dificultad para integrar herramientas personalizadas y posibles retrasos en adopción de nuevos métodos.

7.2. Contextos de uso apropiados

- **Análisis de rutina y diagnóstico:** las plataformas web son ideales para identificación taxonómica de aislamientos bacterianos, detección de resistencia a antibióticos crucial en la era antimicrobiana, análisis de virulencia para evaluar potencial patogénico, y comparación con genomas de referencia mediante análisis filogenéticos automatizados.
- **Investigación exploratoria:** estas plataformas facilitan estudios piloto antes de análisis intensivos, validación de resultados obtenidos por otros métodos, enseñanza y entrenamiento en entornos controlados, y análisis comparativos rápidos que revelan patrones evolutivos o funcionales.
- **Colaboración y divulgación:** son valiosas para proyectos colaborativos multi-institucionales que requieren estandarización, compartir análisis con colegas no especializados, generar resultados preliminares para propuestas de financiamiento, y crear visualizaciones de alta calidad para publicaciones científicas.

7.3. Plataformas recomendadas

A continuación se presentan cinco plataformas web seleccionadas por su robustez, facilidad de uso y especialización en análisis genómicos bacterianos. Cada plataforma se describe en términos de sus características principales, capacidades técnicas, ventajas específicas y limitaciones, proporcionando la información necesaria para seleccionar la herramienta más apropiada según las necesidades específicas del proyecto. Al final de esta sección, la Tabla 14 resume las características clave de cada plataforma para facilitar la comparación y selección.

7.3.1. BV-BRC

BV-BRC¹ (del inglés *Bacterial and Viral Bioinformatics Resource Center*) es una plataforma integral financiada por NIAID/NIH que proporciona herramientas especializadas para el análisis de genomas bacterianos y virales. Representa la evolución y consolidación de varios recursos previos como PATRIC, IRD y ViPR. La plataforma integra una base de datos con más de 2 millones de genomas bacterianos, herramientas específicas para patógenos de importancia clínica, y servicios especializados de anotación funcional con énfasis en virulencia.

La plataforma ofrece algoritmos optimizados para ensamblaje de genomas bacterianos, pipelines especializados de anotación, identificación automática de ortólogos y detección de genes de resistencia antimicrobiana (AMR). Sus principales ventajas incluyen optimización para patógenos de importancia médica, integración con bases de datos clínicas y epidemiológicas, e interfaz diseñada específicamente para microbiólogos clínicos. Sin embargo, presenta limitaciones como enfoque principalmente en patógenos (menos útil para bacterias ambientales), requerimientos de registro con posibles restricciones de uso, y menor flexibilidad para análisis personalizados. El contexto de uso de esta herramienta incluye análisis clínicos, vigilancia epidemiológica e investigación en patógenos bacterianos.

7.3.2. KBase

KBase² (*The Department of Energy Systems Biology Knowledgebase*) es una plataforma de análisis de sistemas biológicos desarrollada por el Departamento de Energía de EE.UU., enfocada en microbiología ambiental, biotecnología y biología de sistemas. Se especializa en biología de sistemas y redes metabólicas, integrando datos genómicos, transcriptómicos y metabólicos con herramientas avanzadas para análisis de comunidades microbianas y modelado metabólico con predicción de fenotipos.

¹BV-BRC - Sitio Web: <https://www.bv-brc.org>

²KBase - Sitio Web: <https://www.kbase.us>

La plataforma proporciona múltiples algoritmos de ensamblaje (SPAdes, MaSuRCA, etc.), pipeline RAST integrado con análisis metabólico, construcción de pangenomas y reconstrucción automática de redes metabólicas. Sus principales ventajas incluyen excelencia para análisis de biología de sistemas, herramientas únicas de modelado metabólico, enfoque en reproducibilidad y documentación, e integración de múltiples tipos de datos ómicos. Las limitaciones abarcan curva de aprendizaje más pronunciada, menor enfoque en aspectos clínicos, y requerimiento de comprensión de conceptos de biología de sistemas. Su contexto de uso incluye investigación en microbiología ambiental, biotecnología y estudios de metabolismo bacteriano.

7.3.3. Galaxy

Galaxy³ es una plataforma de código abierto ampliamente adoptada que proporciona acceso web a herramientas bioinformáticas. Es conocida por su flexibilidad, extensibilidad y fuerte énfasis en la reproducibilidad. Ofrece más de 8,000 herramientas bioinformáticas disponibles, flujos de trabajo completamente personalizables, múltiples instancias especializadas (usegalaxy.org, usegalaxy.eu, etc.), y una comunidad activa de desarrolladores con integración a repositorios de herramientas (Tool Shed).

La plataforma incluye múltiples herramientas de ensamblaje (SPAdes, Unicycler, Flye, Canu), diversas opciones de anotación (Prokka, DFAST, PGAP, RASTtk), control de calidad integral (FastQC, MultiQC, QUAST), y análisis downstream especializados (antiSMASH, RGI, ABRicate). Sus principales ventajas son máxima flexibilidad y personalización, comunidad muy activa con soporte extenso, código abierto instalable localmente, y excelencia para enseñanza. Las limitaciones incluyen posible complejidad abrumadora para novatos, requerimiento de más conocimiento técnico para uso avanzado, y variabilidad entre diferentes instancias. Su contexto de uso incluye investigación general, enseñanza, análisis personalizados y prototipado de flujos de trabajo.

7.3.4. NCBI Genome Workbench

NCBI Genome Workbench⁴ es una plataforma integral del NCBI que combina herramientas de análisis con acceso directo a las bases de datos más completas de secuencias genómicas. Se caracteriza por la integración directa con bases de datos NCBI, herramientas oficiales de anotación (PGAP), servicios de comparación genómica, preparación de datos para envío a GenBank, y análisis filogenético integrado.

³Galaxy - Sitio Web: <https://usegalaxy.org>

⁴NCBI Genome Workbench - Sitio Web: <https://www.ncbi.nlm.nih.gov/genome/gdv/>

La plataforma ofrece el pipeline oficial PGAP (del inglés *Prokaryotic Genome Annotation Pipeline*), herramientas oficiales de genómica comparativa, navegadores genómicos integrados, y preparación automática para GenBank. Sus principales ventajas incluyen estándares oficiales de NCBI para anotación, integración perfecta con bases de datos públicas, preparación automática para publicación, y herramientas especializadas de análisis filogenético. Las limitaciones comprenden menor flexibilidad en parámetros de análisis, enfoque más conservador en métodos utilizados, e interfaz menos intuitiva para usuarios novatos. Dentro del contexto de uso de esta herramienta se incluye la preparación de genomas para publicación, análisis oficiales y estudios filogenéticos.

7.3.5. RAST

RAST⁵ (del inglés *Rapid Annotation using Subsystem Technology*) es una plataforma pionera en anotación automática de genomas procarióticos, desarrollada por el *Argonne National Laboratory*. Se enfoca en la anotación funcional basada en subsistemas metabólicos, ofreciendo anotación automática rápida y precisa, comparación automática con genomas relacionados, herramientas de visualización integradas, y acceso gratuito para la comunidad académica.

La plataforma proporciona pipeline automatizado basado en subsistemas, análisis automático con genomas relacionados, mapas metabólicos y genómicos interactivos, y clasificación por subsistemas metabólicos. Sus principales ventajas incluyen interfaz muy intuitiva para principiantes, análisis metabólicos especializados, procesamiento rápido y eficiente, e idoneidad para propósitos educativos. Las limitaciones abarcan funcionalidad limitada comparada con otras plataformas, menor flexibilidad en parámetros de análisis, enfoque específico en anotación funcional, y opciones limitadas de análisis comparativo. Su contexto de uso incluye anotación funcional rápida, análisis metabólicos y proyectos educativos.

7.4. Comparativa de plataformas

Tabla 14. Comparación de plataformas web para análisis genómicos.

Plataforma	Enfoque principal	Facilidad de uso	Flexibilidad	Ideal para
BV-BRC	Patógenos bacterianos	Alta	Media	Análisis clínicos
KBase	Biología de sistemas	Media	Alta	Investigación ambiental

⁵RAST - Sitio Web: <https://rast.nmpdr.org/>

Plataforma	Enfoque principal	Facilidad de uso	Flexibilidad	Ideal para
Galaxy	Análisis general	Media	Muy alta	Enseñanza y investigación
NCBI	Estándares oficiales	Alta	Baja	Publicación
RAST	Anotación funcional	Muy alta	Baja	Proyectos educativos



Capítulo 8.

Conclusiones

Este libro ha presentado un flujo de trabajo completo para el ensamblaje y anotación de genomas bacterianos individuales utilizando datos de secuenciación Illumina. A través del análisis de cuatro genomas de *Helicobacter pylori*, se ha demostrado la aplicación práctica de herramientas bioinformáticas estándar en un contexto real de genómica bacteriana.

El enfoque metodológico adoptado priorizó la claridad didáctica y la reproducibilidad, presentando cada paso del flujo de trabajo de manera detallada y sistemática. Desde el control de calidad inicial de los datos crudos hasta la visualización básica final de genomas anotados, cada etapa fue diseñada para proporcionar a los lectores las habilidades fundamentales necesarias para realizar análisis genómicos independientes.

Los resultados obtenidos ilustran la efectividad de las herramientas seleccionadas para generar ensamblajes de calidad apropiada y anotaciones funcionales completas. Las métricas de calidad evaluadas mediante un enfoque integral que combina QUAST, BUSCO y análisis de k -meros, confirman que el flujo de trabajo propuesto produce resultados acordes con los estándares contemporáneos de bioinformática para genomas bacterianos.

A continuación se presentan las conclusiones generales, limitaciones del enfoque actual, y posibles direcciones para trabajos futuros que podrían ampliar y mejorar la metodología presentada.

8.1. Conclusiones generales

La metodología presentada en este libro constituye una base sólida y comprensiva para investigadores que inician en el análisis genómico bacteriano mediante herramientas de línea de comandos en ambientes Linux. El flujo de trabajo integra herramientas consolidadas y ampliamente validadas en la comunidad científica, garantizando la compatibilidad con protocolos estándar y la comparabilidad de resultados entre diferentes estudios y laboratorios.

El enfoque metodológico demostró ser efectivo para producir genomas bacterianos de calidad apropiada para análisis posteriores, como se evidencia en los resultados obtenidos con *H. pylori*. La implementación de un protocolo de control de calidad multicapa (QUAST, BUSCO, análisis de *k*-meros) proporciona evaluación integral que abarca aspectos estructurales, biológicos y de consistencia de secuencia, estableciendo un estándar robusto para la validación de ensamblajes.

La selección de *H. pylori* como organismo modelo resulta estratégicamente apropiada para propósitos educativos debido a múltiples factores convergentes: relevancia clínica significativa, tamaño genómico manejable (~1.6 Mbp), estructura relativamente simple sin elementos repetitivos complejos, y disponibilidad de genomas de referencia de alta calidad para validación comparativa. Los cuatro genomas analizados proporcionan diversidad adecuada para demostrar la robustez del flujo de trabajo mientras mantienen la complejidad computacional accesible para usuarios con recursos limitados.

El énfasis en reproducibilidad mediante *scripts* automatizados, ambientes de software controlados con Conda/Mamba, y documentación detallada facilita significativamente la adopción y implementación de la metodología en contextos institucionales diversos. La disponibilidad pública de código fuente, configuraciones de ambiente, y datos de prueba permite que investigadores con diversos niveles de experiencia técnica puedan replicar y adaptar el flujo de trabajo según sus necesidades específicas. Adicionalmente, la inclusión de plataformas web alternativas democratiza el acceso a análisis genómicos para investigadores sin recursos computacionales especializados.

8.2. Limitaciones

El alcance de este libro presenta varias limitaciones importantes que deben considerarse al interpretar y aplicar la metodología presentada. En conjunto, estas limitaciones reflejan las decisiones metodológicas adoptadas para mantener la accesibilidad y claridad didáctica del flujo de trabajo propuesto:

- **Tamaño del conjunto de datos:** el conjunto de datos utilizado, aunque apropiado para propósitos didácticos, representa una escala limitada comparada con estudios genómicos contemporáneos que típicamente analizan cientos o miles de genomas simultáneamente.
- **Flujo de trabajo básico:** la simplificación del flujo de trabajo para el ensamblaje de genomas bacterianos, aunque pedagógicamente valiosa, omite pasos avanzados que pueden ser críticos para genomas complejos o estudios de alta precisión.

- **Uso de herramientas individuales:** la dependencia de una sola herramienta para cada paso del flujo de trabajo (p. ej. SPAdes para ensamblaje, Prokka para anotación funcional) puede introducir sesgos específicos del algoritmo y limitar la detección de artefactos. Los enfoques comparativos que utilizan múltiples herramientas para cada fase del flujo de trabajo pueden mejorar la calidad final mediante la obtención de resultados consensuados.
- **Exclusión de tareas avanzadas de corrección:** la omisión de las tareas para mejoramiento y corrección (Sección 5.6) en el flujo de trabajo, puede resultar en ensamblajes subóptimos para ciertos tipos de muestras o aplicaciones específicas. Sin embargo, dado el objetivo didáctico básico de esta obra, se priorizó la comprensión de los conceptos fundamentales sobre la implementación de técnicas avanzadas que podrían generar complejidad adicional para usuarios principiantes.
- **Visualización de genomas anotados básica:** La visualización genómica es fundamental para validación manual y interpretación biológica, pero recibió tratamiento limitado en esta obra debido a restricciones de espacio y enfoque metodológico.
- **Flujo de trabajo específico para datos de una sola tecnología:** el flujo de trabajo presentado está específicamente optimizado para datos de secuenciación Illumina de genomas bacterianos individuales. Esta especialización excluye ensamblaje metagenómico, genomas eucariotas complejos, y tecnologías de secuenciación de tercera generación que requieren consideraciones metodológicas distintas.

8.3. Trabajos futuros

Las direcciones identificadas para trabajos futuros representan extensiones naturales del flujo de trabajo presentado, cada una abordando limitaciones específicas y expandiendo las capacidades metodológicas hacia contextos genómicos más complejos y avanzados:

- **Ensamblaje de genomas con datos de lecturas largas (Oxford Nanopore y PacBio):** Las tecnologías de secuenciación de tercera generación ofrecen lecturas de decenas de kilobases que pueden resolver regiones repetitivas complejas y mejorar significativamente la contigüidad de ensamblajes. La integración de herramientas especializadas como Flye, Canu, o Unicycler permitiría explorar estas capacidades avanzadas.
- **Ensamblaje híbrido (lecturas cortas + largas):** la combinación de precisión de Illumina con la longitud de Oxford Nanopore o PacBio representa el estado

del arte en ensamblaje genómico. Metodologías híbridas pueden producir genomas de alta calidad con precisión nucleotídica excepcional.

- **Expansión a dominios biológicos diversos:** el flujo de trabajo actual, optimizado para genomas bacterianos individuales, podría adaptarse para abordar la diversidad genómica completa encontrada en diferentes dominios de la vida y ecosistemas complejos. Esta expansión incluiría tres direcciones principales: (1) análisis metagenómico para estudiar comunidades microbianas permitiendo investigaciones en microbiología ambiental y estudios de microbioma; (2) ensamblaje de genomas eucariotas como hongos y parásitos; y (3) reconstrucción de genomas virales. Cada dominio requeriría consideraciones metodológicas específicas, desde el manejo de cobertura desigual en metagenomas hasta la resolución de regiones repetitivas en eucariotas y la gestión de variabilidad genética en virus.
- **Comparación de ensambladores y métodos de control de calidad:** el desarrollo de protocolos de *benchmarking* permitiría evaluar objetivamente el rendimiento de diferentes herramientas bajo condiciones controladas, facilitando la selección óptima de software para aplicaciones específicas.

Estas extensiones no solo ampliarían el alcance técnico de la metodología, sino que también proporcionarían oportunidades para explorar preguntas biológicas más complejas y contribuir al avance del conocimiento en genómica microbiana y campos relacionados. La implementación de estos trabajos futuros requeriría consideración cuidadosa de recursos computacionales, experticia técnica, y objetivos de investigación específicos.



Referencias

- Alneberg, J., Bjarnason, B. S., Bruijn, I. de, Schirmer, M., Quick, J., Ijaz, U. Z., Lahti, L., Loman, N. J., Andersson, A. F., & Quince, C. (2014). Binning Metagenomic Contigs by Coverage and Composition. *Nature Methods*, 11(11), 1144-1146. <https://doi.org/10.1038/nmeth.3103>
- Andrews, S. (2010). *FastQC: A Quality Control Tool for High Throughput Sequence Data* [Software]. <https://www.bioinformatics.babraham.ac.uk/projects/fastqc/>
- Bağcı, C., Patz, S., & Huson, D. H. (2021). DIAMOND+MEGAN: Fast and Easy Taxonomic and Functional Analysis of Short and Long Microbiome Sequences. *Current Protocols*, 1(3), e59. <https://doi.org/10.1002/cpz1.59>
- Bankevich, A., Nurk, S., Antipov, D., Gurevich, A. A., Dvorkin, M., Kulikov, A. S., Lesin, V. M., Nikolenko, S. I., Pham, S., Prjibelski, A. D., Pyshkin, A. V., Sirotkin, A. V., Vyahhi, N., Tesler, G., Alekseyev, M. A., & Pevzner, P. A. (2012). SPAdes: A New Genome Assembly Algorithm and Its Applications to Single-Cell Sequencing. *Journal of Computational Biology*, 19(5), 455-477. <https://doi.org/10.1089/cmb.2012.0021>
- Bashir, A., Klammer, A. A., Robins, W. P., Chin, C.-S., Webster, D., Paxinos, E., Hsu, D., Ashby, M., Wang, S., Peluso, P., Sebra, R., Sorenson, J., Bullard, J., Yen, J., Valdovino, M., Mollova, E., Luong, K., Lin, S., LaMay, B., ... Schadt, E. E. (2012). A Hybrid Approach for the Automated Finishing of Bacterial Genomes. *Nature Biotechnology*, 30(7), 701-707. <https://doi.org/10.1038/nbt.2288>
- Boetzer, M., Henkel, C. V., Jansen, H. J., Butler, D., & Pirovano, W. (2011). Scaffolding Pre-Assembled Contigs Using SSPACE. *Bioinformatics*, 27(4), 578-579. <https://doi.org/10.1093/bioinformatics/btq683>
- Bosi, E., Donati, B., Galardini, M., Brunetti, S., Sagot, M.-F., Li'o, P., Crescenzi, P., Fani, R., & Fondi, M. (2015). MeDuSa: A Multi-Draft Based Scaffold. *Bioinformatics*, 31(15), 2443-2451. <https://doi.org/10.1093/bioinformatics/btv171>
- Challis, R., Richards, E., Rajan, J., Cochrane, G., & Blaxter, M. (2020). BlobToolKit – Interactive Quality Assessment of Genome Assemblies. *G3 Genes | Genomes | Genetics*, 10(4), 1361-1374. <https://doi.org/10.1534/g3.119.400908>
- Chen, S., Zhou, Y., Chen, Y., & Gu, J. (2018). fastp: an ultra-fast all-in-one FASTQ preprocessor. *Bioinformatics*, 34(17), i884-i890. <https://doi.org/10.1093/bioinformatics/bty560>
- Cock, P. J. A., Fields, C. J., Goto, N., Heuer, M. L., & Rice, P. M. (2010). The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina

- FASTQ variants. *Nucleic Acids Research*, 38(6), 1767-1771. <https://doi.org/10.1093/nar/gkp1137>
- Compeau, P. E. C., Pevzner, P. A., & Tesler, G. (2011). Why are de Bruijn graphs useful for genome assembly? *Nature biotechnology*, 29(11), 987-991. <https://doi.org/10.1038/nbt.2023>
- Ding, W., Baumdicker, F., & Neher, R. A. (2018). panX: Pan-Genome Analysis and Exploration. *Nucleic Acids Research*, 46(1), e5. <https://doi.org/10.1093/nar/gkx977>
- Earl, D., Bradnam, K., John, J. S., Darling, A., Lin, D., Fass, J., Yu, H. O. K., Buffalo, V., Zerbino, D. R., Diekhans, M., Nguyen, N., Ariyaratne, P. N., Sung, W.-K., Ning, Z., Haimel, M., Simpson, J. T., Fonseca, N. A., Birol, I., Docking, T. R., ... Paten, B. (2011). Assemblathon 1: A competitive assessment of de novo short read assembly methods. *Genome Research*, 21(12), 2224-2241. <https://doi.org/10.1101/gr.126599.111>
- Espitia-Navarro, H. F., Chande, A. T., Nagar, S. D., Smith, H., Jordan, I. K., & Rishishwar, L. (2020). STing: accurate and ultrafast genomic profiling with exact sequence matches. *Nucleic Acids Research*, 48(14), 7681-7689. <https://doi.org/10.1093/nar/gkaa566>
- Ewels, P., Magnusson, M., Lundin, S., & Käller, M. (2016). MultiQC: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics*, 32(19), 3047-3048. <https://doi.org/10.1093/bioinformatics/btw354>
- Gautreau, G., Bazin, A., Gachet, M., Planel, R., Burlot, L., Dubois, M., Perrin, A., M'edigue, C., Calteau, A., Cruveiller, S., Matias, C., Ambroise, C., Rocha, E. P. C., & Vallenet, D. (2020). PPanGGOLiN: Depicting Microbial Diversity via a Partitioned Pangenome Graph. *PLOS Computational Biology*, 16(3), e1007732. <https://doi.org/10.1371/journal.pcbi.1007732>
- Grant, J. R., Enns, E., Marinier, E., Mandal, A., Herman, E. K., Chen, C., Graham, M., Domselaar, G. V., & Stothard, P. (2023). Proksee: In-Depth Characterization and Visualization of Bacterial Genomes. *Nucleic Acids Research*, 51(W1), W484-W492. <https://doi.org/10.1093/nar/gkad326>
- Gurevich, A., Saveliev, V., Vyahhi, N., & Tesler, G. (2013). QUAST: quality assessment tool for genome assemblies. *Bioinformatics*, 29(8), 1072-1075. <https://doi.org/10.1093/bioinformatics/btt086>
- Haft, D. H., DiCuccio, M., Badretadin, A., Brover, V., Chetvernin, V., O'Neill, K., Li, W., Chitsaz, F., Derbyshire, M. K., Gonzales, N. R., Gwadz, M., Lu, F., Marchler, G. H., Song, J. S., Thanki, N., Yamashita, R. A., Zheng, C., Thibaud-Nissen, F., Geer, L. Y., ... Pruitt, K. D. (2018). RefSeq: An Update on Prokaryotic Genome Annotation and Curation. *Nucleic Acids Research*, 46(D1), D851-D860. <https://doi.org/10.1093/nar/gkx1068>
- Hu, J., Fan, J., Sun, Z., & Liu, S. (2020). NextPolish: A Fast and Efficient Genome Polishing Tool for Long-Read Assembly. *Bioinformatics*, 36(7), 2253-2255. <https://doi.org/10.1093/bioinformatics/btz891>

- Hyatt, D., Chen, G.-L., LoCascio, P. F., Land, M. L., Larimer, F. W., & Hauser, L. J. (2010). Prodigal: Prokaryotic Gene Recognition and Translation Initiation Site Identification. *BMC Bioinformatics*, 11(1), 119. <https://doi.org/10.1186/1471-2105-11-119>
- Kang, D. D., Li, F., Kirton, E., Thomas, A., Egan, R., An, H., & Wang, Z. (2019). MetaBAT 2: An Adaptive Binning Algorithm for Robust and Efficient Genome Reconstruction from Metagenome Assemblies. *PeerJ*, 7, e7359. <https://doi.org/10.7717/peerj.7359>
- Katz, K., Shutov, O., Lapoint, R., Kimelman, M., Brister, J., & O'Sullivan, C. (2022). The Sequence Read Archive: a decade more of explosive growth. *Nucleic Acids Research*, 50(D1), D387-D390. <https://doi.org/10.1093/nar/gkab1053>
- Kim, D., Song, L., Breitwieser, F. P., & Salzberg, S. L. (2016). Centrifuge: Rapid and Sensitive Classification of Metagenomic Sequences. *Genome Research*. <https://doi.org/10.1101/gr.210641.116>
- Kodaman, N., Pazos, A., Schneider, B. G., Piazuelo, M. B., Mera, R., Sobota, R. S., Sicinski, L. A., Shaffer, C. L., Romero-Gallo, J., Sablet, T. de, Harder, R. H., Bravo, L. E., Peek, R. M., Wilson, K. T., Cover, T. L., Williams, S. M., & Correa, P. (2014). Human and Helicobacter pylori coevolution shapes the risk of gastric disease. *Proceedings of the National Academy of Sciences*, 111(4), 1455-1460. <https://doi.org/10.1073/pnas.1318093111>
- Kolmogorov, M., Raney, B., Paten, B., & Pham, S. (2014). Ragout—a Reference-Assisted Assembly Tool for Bacterial Genomes. *Bioinformatics*, 30(12), i302-i309. <https://doi.org/10.1093/bioinformatics/btu280>
- Kolmogorov, M., Yuan, J., Lin, Y., & Pevzner, P. A. (2019). Assembly of Long, Error-Prone Reads Using Repeat Graphs. *Nature Biotechnology*, 37(5), 540-546. <https://doi.org/10.1038/s41587-019-0072-8>
- Koren, S., & Phillippy, A. M. (2015). One Chromosome, One Contig: Complete Microbial Genomes from Long-Read Sequencing and Assembly. *Current Opinion in Microbiology*, 23, 110-120. <https://doi.org/10.1016/j.mib.2014.11.014>
- Koren, S., Walenz, B. P., Berlin, K., Miller, J. R., Bergman, N. H., & Phillippy, A. M. (2017). Canu: Scalable and Accurate Long-Read Assembly via Adaptive k-Mer Weighting and Repeat Separation. *Genome Research*, 27(5), 722-736. <https://doi.org/10.1101/gr.215087.116>
- Laetsch, D. R., & Blaxter, M. L. (2017, julio 31). BlobTools: Interrogation of Genome Assemblies. <https://doi.org/10.12688/f1000research.12232.1>
- Lamkiewicz, K., Barf, L.-M., Sachse, K., & Hölzer, M. (2024). RIBAP: A Comprehensive Bacterial Core Genome Annotation Pipeline for Pangenome Calculation beyond the Species Level. *Genome Biology*, 25(1), 170. <https://doi.org/10.1186/s13059-024-03312-9>
- Lee, I., Chalita, M., Ha, S.-M., Na, S.-I., Yoon, S.-H., & Chun, J. (2017). ContEst16S: An Algorithm That Identifies Contaminated Prokaryotic Genomes Using 16S RNA

- Gene Sequences. *International Journal of Systematic and Evolutionary Microbiology*, 67(6), 2053-2057. <https://doi.org/10.1099/ijsem.0.001872>
- Li, D., Liu, C.-M., Luo, R., Sadakane, K., & Lam, T.-W. (2015). MEGAHIT: An Ultra-Fast Single-Node Solution for Large and Complex Metagenomics Assembly via Succinct de Bruijn Graph. *Bioinformatics*, 31(10), 1674-1676. <https://doi.org/10.1093/bioinformatics/btv033>
- Lu, J., Breitwieser, F. P., Thielen, P., & Salzberg, S. L. (2017). Bracken: Estimating Species Abundance in Metagenomics Data. *PeerJ Computer Science*, 3, e104. <https://doi.org/10.7717/peerj-cs.104>
- Lu, J., Rincon, N., Wood, D. E., Breitwieser, F., Pockrandt, C., Langmead, B., Salzberg, S. L., & Steinegger, M. (2022). Metagenome Analysis Using the Kraken Software Suite. *Nature protocols*, 17(12), 2815-2839. <https://doi.org/10.1038/s41596-022-00738-y>
- Mapleson, D., Garcia Accinelli, G., Kettleborough, G., Wright, J., & Clavijo, B. J. (2017). KAT: A K-mer Analysis Toolkit to Quality Control NGS Datasets and Genome Assemblies. *Bioinformatics*, 33(4), 574-576. <https://doi.org/10.1093/bioinformatics/btw663>
- Marçais, G., & Kingsford, C. (2011). A Fast, Lock-Free Approach for Efficient Parallel Counting of Occurrences of k-Mers. *Bioinformatics*, 27(6), 764-770. <https://doi.org/10.1093/bioinformatics/btr011>
- Mikheenko, A., Saveliev, V., & Gurevich, A. (2016). MetaQUAST: Evaluation of Metagenome Assemblies. *Bioinformatics*, 32(7), 1088-1090. <https://doi.org/10.1093/bioinformatics/btv697>
- National Center for Biotechnology Information. (2024). *SRA Toolkit: Collection of Tools and Libraries for Using Data in the INSDC Sequence Read Archives* [Software]. <https://github.com/ncbi/sra-tools>
- Nurk, S., Meleshko, D., Korobeynikov, A., & Pevzner, P. A. (2017). metaSPAdes: A New Versatile Metagenomic Assembler. *Genome Research*, 27(5), 824-834. <https://doi.org/10.1101/gr.213959.116>
- O'Leary, N. A., Cox, E., Holmes, J. B., Anderson, W. R., Falk, R., Hem, V., Tsuchiya, M. T. N., Schuler, G. D., Zhang, X., Torcivia, J., Ketter, A., Breen, L., Cothran, J., Bajwa, H., Tinne, J., Meric, P. A., Hlavina, W., & Schneider, V. A. (2024). Exploring and Retrieving Sequence and Metadata for Species across the Tree of Life with NCBI Datasets. *Scientific Data*, 11(1), 732. <https://doi.org/10.1038/s41597-024-03571-y>
- Page, A. J., Cummins, C. A., Hunt, M., Wong, V. K., Reuter, S., Holden, M. T. G., Fookes, M., Falush, D., Keane, J. A., & Parkhill, J. (2015). Roary: Rapid Large-Scale Prokaryote Pan Genome Analysis. *Bioinformatics*, 31(22), 3691-3693. <https://doi.org/10.1093/bioinformatics/btv421>
- Parks, D. H., Imelfort, M., Skennerton, C. T., Hugenholtz, P., & Tyson, G. W. (2015). CheckM: Assessing the Quality of Microbial Genomes Recovered from Isolates, Single Cells, and Metagenomes. *Genome Research*, 25(7), 1043-1055. <https://doi.org/10.1101/gr.186072.114>

- Peng, Y., Leung, H. C. M., Yiu, S. M., & Chin, F. Y. L. (2012). IDBA-UD: A de Novo Assembler for Single-Cell and Metagenomic Sequencing Data with Highly Uneven Depth. *Bioinformatics*, 28(11), 1420-1428. <https://doi.org/10.1093/bioinformatics/bts174>
- Prjibelski, A., Antipov, D., Meleshko, D., Lapidus, A., & Korobeynikov, A. (2020). Using SPAdes De Novo Assembler. *Current Protocols in Bioinformatics*, 70(1), e102. <https://doi.org/10.1002/cpbi.102>
- Rice, E. S., & Green, R. E. (2019). New Approaches for Genome Assembly and Scaffolding. *Annual Review of Animal Biosciences*, 7, 17-40. <https://doi.org/10.1146/annurev-animal-020518-115344>
- Robinson, J. T., Thorvaldsdóttir, H., Winckler, W., Guttman, M., Lander, E. S., Getz, G., & Mesirov, J. P. (2011). Integrative Genomics Viewer. *Nature biotechnology*, 29(1), 24-26. <https://doi.org/10.1038/nbt.1754>
- Sahlin, K., Vezzi, F., Nystedt, B., Lundberg, J., & Arvestad, L. (2014). BESST - Efficient Scaffolding of Large Fragmented Assemblies. *BMC Bioinformatics*, 15(1), 281. <https://doi.org/10.1186/1471-2105-15-281>
- Seemann, T. (2014). Prokka: rapid prokaryotic genome annotation. *Bioinformatics*, 30(14), 2068-2069. <https://doi.org/10.1093/bioinformatics/btu153>
- Sieber, C. M. K., Probst, A. J., Sharrar, A., Thomas, B. C., Hess, M., Tringe, S. G., & Banfield, J. F. (2018). Recovery of Genomes from Metagenomes via a Dereplication, Aggregation and Scoring Strategy. *Nature Microbiology*, 3(7), 836-843. <https://doi.org/10.1038/s41564-018-0171-1>
- Simão, F. A., Waterhouse, R. M., Ioannidis, P., Kriventseva, E. V., & Zdobnov, E. M. (2015). BUSCO: Assessing Genome Assembly and Annotation Completeness with Single-Copy Orthologs. *Bioinformatics*, 31(19), 3210-3212. <https://doi.org/10.1093/bioinformatics/btv351>
- Souvorov, A., Agarwala, R., & Lipman, D. J. (2018). SKESA: Strategic k-Mer Extension for Scrupulous Assemblies. *Genome Biology*, 19(1), 153. <https://doi.org/10.1186/s13059-018-1540-z>
- Tonkin-Hill, G., MacAlasdair, N., Ruis, C., Weimann, A., Horesh, G., Lees, J. A., Gladstone, R. A., Lo, S., Beaudoin, C., Floto, R. A., Frost, S. D. W., Corander, J., Bentley, S. D., & Parkhill, J. (2020). Producing Polished Prokaryotic Pangenomes with the Panaroo Pipeline. *Genome Biology*, 21(1), 180. <https://doi.org/10.1186/s13059-020-02090-4>
- Walker, B. J., Abeel, T., Shea, T., Priest, M., Abouelliel, A., Sakthikumar, S., Cuomo, C. A., Zeng, Q., Wortman, J., Young, S. K., & Earl, A. M. (2014). Pilon: An Integrated Tool for Comprehensive Microbial Variant Detection and Genome Assembly Improvement. *PLOS ONE*, 9(11), e112963. <https://doi.org/10.1371/journal.pone.0112963>
- Wick, R. R., Judd, L. M., Gorrie, C. L., & Holt, K. E. (2017). Unicycler: Resolving Bacterial Genome Assemblies from Short and Long Sequencing Reads. *PLOS Computational Biology*, 13(6), e1005595. <https://doi.org/10.1371/journal.pcbi.1005595>

- Wick, R. R., Schultz, M. B., Zobel, J., & Holt, K. E. (2015). Bandage: interactive visualization of de novo genome assemblies. *Bioinformatics*, 31(20), 3350-3352. <https://doi.org/10.1093/bioinformatics/btv383>
- Wingett, S. W., & Andrews, S. (2018, septiembre 17). *FastQ Screen: A Tool for Multi-Genome Mapping and Quality Control*. <https://doi.org/10.12688/f1000research.15931.2>
- Wood, D. E., Lu, J., & Langmead, B. (2019). Improved Metagenomic Analysis with Kraken 2. *Genome Biology*, 20(1), 257. <https://doi.org/10.1186/s13059-019-1891-0>
- Wu, Y.-W., Simmons, B. A., & Singer, S. W. (2016). MaxBin 2.0: An Automated Binning Algorithm to Recover Genomes from Multiple Metagenomic Datasets. *Bioinformatics*, 32(4), 605-607. <https://doi.org/10.1093/bioinformatics/btv638>
- Xu, G.-C., Xu, T.-J., Zhu, R., Zhang, Y., Li, S.-Q., Wang, H.-W., & Li, J.-T. (2019). LR_Gapcloser: A Tiling Path-Based Gap Closer That Uses Long Reads to Complete Genome Assembly. *GigaScience*, 8(1), giy157. <https://doi.org/10.1093/gigascience/giy157>
- Xu, M., Guo, L., Gu, S., Wang, O., Zhang, R., Peters, B. A., Fan, G., Liu, X., Xu, X., Deng, L., & Zhang, Y. (2020). TGS-GapCloser: A Fast and Accurate Gap Closer for Large Genomes with Low Coverage of Error-Prone Long Reads. *GigaScience*, 9(9), giaa094. <https://doi.org/10.1093/gigascience/giaa094>
- Zerbino, D. R., & Birney, E. (2008). Velvet: Algorithms for de Novo Short Read Assembly Using de Bruijn Graphs. *Genome Research*, 18(5), 821-829. <https://doi.org/10.1101/gr.074492.107>



Glosario

- ADN (DNA).** Ácido Desoxirribonucleico. Molécula que contiene la información genética de los organismos.
- Adaptadores de secuenciación.** Secuencias cortas añadidas a fragmentos de ADN para facilitar su amplificación y secuenciación.
- Afinidad taxonómica.** Grado de similitud o relación evolutiva entre organismos, determinado mediante comparación de características genéticas o fenotípicas.
- Agrupamiento (*binning*).** Proceso de agrupar secuencias en metagenómica para identificar organismos presentes en una muestra.
- Ambiente virtual.** Entorno aislado que permite instalar paquetes y dependencias sin afectar el sistema principal.
- ARN (RNA).** Ácido Ribonucleico. Molécula que participa en la síntesis de proteínas y regulación génica, incluyendo ARNm, ARNt y ARNr.
- ARN de transferencia (ARNt o *tRNA*).** Tipo de ARN que transporta aminoácidos específicos durante la síntesis de proteínas, actuando como adaptador entre el código genético y la secuencia de aminoácidos.
- ARN ribosomal (ARNr o *rRNA*).** Componente estructural y catalítico de los ribosomas, esencial para la síntesis de proteínas y altamente conservado evolutivamente.
- ARN ribosomal 16S (ARNr 16S o *16S rRNA*).** Subunidad ribosómica pequeña del ARN ribosomal, altamente conservada evolutivamente, utilizada como marcador molecular para identificación taxonómica y detección de contaminación en genomas bacterianos.
- Artemis.** Herramienta de visualización y edición de genomas que permite examinar secuencias genómicas, anotaciones y realizar análisis comparativos.
- Biblioteca de ADN.** Colección de fragmentos de ADN preparados para secuenciación, donde cada fragmento ha sido procesado y adaptado para ser compatible con una plataforma de secuenciación específica.
- BLAST.** *Basic Local Alignment Search Tool*. Herramienta fundamental para búsquedas de homología entre secuencias biológicas mediante alineamiento local.

- BUSCO.** *Benchmarking Universal Single-Copy Orthologs.* Herramienta para evaluar la completitud biológica de ensamblajes genómicos mediante la identificación de genes ortólogos universales de copia única específicos por linaje taxonómico.
- Canónico.** Forma estándar o representativa de una secuencia de ADN que considera ambas hebras complementarias, eligiendo la lexicográficamente menor para análisis computacionales.
- Cepa.** Variante genética de una especie bacteriana, generalmente aislada de una fuente específica y mantenida en cultivo puro.
- Código genético.** Sistema universal de reglas que define cómo la información contenida en el ADN y ARN se traduce en secuencias de aminoácidos para formar proteínas.
- Completitud biológica.** Grado en que un ensamblaje genómico contiene el conjunto completo de genes esperados para un organismo, especialmente genes esenciales y conservados.
- Conda.** Sistema de gestión de paquetes y ambientes virtuales para software científico y de bioinformática.
- Conda-forge.** Comunidad y repositorio de paquetes para Conda, que ofrece software actualizado y de fácil instalación.
- Cobertura.** Número promedio de veces que cada posición del genoma es representada por las lecturas de secuenciación. Una mayor cobertura proporciona mayor confianza en la precisión de la secuencia ensamblada.
- Cobertura desigual.** Variación no uniforme en el número de lecturas que cubren diferentes regiones del genoma, resultando en algunas áreas con alta cobertura y otras con baja cobertura.
- Contigüidad.** Medida de continuidad en un ensamblaje genómico que indica qué tan bien conectadas están las secuencias, reflejada en métricas como N50 y número total de contigs.
- Control de calidad (QC del inglés *Quality Control*).** Proceso de evaluación de la calidad de las lecturas de secuenciación, eliminando errores, contaminantes y artefactos.
- Contenido GC.** Porcentaje de nucleótidos guanina (G) y citosina (C) en una secuencia de ADN, característica importante para análisis genómicos.
- datasets.** Herramienta para acceder y descargar información genómica, incluyendo metadatos, desde las bases de datos del NCBI.
- Deleción.** Tipo de mutación que implica la pérdida de uno o más nucleótidos en una secuencia de ADN.

- Descontaminación.** Proceso de identificar y remover secuencias contaminantes de datos de secuenciación o ensamblajes genómicos, mejorando la pureza y precisión del análisis.
- Diploide.** Organismo o célula que posee dos copias de cada cromosoma, una de cada progenitor.
- Directorio personal (home).** Directorio personal del usuario en sistemas Unix/Linux, típicamente `/home/usuario`, donde se almacenan archivos y configuraciones personales.
- Distancia de Jaccard (Jaccard distance).** Métrica de disimilitud entre conjuntos, calculada como $1 - (\text{intersección} / \text{unión})$, utilizada en análisis de k-meros para evaluar completitud de ensamblajes.
- eggNOG-mapper.** Herramienta para anotación funcional que asigna funciones a genes mediante búsquedas en la base de datos eggNOG de grupos ortólogos.
- Eucariota.** Organismo con células que poseen núcleo membranos definido y orgánulos especializados, incluyendo plantas, animales, hongos y protistas.
- Ensamblador de novo.** Software que reconstruye secuencias genómicas sin referencia previa, usando únicamente los datos de secuenciación.
- Ensamblaje.** Proceso de reconstrucción de una secuencia genómica a partir de fragmentos cortos obtenidos por secuenciación.
- FASTA.** Formato de archivo que contiene secuencias de nucleótidos o aminoácidos sin información de calidad. Usado para representar ensamblajes o referencias.
- FastANI.** Herramienta para calcular identidad nucleotídica promedio entre genomas mediante alineamiento rápido de fragmentos.
- FastQC.** Herramienta para el control de calidad en datos de secuenciación de alto rendimiento.
- Fidelidad de secuencia.** Exactitud de la secuencia ensamblada comparada con el genoma original del organismo, medida por la ausencia de errores como sustituciones, inserciones o deleciones.
- FASTQ.** Formato de archivo que almacena secuencias de ADN junto con sus puntuaciones de calidad. Es ampliamente usado en datos de secuenciación masiva.
- fastp.** Herramienta para el pre-procesamiento rápido de archivos FastQ.
- Fragmentación.** Grado de división de un ensamblaje genómico en múltiples contigs, donde mayor fragmentación indica menor continuidad del ensamblaje.

- Genes marcadores.** Conjunto de genes conservados utilizados como referencia para evaluar la completitud y calidad de ensamblajes genómicos, como los genes ortólogos universales de copia única.
- Genes ortólogos universales.** Genes conservados presentes en una sola copia en la mayoría de los organismos de un linaje taxonómico específico, utilizados para evaluar completitud del ensamblaje.
- GenBank.** Base de datos pública del NCBI que almacena secuencias genéticas anotadas y metadatos asociados, utilizada para depósito y consulta de información genómica.
- GenBank (formato de archivo).** Formato estándar para almacenar secuencias genéticas con anotaciones detalladas, incluyendo características genómicas y metadatos, compatible con herramientas de visualización y análisis.
- Geneious.** Software comercial de bioinformática para análisis, visualización y anotación de secuencias genómicas con interfaz gráfica intuitiva.
- Grafo de De Bruijn (De Bruijn graph).** Representación gráfica usada en ensamblaje de genomas, donde los nodos son k -meros y las aristas representan solapamientos. Permite reconstruir secuencias a partir de lecturas cortas.
- Haploide.** Organismo o célula que posee una sola copia de cada cromosoma, típico en gametos o ciertos microorganismos.
- Haplotipo.** Conjunto de alelos estrechamente ligados en un cromosoma que tienden a heredarse juntos como una unidad.
- Heterocigosidad.** Presencia de diferentes alelos en *loci* homólogos de cromosomas pareados en organismos diploides.
- Hilo de procesamiento (thread).** Hilo de ejecución en un procesador, utilizado para realizar tareas en paralelo y acelerar procesos computacionales.
- HMMER.** Suite de programas para búsquedas sensibles de homología entre secuencias utilizando modelos ocultos de Markov (HMM).
- Homocigoto.** Individuo que posee alelos idénticos en un *locus* particular de cromosomas homólogos.
- Identidad nucleotídica.** Porcentaje de nucleótidos idénticos entre dos secuencias de ADN cuando se alinean, utilizado como medida de similitud entre secuencias.
- Illumina.** Plataforma de secuenciación de alto rendimiento basada en síntesis, ampliamente utilizada en genómica.

- Indicador de comandos (*prompt*).** Texto mostrado por el intérprete de comandos (*shell*) que indica que el sistema está listo para recibir instrucciones. Típicamente incluye información como el usuario actual, el directorio de trabajo y un símbolo terminal (\$) para usuarios normales, # para administrador).
- Inserción.** Tipo de mutación que involucra la adición de uno o más nucleótidos en una secuencia de ADN.
- InterProScan.** Herramienta para predicción funcional de proteínas mediante búsquedas en múltiples bases de datos de dominios y familias proteicas.
- Jellyfish.** Herramienta rápida y eficiente para el conteo de *k*-meros en secuencias de ADN utilizando *hash tables* optimizados.
- KAT.** *K-mer Analysis Toolkit*. Conjunto de herramientas para análisis comparativo de *k*-meros, especialmente útil para evaluación de calidad de ensamblajes.
- k*-mero (*k-mer*).** Subcadena de longitud *k* extraída de una secuencia. Los *k*-mers son fundamentales en algoritmos de ensamblaje y análisis de secuencias.
- L50.** Número mínimo de contigs cuya suma de longitudes alcanza al menos el 50% del ensamblaje total.
- Lecturas (*reads*).** Fragmentos cortos de secuencia de ADN o ARN obtenidos en un experimento de secuenciación.
- Lecturas largas (*long reads*).** Secuencias de ADN obtenidas por tecnologías como PacBio u Oxford Nanopore, típicamente de varios kilobases de longitud, útiles para resolver regiones repetitivas.
- LG50.** Número mínimo de contigs cuya suma de longitudes cubre el 50% del genoma de referencia.
- Linaje taxonómico.** Grupo de organismos que comparten un ancestro común y una clasificación taxonómica específica, utilizado como referencia para comparaciones filogenéticas.
- Linux.** Sistema operativo de código abierto ampliamente utilizado en bioinformática por su estabilidad y compatibilidad con herramientas de línea de comandos.
- Llenado de espacios (*Gap filling*).** Proceso de llenar espacios vacíos (*gaps*) entre contigs en un scaffold utilizando información adicional como lecturas pareadas o secuencias de solapamiento.
- Mapa óptico.** Tecnología que genera mapas físicos de moléculas individuales de ADN de alto peso molecular, utilizada para validación y mejora de ensamblajes genómicos.

- eggNOG-mapper.** Herramienta para anotación funcional que asigna funciones a genes mediante búsquedas en la base de datos eggNOG de grupos ortólogos.
- Marco abierto de lectura (ORF).** *Open Reading Frame.* Secuencia de ADN que comienza con un codón de inicio (generalmente ATG) y termina con un codón de parada, representando una región que potencialmente codifica para una proteína.
- Mega pares de bases (Mpb o Mbp).** *Mega base pair.* Unidad de medida equivalente a un millón de pares de bases.
- MEGAHIT.** Ensamblador de genomas especializado en metagenómica, diseñado para manejar grandes volúmenes de datos de secuenciación.
- MetaBAT.** Herramienta para binning de secuencias en metagenómica, permitiendo la reconstrucción de genomas individuales a partir de muestras ambientales.
- Metadatos.** Información descriptiva sobre datos de secuenciación que incluye detalles sobre la muestra, condiciones experimentales, parámetros de secuenciación y procesamiento.
- Metagenoma.** Conjunto de material genético recuperado directamente de muestras ambientales, sin necesidad de cultivo previo.
- MetaSPAdes.** Ensamblador de metagenomas basado en SPAdes, optimizado para datos de secuenciación de comunidades microbianas.
- Miniforge.** Distribución ligera de Conda que facilita la instalación de ambientes y paquetes, especialmente en sistemas Linux y ARM.
- MultiQC.** Herramienta para unificar y resumir múltiples reportes generados por FastQC y otros programas de bioinformática.
- N50.** Métrica que indica la longitud mínima de los contigs más largos que juntos suman al menos el 50% del ensamblaje total.
- NG50.** Métrica similar a N50, pero calculada sobre el subconjunto de contigs que cubren el genoma de referencia.
- Núcleo de procesador (core).** Unidad de procesamiento dentro de un CPU capaz de ejecutar instrucciones de manera independiente.
- Número de acceso (accession number).** Identificador único asignado a una muestra o secuencia en una base de datos pública.
- Operon.** Unidad funcional de ADN que contiene múltiples genes bajo el control de un solo promotor, común en procariotas para regulación coordinada de expresión génica.

- Ortólogo.** Gen en diferentes especies que evolucionó de un ancestro común mediante especiación, típicamente conservando la misma función.
- PacBio.** Tecnología de secuenciación de ADN de molécula única, conocida por generar lecturas largas y alta precisión.
- Pares de bases (bp).** Unidad de medida que representa la longitud de una secuencia de ADN o ARN, equivalente a una pareja de nucleótidos.
- Plásmido.** Molécula de ADN extracromosómico presente en bacterias, que puede portar genes de interés como resistencia a antibióticos.
- Prodigal.** Herramienta para predicción de genes en genomas bacterianos y arqueales, utilizada internamente por programas como Prokka y BUSCO.
- Procariota.** Organismo unicelular sin núcleo membranos definido, incluyendo bacterias y arqueas, caracterizados por su material genético disperso en el citoplasma.
- Prokka.** Software para la anotación rápida de genomas bacterianos, integrando múltiples bases de datos y herramientas.
- Pulido (polishing).** Proceso de corrección de errores en secuencias ensambladas utilizando información de lecturas originales para mejorar la exactitud nucleotídica.
- Puntuación Phred (Phred score).** Valor que indica la calidad de una base en una secuencia de ADN, utilizado en el formato FASTQ.
- QUAST.** Herramienta para evaluar la calidad de ensamblajes genómicos, proporcionando métricas como N50, número de contigs y errores.
- Roary.** Herramienta para análisis de pangenoma bacteriano que identifica genes core, accesorio y únicos a partir de genomas anotados.
- Scaffold.** Conjunto de contigs ordenados y orientados con posibles huecos entre ellos, usando información adicional como pares de lectura o mapas físicos.
- Scaffolding.** Proceso de ordenar y orientar contigs en scaffolds utilizando información de conectividad, como lecturas pareadas o datos de mapas físicos, para crear estructuras más largas.
- Secuenciación de alto rendimiento.** Tecnología que permite obtener grandes cantidades de datos de secuencias de ADN o ARN de manera rápida y económica.
- Secuenciación de nueva generación (NGS).** *Next Generation Sequencing.* Tecnologías de secuenciación de ADN de segunda generación que permiten obtener millones de lecturas cortas de forma rápida y económica.

- Secuencia contigua (*contig*).** Fragmento continuo de ADN ensamblado a partir de lecturas cortas. Representa una secuencia sin huecos.
- Secuenciación de célula individual o única (*single-cell sequencing*).** Técnica para secuenciar el ADN de células individuales, permitiendo el estudio de la heterogeneidad celular.
- Secuenciación de genoma completo (WGS).** *Whole Genome Sequencing*. Secuenciación del genoma completo de un organismo, proporcionando datos para análisis genómicos detallados.
- Serotipo.** Clasificación de microorganismos basada en características antigénicas detectables mediante pruebas serológicas.
- Sintenia.** Conservación del orden y orientación de genes u otras características genómicas entre especies o regiones cromosómicas relacionadas.
- SnapGene.** Software comercial para visualización, edición y análisis de secuencias de ADN con interfaz gráfica intuitiva.
- SNPs.** *Single Nucleotide Polymorphisms*. Variaciones de un solo nucleótido en posiciones específicas del ADN entre individuos de la misma especie.
- SPAdes.** Ensamblador de genomas diseñado para datos de secuenciación de Illumina. Utiliza grafos de De Bruijn y es especialmente útil para genomas bacterianos.
- SRA.** *Sequence Read Archive*. Base de datos pública del NCBI que almacena datos crudos de secuenciación de alto rendimiento.
- SRA Tools.** Colección de herramientas para descargar datos desde la base de datos SRA.
- Sustitución.** Tipo de mutación puntual donde un nucleótido es reemplazado por otro en una secuencia de ADN.
- Tabla hash (*hash table*).** Estructura de datos que permite almacenamiento y búsqueda eficiente de información mediante funciones de hash, utilizada en conteo de k-meros.
- tbl2asn.** Herramienta del NCBI para convertir archivos de anotación en formato tabla a archivos ASN.1 requeridos para envío a GenBank.
- Tecnología Oxford Nanopore (*Oxford Nanopore*).** Tecnología de secuenciación de tercera generación que utiliza nanoporos para secuenciar moléculas individuales de ADN, produciendo lecturas muy largas en tiempo real.
- Terminal.** Interfaz de línea de comandos que permite a los usuarios interactuar con el sistema operativo mediante texto, ejecutando comandos y programas sin interfaz gráfica.

Tipificación molecular. Conjunto de métodos que utilizan características genéticas para clasificar y diferenciar microorganismos a nivel molecular.

Tipificación multilocus (MLST). *Multilocus Sequence Typing.* Método de tipificación molecular que caracteriza aislados mediante secuenciación de múltiples loci housekeeping.

Tipificación multilocus del genoma central (cgMLST). *Core genome Multilocus Sequence Typing.* Método de tipificación molecular que analiza múltiples loci del genoma central para caracterización de alta resolución.

Transcriptoma. Conjunto completo de transcritos (ARN) presentes en una célula o población de células en un momento dado.

Transposón. Elemento genético móvil capaz de moverse o copiarse a diferentes ubicaciones dentro del genoma, también conocido como “gen saltarín”.



Índice de figuras

1. Diagrama de flujo de trabajo para el ensamblaje y anotación de genomas bacterianos.	13
2. Representación esquemática de la secuenciación <i>paired-end</i>	29
3. Captura de pantalla de un reporte de calidad generado con la aplicación FastQC.	34
4. Captura de pantalla del reporte de calidad unificado generado con MultiQC.	36
5. Gráfico de calidad de secuenciación por base.	37
6. Gráfico de calidad de secuenciación por base generado sobre los datos limpios.	44
7. Reporte HTML generado por QUAST.	60
8. Gráfico generado por QUAST de la longitud acumulada de los ensamblajes.	63
9. Sección del reporte de MultiQC con los resultados de QUAST y BUSCO.	69
10. Ejemplo de <i>k</i> -meros.	72
11. Espectro de número de copias generado por KAT para la muestra SRR22388518.	81
12. Graficos de espectro de número de copias para las cuatro muestras (lecturas y ensamblajes) analizadas con KAT.	85
13. Visualización del genoma anotado de <i>H. pylori</i> , muestra SRR22388518, en IGV.	100
14. Visualización del genoma anotado de <i>H. pylori</i> , muestra SRR22388518, en Proksee.	101



Índice de tablas

1. Software requerido para la obtención de datos.....	26
2. Software requerido para control de calidad y limpieza de datos.....	32
3. Posibles resultados de la evaluación de calidad realizada por la herramienta FastQC.....	34
4. Software requerido para ensamblaje.....	46
5. Software requerido para control de calidad de ensamblaje.....	56
6. Métricas de calidad de los ensamblajes de genoma de <i>H. pylori</i>	62
7. Resultados de BUSCO para los ensamblajes de genoma de <i>H. pylori</i> con respecto del linaje de los <i>Campylobacterales</i> con un total de 682 genes marcadores.....	70
8. Criterios integrados para evaluación de calidad de ensamblajes bacterianos.....	70
9. Resumen de análisis de <i>k</i> -meros con KAT para los ensamblajes de <i>H. pylori</i>	83
10. Software requerido para anotación.	92
11. Descripción de los archivos de salida generados por Prokka y sus usos principales.....	94
12. Descripción de los campos del formato GFF3 utilizado en el archivo de anotación principal de Prokka.....	96
13. Comparación entre IGV y Proksee para visualización de resultados de anotación.....	102
14. Comparación de plataformas web para análisis genómicos.....	107
15. Tabla comparativa de ensambladores de genomas.	142
16. Problemas comunes y posibles soluciones en ensamblajes con SPAdes...	143
17. Comparación entre ensamblaje de genomas individuales y metagenómico.....	146
18. Herramientas especializadas para análisis metagenómico.....	148

| Apéndice

Apéndice A

Problemas comunes y soluciones sugeridas

Este apéndice describe algunos de los problemas más comunes que se pueden presentar durante el desarrollo del proyecto de ensamblaje de genomas bacterianos, así como las soluciones recomendadas. Los problemas están organizados según las etapas del flujo de trabajo bioinformático: administración de paquetes con Conda, descarga de datos desde repositorios públicos, control de calidad de secuencias, ensamblaje *de novo*, evaluación de calidad de ensamblajes, y anotación funcional. Cada problema incluye una descripción del error o síntoma, las causas probables que lo originan, y múltiples soluciones prácticas con comandos específicos. Esta guía de referencia rápida está diseñada para facilitar la resolución eficiente de inconvenientes técnicos y reducir el tiempo de depuración durante la implementación del proyecto.

A.1. Problemas comunes con Conda

A.1.1. Conflictos de dependencias

Problema/error: mensajes como *“Solving environment: failed”* o *“The following packages are incompatible”* durante la instalación de paquetes.

Causa probable: versiones incompatibles entre paquetes o dependencias conflictivas en el entorno actual.

Soluciones:

- Crear un entorno nuevo específico para el proyecto: `mamba create -n nuevo_entorno`
- Especificar versiones compatibles: `mamba install spades=3.15.5`
- Usar canales específicos: `mamba install -c bioconda -c conda-forge paquete`
- Actualizar conda/mamba: `mamba update mamba`

A.1.2. Entorno no encontrado

Problema/error: *“CondaEnvironmentError: cannot locate environment”* al intentar activar un entorno.

Causa probable: el entorno no existe, fue eliminado, o hay problemas en la ruta de conda.

Soluciones:

- Verificar entornos disponibles: `mamba env list`
- Recrear el entorno si es necesario
- Verificar la instalación de conda/mamba: `which mamba`

A.1.3. Problemas de permisos

Problema/error: *“PermissionError”* o *“Access denied”* durante la instalación.

Causa probable: permisos insuficientes en el directorio de instalación o conflictos con instalaciones del sistema.

Soluciones:

- Instalar en el directorio de usuario (evitar `sudo`)
- Verificar permisos del directorio conda: `ls -la ~/mambaforge`
- Reinstalar mamba en el directorio de usuario

A.1.4. Comandos no encontrados después de la instalación

Problema/error: *“command not found”* para herramientas recién instaladas.

Causa probable: el entorno no está activado o problemas en la variable `PATH`.

Soluciones:

- Activar el entorno: `mamba activate nombre_entorno`
- Verificar la instalación: `mamba list paquete`
- Reiniciar la terminal o recargar el perfil: `source ~/.bashrc`

A.1.5. Lentitud en la resolución de dependencias

Problema/error: proceso extremadamente lento al resolver dependencias.

Causa probable: configuración de canales inadecuada o uso de `conda` en lugar de `mamba`.

Soluciones:

- Usar `mamba` en lugar de `conda`: `mamba install paquete`
- Configurar canales prioritarios: `mamba config --add channels conda-forge`
- Limpiar caché: `mamba clean --all`

A.2. Problemas comunes durante la descarga de datos

A.2.1. Fallas en la descarga desde SRA

Problema/error: “Network error”, “Connection timeout” o descargas que se interrumpen inesperadamente.

Causa probable: problemas de conectividad de red, servidor SRA temporalmente inaccesible, o archivos corruptos durante la transferencia.

Soluciones:

- Reintentar la descarga: usar `fastq-dump` o `fasterq-dump` nuevamente
- Verificar conectividad: `ping www.ncbi.nlm.nih.gov`
- Usar `--retry` con `fastq-dump` de SRA Tools: `fastq-dump --retry 5 --split-files SRR_ID`
- Cambiar ubicación de descarga con SRA Tools: `vdb-config -i`

A.2.2. Archivos FASTQ incompletos o corruptos

Problema/error: archivos con tamaño menor al esperado o errores al intentar abrirlos.

Causa probable: descarga interrumpida, problemas de espacio en disco, o corrupción durante la transferencia.

Soluciones:

- Verificar integridad con `gzip`: `gzip -t archivo.fastq.gz`
- Comprobar espacio disponible: `df -h`
- Re-descargar con validación: `fastq-dump --validate archivo.sra`

- Usar checksums si están disponibles para verificar integridad

A.2.3. Problemas con identificadores SRA

Problema/error: *"Run not found"* o *"Invalid accession"* al intentar descargar.

Causa probable: identificador SRA incorrecto, muestra restringida, o problemas temporales en la base de datos.

Soluciones:

- Verificar el identificador en el sitio web SRA: <https://www.ncbi.nlm.nih.gov/sra>
- Usar [esearch](#) con EDirect: `esearch -db sra -query "SRR_ID"`
- Intentar con prefetch de SRA Tools: `prefetch SRR_ID`
- Verificar restricciones de acceso a los datos

A.3. Problemas comunes durante el control de calidad de los datos

A.3.1. FastQC no genera reportes

Problema/error: FastQC se ejecuta pero no produce archivos HTML o falla silenciosamente.

Causa probable: permisos insuficientes, espacio en disco, o archivos FASTQ corruptos.

Soluciones:

- Verificar permisos del directorio de salida: `ls -la directorio_salida`
- Comprobar espacio en disco: `df -h`
- Probar con un solo archivo: `fastqc archivo.fastq.gz`
- Verificar que Java esté instalado: `java -version`

A.3.2. Interpretación de resultados FastQC

Problema/error: reportes con múltiples alertas rojas sin saber qué hacer.

Causa probable: falta de experiencia en interpretación de métricas de calidad o datos naturalmente problemáticos.

Soluciones:

- Consultar documentación de FastQC: revisar significado de cada métrica
- Enfocarse en métricas críticas: *Per base sequence quality, Adapter Content*
- Comparar con muestras similares del mismo proyecto
- Considerar que algunas alertas pueden ser normales según el tipo de muestra

A.3.3. Errores en fastp durante el preprocesamiento

Problema/error: fastp falla con *"too many failures"* o *"no reads pass filters"*.

Causa probable: filtros demasiado estrictos, datos de muy baja calidad, o parámetros inadecuados.

Soluciones:

- Relajar filtros de calidad: `--qualified_quality_phred 20` (en lugar de 30)
- Ajustar longitud mínima: `--length_required 50`
- Revisar reportes previos de FastQC para ajustar parámetros
- Usar modo menos estricto: `--dont_eval_duplication`

A.4. Problemas comunes durante el ensamblaje

Fallos típicos durante los ensamblajes, como errores de memoria, fragmentaciones inesperadas, o mensajes de advertencia en la generación de contigs.

A.4.1. Errores de memoria insuficiente

Problema/error: *"std::bad_alloc"* o *"Cannot allocate memory"* durante la ejecución de SPAdes.

Causa probable: memoria RAM insuficiente para el tamaño del dataset o configuración inadecuada de memoria.

Soluciones:

- Reducir el parámetro `-m`: `spades.py -m 16` (para 16 GB)
- Usar el modo `--careful` para optimizar memoria
- Filtrar lecturas de baja calidad previamente
- Usar *subsample* de datos para pruebas con seqtk: `seqtk sample`

A.4.2. Ensamblaje fragmentado (muchos contigs pequeños)

Problema/error: miles de contigs cortos (<1000 bp) en lugar de pocos contigs largos.

Causa probable: baja calidad de secuenciación, contaminación, o parámetros inadecuados.

Soluciones:

- Mejorar el preprocesamiento con fastp o Trimmomatic
- Usar `--careful mode` en SPAdes
- Verificar contaminación con FastQC
- Ajustar parámetros de *k*-mero: `--cov-cutoff auto`

A.4.3. Archivos de salida vacíos o incompletos

Problema/error: archivos `contigs.fasta` o `scaffolds.fasta` vacíos o muy pequeños.

Causa probable: falla en el ensamblaje debido a datos insuficientes o de muy baja calidad.

Soluciones:

- Verificar calidad de datos de entrada con FastQC
- Revisar logs de SPAdes: `spades.log`
- Intentar con diferentes parámetros de *k*-mero
- Verificar que los archivos de entrada no estén corruptos

A.4.4. Errores de formato de archivos

Problema/error: *“Invalid FASTQ format”* o errores de *parsing* durante la lectura de archivos.

Causa probable: archivos FASTQ corruptos, formato incorrecto, o caracteres especiales.

Soluciones:

- Verificar integridad: `gzip -t archivo.fastq.gz`
- Validar formato FASTQ con seqkit: `seqkit stats archivo.fastq`
- Reconvertir desde SRA si es necesario con fastq-dump de SRA Tools: `fastq-dump --split-files`
- Verificar codificación de caracteres

A.4.5. SPAdes se detiene inesperadamente

Problema/error: el proceso se interrumpe sin mensaje de error claro o con *“Segmentation fault”*.

Causa probable: problemas de hardware, memoria corrupta, o archivos de entrada dañados.

Soluciones:

- Verificar espacio en disco: `df -h`
- Reintentar con `--continue` si es posible
- Verificar integridad de archivos de entrada
- Monitorear uso de recursos: `htop` o `top`

A.4.6. Advertencias sobre cobertura irregular

Problema/error: mensajes de *“Low coverage”* o *“High coverage variation”* en los logs.

Causa probable: cobertura desigual, posible contaminación, o problemas en la preparación de librerías.

Soluciones:

- Analizar distribución de cobertura con KAT
- Usar `--cov-cutoff` para filtrar regiones de baja cobertura
- Verificar contaminación con herramientas como Kraken2
- Revisar calidad de la preparación de librerías

A.4.7. Problemas con paired-end reads

Problema/error: *“Reads are not properly paired”* o errores en el reconocimiento de pares.

Causa probable: archivos R1 y R2 desincronizados o nombres incorrectos.

Soluciones:

- Verificar sincronización con BBTools: `reformat.sh`
- Usar nombres estándar: `_R1.fastq` y `_R2.fastq`
- Reparar pares con BBTools: `repair.sh in1=R1.fq in2=R2.fq out1=fixed1.fq out2=fixed2.fq`
- Verificar el orden de archivos en el comando SPAdes

A.5. Problemas comunes durante la evaluación de calidad de ensamblajes

A.5.1. QUAST no encuentra ensamblajes

Problema/error: *“No assemblies found”* o QUAST no procesa los archivos FASTA.

Causa probable: rutas incorrectas, archivos FASTA vacíos, o formato incorrecto.

Soluciones:

- Verificar que los archivos existen: `ls -la *.fasta`
- Comprobar formato FASTA con seqkit: `seqkit stats archivo.fasta`
- Usar rutas absolutas en lugar de relativas
- Verificar que los archivos no estén corruptos: `head archivo.fasta`

A.5.2. BUSCO no encuentra genes marcadores

Problema/error: BUSCO reporta 0% de completitud o falla al ejecutar análisis.

Causa probable: base de datos BUSCO incorrecta, genoma de muy baja calidad, o problemas con dependencias.

Soluciones:

- Verificar base de datos disponibles: `busco --list-datasets`
- Usar dataset correcto para bacterias: `busco --lineage bacteria_odb10`
- Actualizar base de datos: `busco --download bacteria_odb10`

- Comprobar que Augustus esté instalado correctamente

A.5.3. Errores en descarga con datasets (NCBI)

Problema/error: “API rate limit exceeded” o fallas en descarga de genomas de referencia.

Causa probable: demasiadas solicitudes a la API de NCBI o problemas de conectividad.

Soluciones:

- Esperar unos minutos y reintentar la descarga
- Usar VPN si hay restricciones geográficas
- Descargar manualmente desde el sitio web de NCBI
- Verificar que el *accession number* sea correcto

A.5.4. KAT produce gráficos vacíos

Problema/error: KAT se ejecuta sin errores pero genera gráficos vacíos o sin datos.

Causa probable: problemas con las librerías de Python/matplotlib, datos insuficientes, o errores en paths.

Soluciones:

- Verificar instalación de matplotlib: `python -c "import matplotlib"`
- Comprobar que los archivos de entrada existen y tienen contenido
- Usar modo verbose para ver más detalles: `kat comp -v`
- Regenerar con diferentes parámetros de *k*-mer: `kat comp -m 21`

A.6. Problemas comunes durante la anotación

A.6.1. Prokka falla en la predicción de genes

Problema/error: “No genes found” o Prokka termina con muy pocas características anotadas.

Causa probable: genoma de muy baja calidad, parámetros inadecuados, o problemas con las bases de datos.

Soluciones:

- Verificar calidad del ensamblaje previamente
- Usar `--force` para sobrescribir resultados previos
- Actualizar bases de datos: `prokka --setupdb`
- Probar con `--kingdom Bacteria` explícitamente

A.6.2. Errores de dependencias en Prokka

Problema/error: “*Command not found*” para herramientas internas como `tblastn` o `hmmscan`.

Causa probable: instalación incompleta de Prokka o dependencias faltantes.

Soluciones:

- Reinstalar Prokka con todas las dependencias: `mamba install -c bioconda prokka`
- Verificar dependencias: `prokka --listdb`
- Actualizar el entorno: `mamba update --all`
- Comprobar que todas las herramientas están en PATH

A.6.3. Problemas con archivos de salida de Prokka

Problema/error: archivos `.gbk` o `.gff` malformados que causan errores en herramientas posteriores.

Causa probable: interrupciones durante la ejecución, caracteres especiales en nombres, o *bugs* en versiones específicas.

Soluciones:

- Usar `--force` para regenerar archivos completos
- Validar formato GFF con herramientas: `gff3_validator archivo.gff`
- Evitar caracteres especiales en `--locustag` y `--prefix`
- Verificar que la ejecución se completó exitosamente (revisar logs)

Apéndice B

Ensambladores

La Tabla 15 (siguiente página) compara varios ensambladores de genomas, destacando sus propósitos, algoritmos, casos de uso recomendados, ventajas y limitaciones. Úsela como referencia para seleccionar el ensamblador más adecuado según sus necesidades específicas.

Tabla 15. Tabla comparativa de ensambladores de genomas.

Ensamblador	Propósito	Algoritmo/Estrategia	Casos de uso	Ventajas	Limitaciones
SPAdes	Ensamblador general para lecturas cortas; variantes meta/híbridas	Grafo De Bruijn multi- <i>k</i> , corrección BayesHammer; variantes meta/RNA/híbridas	Bacterias (Illumina), híbridos (Illumina+largas), metagenomas	Alta contigüidad (N50), corrección integrada, flexible	Alto uso RAM/CPU; muchos parámetros; posibles misassemblies sin ajuste
SKESA	Conservador, optimizado para vigilancia	Extensión estratégica <i>k</i> -mers, heurísticas conservadoras	Vigilancia bacteriana, pipelines MLST/cgMLST	Rápido, baja memoria, determinista, pocos misassemblies	Más fragmentado que SPAdes; no híbrido; pocos modos
Velvet	Clásico De Bruijn para lecturas cortas	Grafo De Bruijn (<i>k</i> único)	Datos pequeños Illumina; pruebas rápidas; recursos limitados	Sencillo, bajo consumo, rápido	Requiere <i>k</i> manual; antiguo; menos funcionalidades
Canu	Para lecturas largas (PacBio/ONT)	OLC, corrección y recorte	Lecturas largas (PacBio/ONT), genomas grandes	Para lecturas largas ruidosas; corrección robusta	No para lecturas cortas; lento en genomas grandes
Unicycler	Híbrido para bacterias (cor-tas+largas)	SPAdes interno + lecturas largas; detecta circularidad	Bacterias Illumina+largas; plásmidos	Excelente para circulares/plásmidos; automatiza flujo híbrido	Solo bacterias; requiere ambos tipos de datos

Apéndice C

Contenido y uso del archivo spades.log

C.1. Contenido del archivo

- Versión de SPAdes y parámetros utilizados
- Información del sistema: CPU, RAM disponible, número de hilos
- Etapas del ensamblaje: lectura de datos, corrección de errores, construcción del grafo, ensamblaje final
- Mensajes de advertencia o error
- Tiempos de ejecución por etapa

C.2. Diagnóstico de problemas

Para identificar problemas en el ensamblaje, revise el archivo [spades.log](#) y busque mensajes de error o advertencia, los cuales generalmente contienen las palabras “ERROR” o “WARNING”. A continuación, se presentan problemas comunes que pueden surgir durante el ensamblaje con SPAdes, junto con las señales en el archivo log y posibles soluciones (Tabla 16).

Tabla 16. Problemas comunes y posibles soluciones en ensamblajes con SPAdes.

Problema común	Señal/mensaje en el log	Posible solución
Lecturas mal formateadas	<i>“Invalid read format”</i> o <i>“Cannot parse read”</i>	Verifique el formato FASTQ y los encabezados
Memoria insuficiente	<i>“Not enough memory to continue”</i> o <i>“Killed”</i>	Ejecute SPAdes con menos hilos (<i>threads</i>) o en una máquina con más RAM
Lecturas de baja calidad	Advertencias en la etapa de corrección	Preprocese con herramientas como FastQC y fastp o Trimmomatic
Parámetros incompatibles	<i>“Unsupported combination of options”</i>	Revise la línea de comandos y la documentación

Problema común	Señal/mensaje en el log	Posible solución
Ensamblaje incompleto	El log termina abruptamente o sin reportar el ensamblaje final	Revise si el proceso fue interrumpido o si hubo errores fatales

Apéndice D

Ensamblaje de datos metagenómicos

La metagenómica es una disciplina que estudia el contenido genético colectivo de comunidades microbianas directamente desde muestras ambientales, sin necesidad de aislar y cultivar microorganismos individualmente. Este enfoque permite analizar ecosistemas microbianos complejos como el suelo, océanos, intestino humano, y otros ambientes donde la mayoría de microorganismos no pueden ser cultivados en laboratorio usando métodos tradicionales.

A diferencia del ensamblaje de genomas individuales, donde las lecturas de secuenciación provienen de un organismo único (como los ejemplos de *H. pylori* presentados en este libro), el ensamblaje metagenómico debe reconstruir múltiples genomas simultáneamente a partir de una mezcla de ADN de diferentes especies presentes en abundancias variables. Este proceso presenta desafíos técnicos y computacionales que requieren herramientas especializadas y estrategias metodológicas específicas.

D.1. Diferencias entre ensamblaje genómico y metagenómico

El ensamblaje de genomas individuales y el ensamblaje metagenómico difieren fundamentalmente en varios aspectos, entre los cuales se pueden destacar la complejidad de la muestra, la desigualdad de cobertura, la generación de contenido quimérico, y la resolución taxonómica. A continuación se describe brevemente cada uno de los aspectos mencionados.

- Complejidad de la muestra: en el ensamblaje tradicional de genomas, todas las lecturas provienen de un solo organismo, lo que facilita la identificación de solapamientos entre secuencias y la reconstrucción del genoma. En contraste, las muestras metagenómicas contienen ADN de múltiples especies (desde decenas hasta miles), cada una presente en diferentes abundancias. Esta heterogeneidad complica significativamente la identificación de qué lecturas pertenecen a cada organismo.

- Cobertura desigual: los genomas individuales típicamente presentan cobertura relativamente uniforme, mientras que en metagenomas la cobertura varía entre especies según su abundancia relativa en la muestra. Especies abundantes pueden tener alta cobertura ($>100\times$), mientras que especies raras pueden presentar cobertura muy baja ($<1\times$), lo que dificulta su ensamblaje.
- Regiones conservadas y quimeras: las secuencias conservadas entre especies relacionadas (como genes ribosomales 16S rRNA) pueden generar *scaffolds* quiméricos que combinan fragmentos de múltiples organismos. Los ensambladores tradicionales pueden interpretar incorrectamente estas regiones como pertenecientes a un solo genoma.
- Resolución taxonómica: el ensamblaje genómico busca reconstruir un genoma completo y continuo, mientras que el ensamblaje metagenómico debe además identificar y separar los genomas de diferentes especies presentes en la muestra. Esta separación requiere análisis de composición nucleotídica, patrones de cobertura, y marcadores taxonómicos.

La Tabla 17 resume las diferencias entre el ensamblaje de genomas individuales y el ensamblaje metagenómico.

Tabla 17. Comparación entre ensamblaje de genomas individuales y metagenómico.

Aspecto	Ensamblaje de genomas individuales	Ensamblaje metagenómico
Origen de lecturas	Un solo organismo (monocultura)	Múltiples organismos (comunidad microbiana)
Complejidad de muestra	Baja (homogénea)	Alta (heterogénea, 10-1000+ especies)
Cobertura	Relativamente uniforme ($20 - 100\times$)	Muy variable ($0.1 - 500\times$) según abundancia
Objetivo principal	Genoma completo y continuo	Múltiples genomas + estructura comunitaria
Desafíos principales	Regiones repetitivas, poliploidía	Separación de especies, regiones conservadas
Riesgo de quimeras	Bajo	Alto (genes ribosomales, elementos móviles)
Herramientas típicas	SPAdes, Canu, Velvet	MetaSPAdes, MEGAHIT, IDBA-UD
Pasos adicionales	Pulido, <i>scaffolding</i>	<i>Binning</i> , refinamiento, clasificación taxonómica
Métricas de calidad	N50, completitud BUSCO	Completitud por <i>bin</i> , contaminación cruzada
Interpretación	Genoma de referencia	Diversidad y función comunitaria

Aspecto	Ensamblaje de genomas individuales	Ensamblaje metagenómico
Recursos computacionales	Moderados	Altos (memoria y tiempo de procesamiento)
Profundidad requerida	30-100×	10-50 Gb según diversidad esperada

D.2. Flujo de trabajo típico

Un análisis metagenómico típico incluye las siguientes etapas:

1. **Preprocesamiento:** control de calidad, eliminación de adaptadores y contaminantes
2. **Ensamblaje:** reconstrucción de los genomas usando herramientas especializadas como MetaSPAdes o MEGAHIT
3. **Predicción de genes:** identificación de marcos abiertos de lectura (ORFs)
4. **Binning:** agrupación de *contigs* en genomas putativos individuales llamados *bins*
5. **Refinamiento:** evaluación y mejora de la calidad de *bins*
6. **Anotación taxonómica:** asignación de identidad taxonómica a *bins*
7. **Anotación funcional:** predicción de funciones metabólicas
8. **Análisis ecológico:** interpretación de estructura y función comunitaria

Este flujo de trabajo integra múltiples herramientas especializadas y requiere consideraciones cuidadosas en cada etapa para obtener resultados biológicamente significativos.

D.3. Herramientas especializadas para ensamblaje metagenómico

Tabla 18. Herramientas especializadas para análisis metagenómico.

Categoría	Herramienta	Función principal	Características clave	Referencias
Ensamblaje	MetaSPAdes	Ensamblador metagenómico	Extensión de SPAdes, manejo de variación de cobertura, prevención de quimeras	Nurk et al. (2017)
	MEGAHIT	Ensamblador ultra-rápido	Eficiente en memoria, k -meros múltiples, algoritmos de grafos sucintos	Li et al. (2015)
	IDBA-UD	Ensamblador para datos desiguales	Especializado en cobertura desigual, k -meros iterativos, optimizado para metagenómica	Peng et al. (2012)
	Flye	Ensamblador híbrido	Optimizado para lecturas largas, adaptado para metagenómica	Kolmogorov et al. (2019)
Binning	MetaBAT 2	Agrupación de contigs	Cobertura diferencial y composición tetranucleótida	Kang et al. (2019)
	MaxBin 2	Agrupación basada en k -meros	Frecuencia de k -meros y patrones de cobertura	Wu et al. (2016)
	CONCOCT	Clustering composicional	Composición nucleotídica y cobertura diferencial multi-muestra	Alneberg et al. (2014)
Refinamiento	DAS Tool	Integración de resultados	Consenso de múltiples herramientas de <i>binning</i>	Sieber et al. (2018)

Categoría	Herramienta	Función principal	Características clave	Referencias
Evaluación	CheckM	Evaluación de calidad	Estimación de completitud y contaminación con marcadores filogenéticos	Parks et al. (2015)
	BUSCO	Completitud biológica	Genes ortólogos universales específicos por linaje	Simão et al. (2015)
	MetaQUAST	Métricas de ensamblaje	Adaptación de QUAST para metagenómica para estadísticas de <i>contigs</i> y <i>scaffolds</i>	Mikheenko et al. (2016)

D.4. Consideraciones técnicas importantes

Entre la consideraciones de carácter técnico para los ensamblajes metagenómicos se pueden destacar las siguientes:

- Contaminación: las muestras metagenómicas son susceptibles a contaminación durante la colección, extracción de ADN, y preparación de bibliotecas. Esta contaminación puede introducir secuencias de organismos no presentes en la muestra original, afectando la interpretación de la comunidad microbiana.
- Sesgos de amplificación: los protocolos de amplificación del ADN pueden introducir sesgos que favorecen ciertos tipos de secuencias. Estos sesgos pueden resultar en representación desigual de diferentes organismos en los datos de secuenciación.
- Profundidad de secuenciación: la detección de especies raras requiere profundidades de secuenciación considerablemente mayores que el ensamblaje de genomas individuales. La relación entre profundidad de secuenciación y detección de diversidad debe considerarse en el diseño experimental.
- Estrategias de muestreo: la representatividad de la muestra es crucial para interpretaciones ecológicas válidas. Consideraciones espaciales, temporales, y metodológicas pueden afectar los resultados del análisis metagenómico.

Acerca de los autores



Héctor Fabio Espitia Navarro



Doctor en bioinformática por el Georgia Institute of Technology e ingeniero en sistemas y magíster en ingeniería por la Universidad del Valle. Actualmente, es especialista en ciencias de datos en Solventum (EE.UU.) y miembro de los grupos de investigación Salud Pública¹ de la Universidad de Nariño y Bioinformática y Biocomputación² de la Universidad del Valle. Sus áreas de trabajo son la transcriptómica, la genómica comparativa y la metagenómica. Su trayectoria incluye la aplicación de la bioinformática al cultivo de caña de azúcar, a la epidemiología molecular y, recientemente, a la investigación en el cuidado de heridas y en la salud dental.

¹ Grupo de Investigación Salud Pública, GrupLac, MinCiencias <https://scienti.minciencias.gov.co/gruplac/jsp/visualiza/visualizagr.jsp?nro=00000000008168>

² Grupo de Investigación de Bioinformática y Biocomputación, GrupLac, Minciencias <https://scienti.minciencias.gov.co/gruplac/jsp/visualiza/visualizagr.jsp?nro=00000000000759>

Alvaro Jairo Pazos Moncayo



Es Profesor Titular del Departamento de Biología de la Universidad de Nariño. En la Universidad del Valle obtuvo los títulos de Bacteriología; Maestría en Microbiología; Doctorado en Ciencias Biomédicas y en Vanderbilt University, EE.UU. desarrollo su tesis doctoral. Hizo un posdoctorado en metagenómica clínica en Georgia Institute of Technology, EE.UU. Como Director del Grupo de Investigación Salud Pública de la Universidad de Nariño, gestionó su reconocimiento por las convocatorias No 894-2021 y No 957-2024 de MinCiencias la máxima clasificación A11. En la convocatoria No 894-2021 obtuvo la categoría de Investigador Emérito con vigencia vitalicia². Su producción intelectual se plasma en su CVLac².

¹ Grupo de Investigación Salud Pública, GrupLac, MinCiencias <https://scienti.minciencias.gov.co/gruplac/jsp/visualiza/visualizagr.jsp?nro=00000000008168>

² Alvaro Pazos, CVLac, MinCiencias https://scienti.minciencias.gov.co/cvlac/visualizador/generar-CurriculoCv.do?cod_rh=0000194972

èditorial

Universidad de **Nariño**

Año de publicación: 2026

San Juan de Pasto - Nariño - Colombia

La secuenciación de genomas bacterianos completos se ha consolidado como una herramienta fundamental en microbiología moderna y epidemiología molecular. Los datos de secuenciación de nueva generación requieren procesamiento bioinformático especializado para transformar millones de lecturas cortas en genomas ensamblados y funcionalmente anotados. En este contexto, las habilidades en bioinformática representan actualmente una necesidad imperativa en el mundo competitivo de las ciencias biológicas, que se apoyan cada vez más en análisis computacionales avanzados. Esta obra surge precisamente para responder a esta necesidad, proporcionando un flujo de trabajo completo, accesible y reproducible para generar genomas bacterianos de calidad apropiada desde datos de secuenciación Illumina.

Utilizando cuatro genomas de *Helicobacter pylori* como caso de estudio, el libro guía al lector a través de ocho capítulos principales: (1) preparación del entorno computacional con Conda/Mamba, (2) obtención de datos desde repositorios públicos usando SRA Tools, (3) control de calidad y limpieza con FastQC y fastp, (4) ensamblaje de novo con SPAdes, (5) evaluación integral mediante QUAST, BUSCO y análisis de k-meros, (6) anotación funcional con Prokka, (7) plataformas web alternativas, y (8) conclusiones. Apéndices especializados complementan el contenido principal con guías de resolución de problemas, comparación de ensambladores alternativos, y recursos avanzados.

El enfoque metodológico combina fundamentos teóricos con implementación práctica, utilizando comandos específicos, interpretación detallada de resultados y buenas prácticas en bioinformática. Todos los scripts y configuraciones están disponibles públicamente en GitHub, garantizando la reproducibilidad completa del flujo de trabajo.

Este libro está dirigido especialmente a estudiantes e investigadores en los campos de la biología y la microbiología que poseen conocimientos básicos de Linux y buscan desarrollar competencias prácticas en bioinformática genómica. La metodología presentada es directamente aplicable en contextos de investigación básica, vigilancia epidemiológica y análisis de genómica comparativa, brindando herramientas versátiles para diversos escenarios profesionales y académicos.

ISBN: 978-628-7864-38-2

