Universidad Nacional Autónoma de México

Centro de Ciencias Genómicas

Proyecto de ensamble de genomas

"*Ensamble del genoma de bacteria perteneciente al género Cupriavidus*"

Miembros del equipo:

López Delgado Phabel Antonio

Montes Gabriel Diego

Ramírez Bernabé Ignacio Emmanuel

1. **Introducción**

El humano casi siempre se ha caracterizado por sus ganas de comprender el mundo que lo rodea, esta hambre de conocimiento nos ha llevado a descubrimientos e hipótesis más que impresionantes, cuando nos comenzamos a preguntar sobre la vida misma, surgieron cuestiones maravillosas ¿Qué es la vida? ¿Cómo surgió la vida? ¿Qué es lo que dictamina el ser? En algún punto nos preguntamos por la herencia, el porqué éramos semejantes a nuestros padres o familiares, para comenzar ¿Dónde guardamos la información que hace posible la herencia? Hoy en día nos parece sencillo decir “Es obvio que en los genes” pero imagina que no te lo haya dicho alguien, que lo tuvieras que descubrir por ti mismo, así no sería obvio; tardamos mucho en saber que la herencia se guardaba dentro del núcleo de las células, lo que hoy en día son los ácidos nucleicos en algún momento fue la nucleina, una sustancia totalmente incomprendida que se sabía que no era ni lípido, ni proteína o glúcido y al conocer parcialmente la funcionalidad de todas estas, y sabiendo que ninguna guardaba la herencia, por descarte se le atribuyó este gran don.

Muchísimos años después en los que se tuvo grandes avances sobre nuestro entendimiento de la genética, ahora sabemos que la información se guarda en genes, estos estaban constituidos por ácidos nucleicos, el ADN tenía una estructura de doble hebra que le permitía replicarse y ser complementarias ¡Vaya! ya teníamos demasiado conocimiento sobre la vida y su herencia! Pero para nosotros no era suficiente, sabíamos que que el lenguaje de la vida estaba expresado en cuatro letras A, T, C y G, la lógica nos dijo que si pudiéramos plasmar estas letras en orden para formar algo como palabras, nos sería similar a leer en un manual para comprender el funcionamiento de una máquina, así queríamos comprender cada aspecto de un organismo, esta idea tan disruptiva e innovadora sería llamada genoma, tener toda letra dentro de nuestra información genética en orden, para formar “palabras” que pudiéramos comprender, obvio no pudimos partir por un genoma complejo, todos conocemos el proyecto del genoma humano, pero pocos recordamos al bacteriófago φX174, esto fue un hito grandioso, en aquella época la secuenciación era muy distinta a como se realiza hoy en día, mucho más complicado, era más “artesanal”; pero sin la secuenciación de Sanger posiblemente no estaríamos escribiendo esto y todos veríamos los genomas como algo exageradamente tedioso, sin embargo incluso con Sanger realizar un ensamble de genoma era algo demasiado complejo ¿Sabes que es lo que vino a facilitarnos la vida? Esta vez la respuesta sí que es obvia, la tecnología, los avances tecnológicos permitieron que la secuenciación y el ensamble fuese mucho más sencillo (por secuenciadores y herramientas bioinformáticas) y esto es lo que nos tiene aquí el ensamble de genoma a través de herramientas bioinformáticas.

2. **Antecedentes**

¿En qué consiste el género Cupriavidus?

Cupriavidus es un género de bacterias, estas bacterias se caracterizan por ser del tipo Gram-negativas; normalmente tienen forma de “bastón” y poseen flagelos, además de ser organismos aeróbicos obligatorios, y son quimioorganotróficos o quimiolitotróficos. El género Cupriavidus fue propuesto en 1987 por Makkar y Casida quienes describieron un depredador bacteriano no obligado de bacterias en el suelo como Cupriavidus necator (Makkar & Casida, 1987), que fue la primera especie encontrada del género Cupriavidus, este género está formado por 17 especies distintas, tomando en cuenta que Wautersia (un género que se integró a Cupriavidus en 2005, porque se determinó que era igual al género Cupriavidus ya existente), así que no es un género extremadamente grande. Cabe aclarar que son 17 especies reconocidas por la comunidad científica, pero que es muy complicado diferenciar especies dentro del dominio de las bacterias (diferenciadas principalmente por las secuencias del gen de ARNr 16S) así que si bien tenemos más estudios que intentan el reconocimiento de nuevas especies, no han sido oficializadas. [5]

¿Cuál es el ambiente óptimo de este género bacteriano?

Miembros del género Cupriavidus han sido aislados de diversos ambientes ecológicos como suelo, agua, sedimentos de estanques, nódulos de leguminosas, depósitos de flujo de lodo volcánico y fuentes clínicas humanas. [5]

¿Y por qué sería de interés este género para considerar su estudio?

Se han encontrado una variedad de funciones fisiológicas en los miembros del género Cupriavidus; por ejemplo, Cupriavidus taiwanensis puede nodular diferentes especies de leguminosas (da Silva et al, 2012), además algunas especies poseen características de promoción del crecimiento de las plantas, como la secreción de ácido indol acético y sideróforos y la solubilización del fosfato (Pongsilp et al,2012), y aún más interesante se ha informado que Cupriavidus pampae y C. numazuensis degradan contaminantes orgánicos como el ácido 4- (2,4-diclorofenoxi) butírico (2,4-DB) y el tricloroetileno. [6]

3. **Preanálisis de Datos y archivos de trabajo**

Los archivos que recibimos eran secuencias obtenidas a partir de la lectura de secuencias de un secuenciador de illumina (de donde obtuvimos las lecturas cortas) y PacBio (que nos proporcionó las lecturas largas). Al analizar en un principio con fastQC pudimos darnos cuenta que, si bien nuestros archivos originales sin depuración alguna no eran de mala calidad, nuestro ensamble final tampoco lo sería. Por lo que nuestra depuración inicial no fue demasiado densa y con pequeños ajustes obtuvimos un salto de calidad considerable tomando en cuenta que tuvimos un excelente punto de partida.

Uso de fastQC

FastQC permite visualizar de una forma más sencilla la calidad de los datos de una secuencia, para poder ser evaluada por nosotros, esta procede de un secuenciador de alto rendimiento como Illumina, PacBio o Nanopore; además es importante porque nos da una visualización general de la calidad de la secuencia, esto nos permite identificar problemas en nuestra información antes de intentar trabajar con esta información.

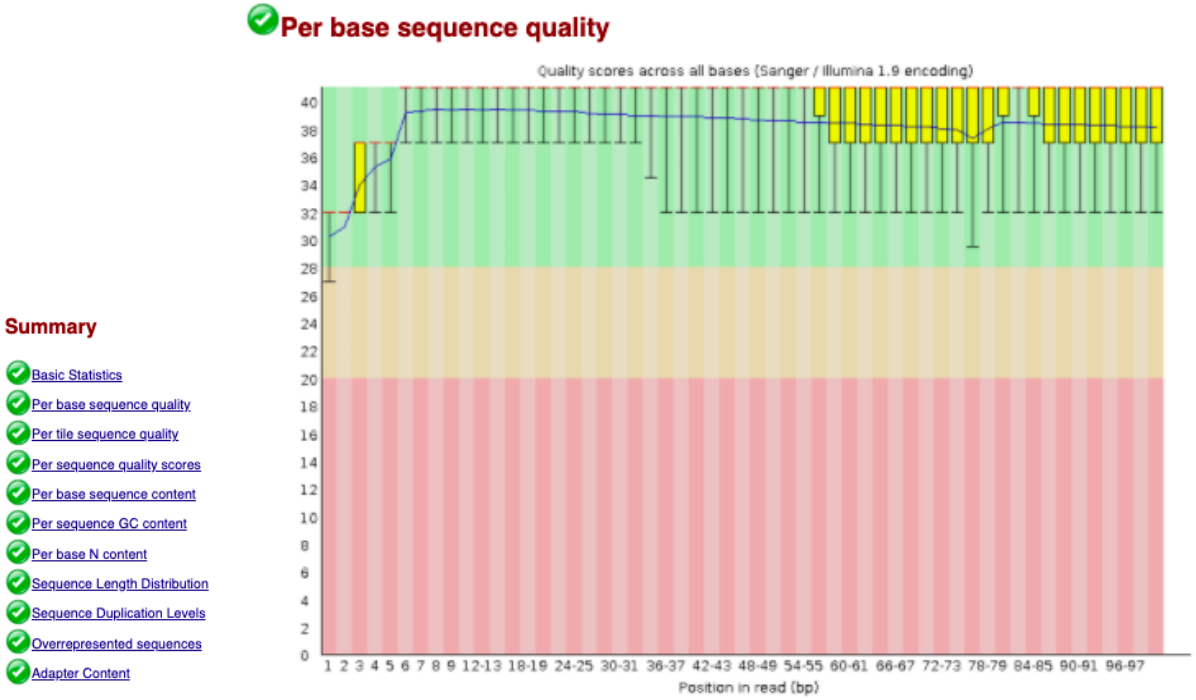
Las principales funciones de FastQC son:

- Importación de datos de archivos BAM, SAM o FastQ (cualquier variante)
- Proporcionar una descripción general rápida para indicarle en qué áreas puede haber problemas.
- Gráficos y tablas de resumen para evaluar rápidamente sus datos
- Exportación de resultados a un informe permanente basado en HTML
- Descarga de los datos, para ser visualizados de manera remota (fuera de la aplicación)

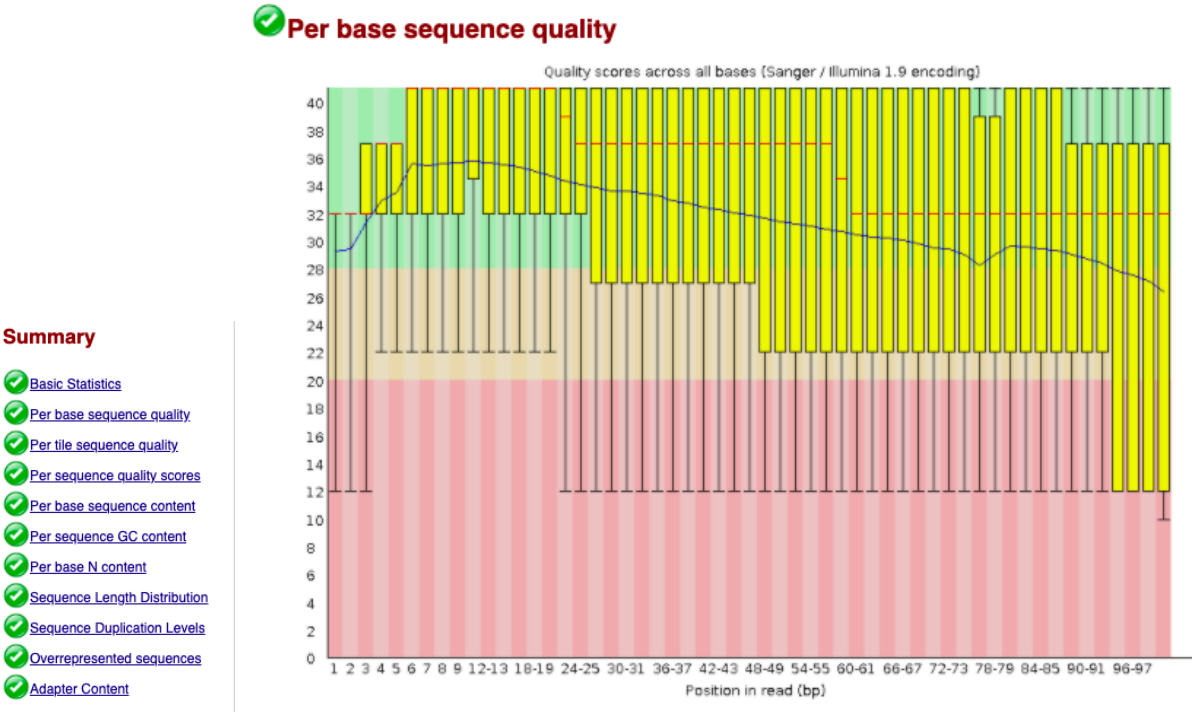
Lo que encontramos en las 2 lecturas iniciales

Como se había mencionado anteriormente, en general eran lecturas de no mala calidad puesto que los indicativos de FastQC tan no arrojaron warnings iniciales, como se puede parcear en las figuras 1 y 2 de los análisis iniciales:

Lectura 1



Lectura 2



No obstante, se consideró que el comportamiento de *Per base sequence quality* de la segunda lectura dejaba qué desear, puesto que los cuantiles llegaban a la zona roja aunque las medias no. Por lo que se consideró realizar una limpieza tanto con trimgalore como con trimmomatic.

4. **Análisis de calidad y limpieza de secuencias**

Trim Galore

Esta herramienta de depuración nos ha sido útil para el ajuste de la calidad de nuestros archivos, primero recortamos del extremo de 3 'de las lecturas antes de quitar el adaptador. Esto eliminó de manera eficiente las porciones de mala calidad de las lecturas.

```
trim_galore --fastqc --paired --retain_unpaired ../CUP_GRP1_1.fq../CUP_GRP1_2.fq

trim_galore --fastqc --three_prime_clip_R1 2 --three_prime_clip_R2 2 --paired --retain_unpaired ../../CUP_GRP1_1.fq ../../CUP_GRP1_2.fq
```

Trimmomatic

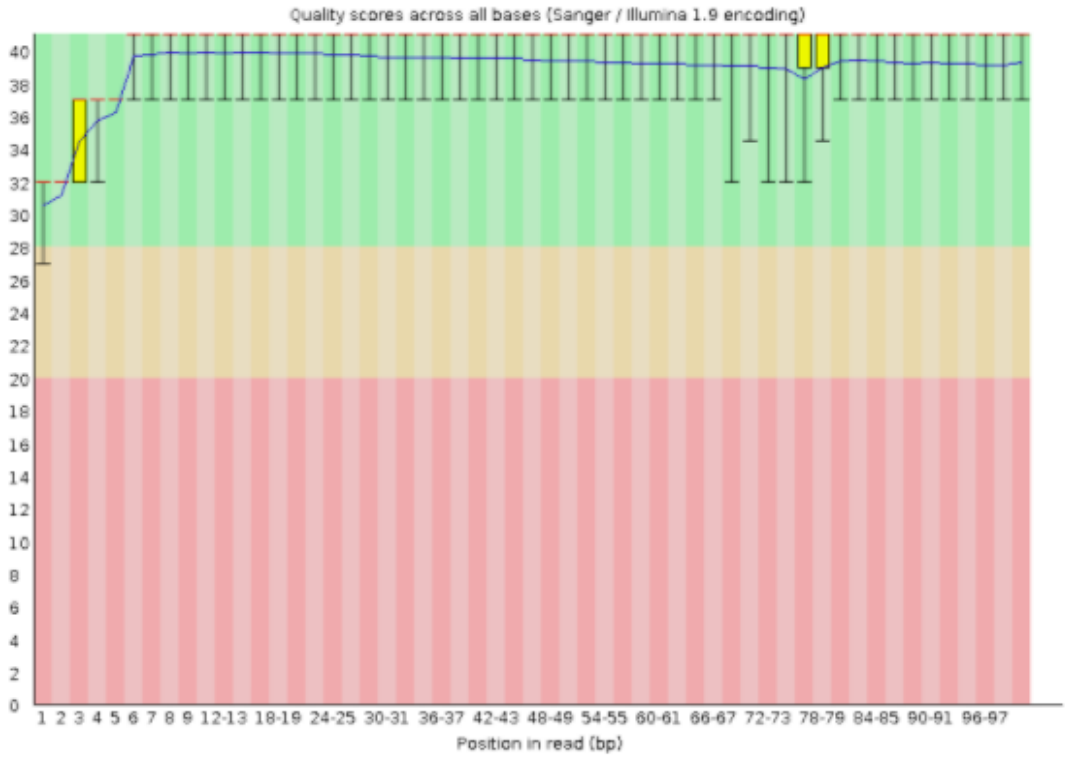
Trimmomatic es una herramienta que nos permite recortar y ajustar los archivos obtenidos de secuenciadores, igual nos permite el ajuste de los adaptadores, al ser multifuncional nos permite hacer ajustes más precisos en nuestra secuencia, con la siguiente línea de comando:

```
trimmomatic PE ../CUP_GRP1_1.fq ../CUP_GRP1_2.fq CUP_GRP1_1_pair.fastq CUP_GRP1_1_unpair.fastq CUP_GRP1_2_pair.fastq CUP_GRP1_2_unpair.fastq ILLUMINACLIP:TrueSeq3-PE.fa:2:30:10 LEADING:3 TRAILING:3 SLIDINGWINDOW:4:15 MINLEN:36
```

Debido a que el ajuste obtenido con esta herramienta fue el que nos arrojó mejores resultados porque no hubo ninguna anomalía dentro del análisis de calidad, principalmente en el apartado de “Per base sequence content” (los análisis de calidad fueron visualizados a través de FastQC), estas fueron las lecturas seleccionadas para trabajar de aquí en adelante, obtuvimos una calidad realmente muy alta (sólo un Warning en “Length distribution” pero que esto no es preocupante), por lo que se esperó que el desarrollo del proyecto después de esta depuración fuera muy satisfactorio. Los cambios principales se pueden apreciar en las siguientes imágenes.

Lectura 1

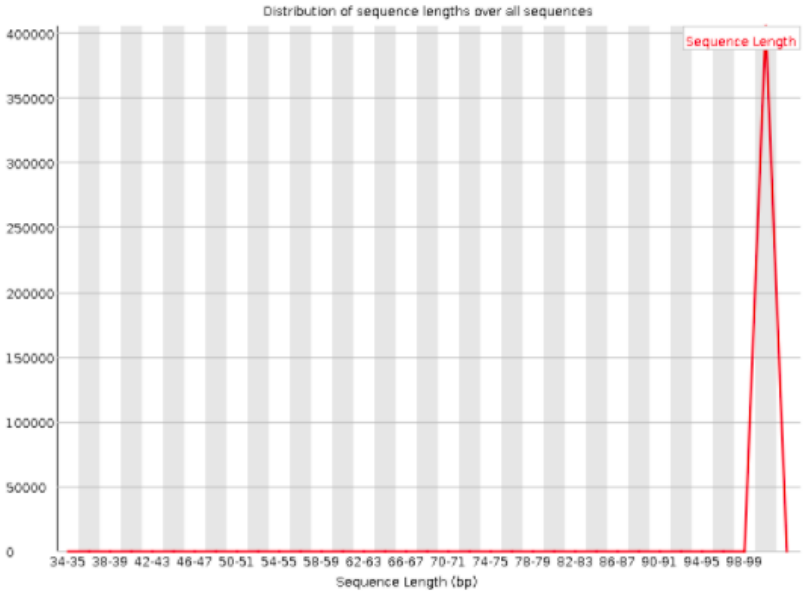
Per base sequence quality



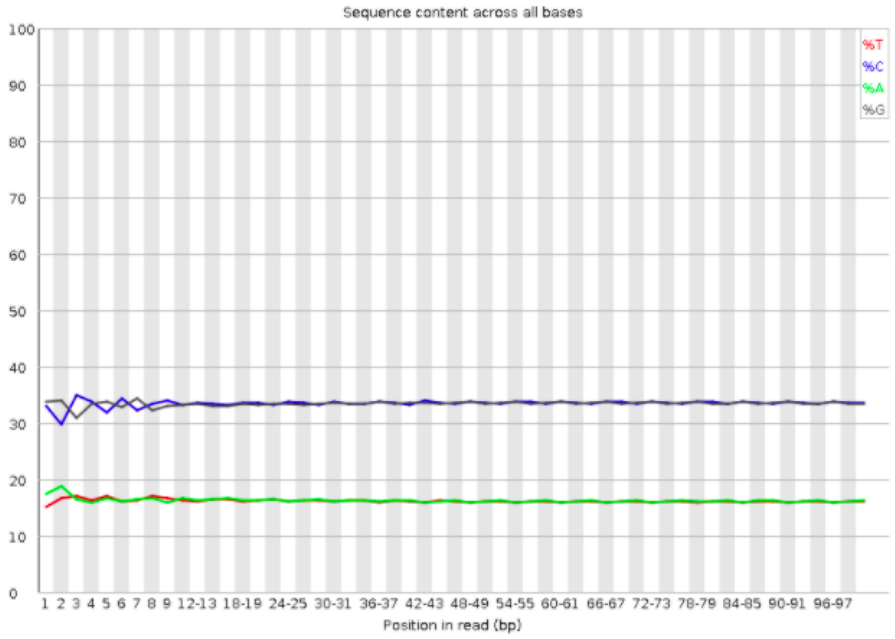
Summary

- Basic Statistics
- Per base sequence quality
- Per tile sequence quality
- Per sequence quality scores
- Per base sequence content
- Per sequence GC content
- Per base N content
- Sequence Length Distribution
- Sequence Duplication Levels
- Overrepresented sequences
- Adapter Content

Sequence Length Distribution

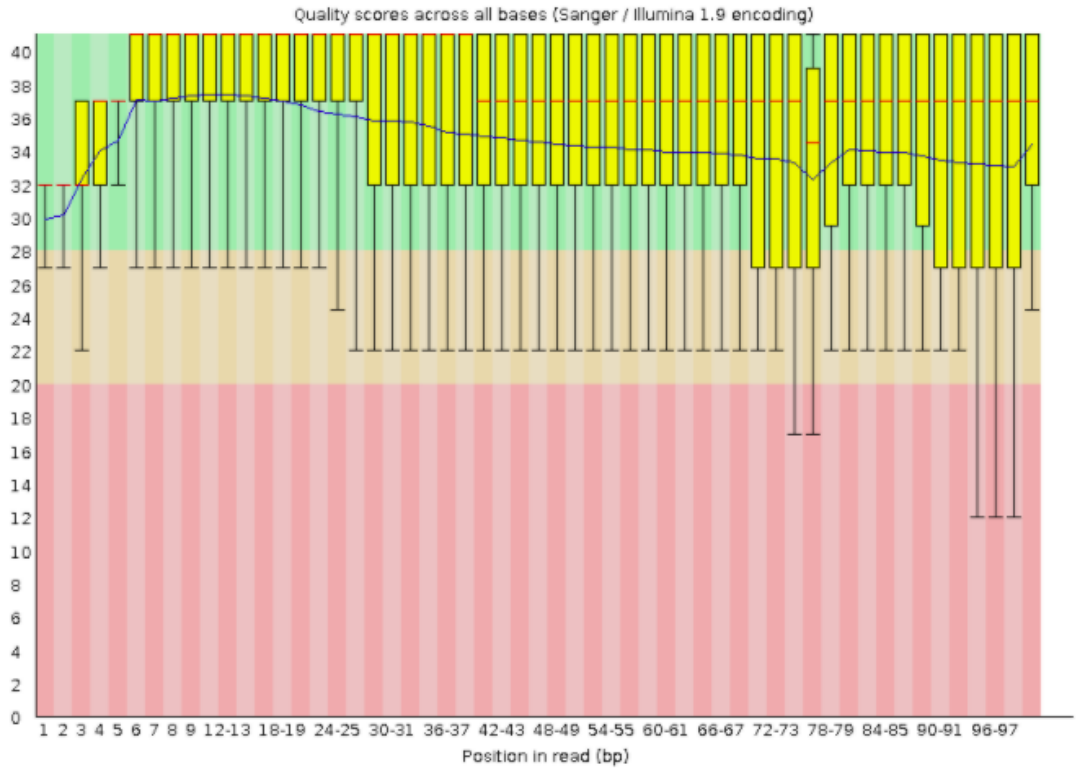


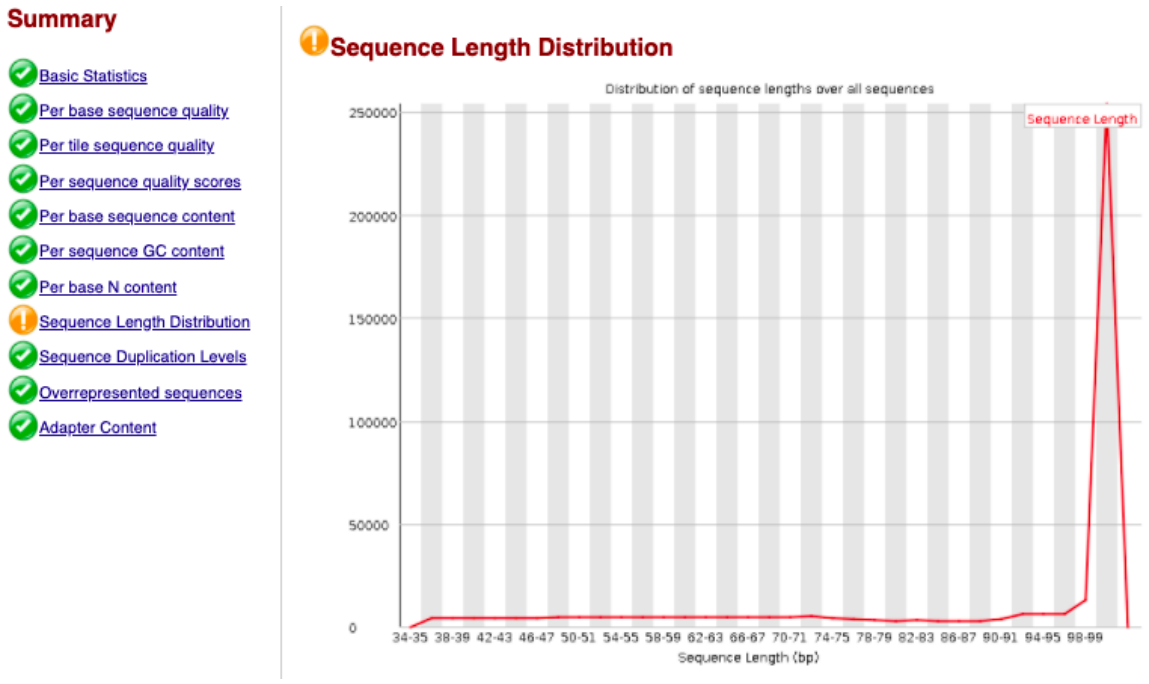
Per base sequence content

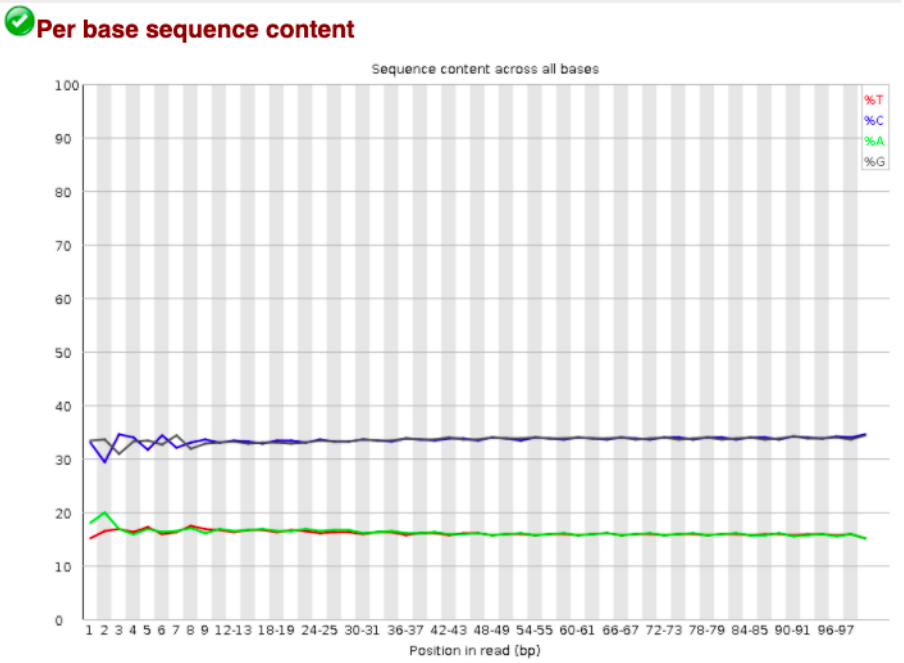


Lectura 2

Per base sequence quality







Como es apreciable, se mejoró considerablemente la condición de la segunda lectura, pero apareció la advertencia de que ambas sacrificaron su *Sequence Length Distribution*, pues de una constante en 100 bases, se redujo a 98-100 bases. Pero se consideró que este cambio es normal debido al trimeo de las lecturas, además, la advertencia no se convirtió en un *warning* demasiado peligroso, por lo que se decidió proseguir con estas lecturas depuradas con *Trimmomatic*.

5. Ensambles

Ensambles de Velvet

Velvet es un ensamblador de novo que trabaja con lecturas cortas para generar contigs o scaffolds. Este ensamblador se centra en 2 programas, velveth y velvetg. velveth permite leer los archivos de las secuencias y generar un diccionario de todas las palabras de longitud k. velvetg permite alinear las lecturas y crear una gráfica de 'de Brujin' con ellos, remover errores y resolver la gráfica [1].

Se usó VelvetOptimiser versión 2.2.6 para obtener los parámetros óptimos para el mejor ensamble posible, los cuales se utilizaron para velveth versión 1.2.10 y velvetg versión 1.2.10 para realizar ensambles de las lecturas cortas pareadas.

- **VelvetOptimiser**

Utilizamos VelvetOptimiser para realizar la búsqueda del mejor k-mero entre distintos rangos de tamaño, cambiando los valores del tamaño inicial y final a valores cercanos al valor óptimo recomendado por cada ejecución.

Los parámetros empleados fueron los siguientes:

Parámetro	Descripción
--s	Valor del hash inicial
--e	Valor del hash final
--x	Valor de los pasos de la búsqueda
-f	La sección de archivo de la línea de comando de velveth
-fastq	Formato de archivo fastq
-shortPaired	Lecturas cortas pareadas
-separate	Leer 2 archivos separados para lecturas pareadas

Para el parámetro -f se empleó la siguiente línea de comando:

```
fastq -shortPaired -separate ../CUP_GRP1_1_pair.fastq ../CUP_GRP1_2_pair.fastq
```

Realizamos cuatro ensambles con VelvetOptimiser en los que cuando se obtenía un tamaño de k-mero recomendado se hacía un nuevo ensamble pero reduciendo el rango y los pasos de búsqueda para ver si mejoraba el resultado, donde en el primero se hizo en el rango de 21 a 91, con pasos de búsqueda de 10; el segundo en el rango de 21 a 51, con pasos de búsqueda de 2; el tercero en el rango de 31 a 41, con pasos de búsqueda de 2; y el cuarto en el rango de 31 a 35, con pasos de búsqueda de 2.

De los cuatro ensambles se obtuvieron como mejores k-meros 31, 33, 35, y 33, respectivamente. Comparando los detalles de los ensambles, se obtuvo que el mejor tamaño de los k-meros obtenidos era el k-mero de 33.

Comando del mejor ensamble

```
VelvetOptimiser.pl --s 31 --e 35 --x 2 -f '-fastq -shortPaired -separate
../CUP_GRP1_1_pair.fastq ../CUP_GRP1_2_pair.fastq'
```

- **Velveth y Velvetg**

Dado los valores de k-meros obtenidos anteriormente, hicimos un ensamble para cada uno de ellos con velveth y velvetg, y los comparamos (junto con el mejor ensamble de VelvetOptimiser) mediante un quast para obtener el tamaño de k-mero que diera mejores resultados.

Para velveth utilizamos los siguientes parámetros:

Parámetro	Descripción
Directory	Directorio de los archivos de salida
hash_length	Valor del tamaño de palabra, k
-fastq	Formato de archivo fastq
-shortPaired	Lecturas cortas pareadas
-separate	Leer 2 archivos separados para lecturas pareadas

Para velvetg utilizamos los siguientes parámetros:

Parámetro	Descripción
Directory	Directorio de trabajo
-ins_length	Distancia esperada entre los finales de dos lecturas pareadas
-cov_cutoff	Cobertura mínima de los contigs
-exp_cov	Cobertura esperada
-scaffolding	Indica si se realizará scaffolding

Al terminar los ensambles, se realizó la comparación mediante quast, y se obtuvo que el mejor tamaño de k-mero era de 31, dado que obtuvo el menor número de contigs, la menor L50 y L75, el mayor N50 y N75, el mayor tamaño total del genoma y del contig más grande, y no obtuvo Ns. Los resultados se muestran a continuación.

All statistics are based on contigs of size >= 500 bp, unless otherwise noted (e.g., "# contigs (>= 0 bp)" and "Total length (>= 0 bp)" include all contigs).

Assembly	VELVET31_contigs	VELVET33_contigs	VELVET35_contigs	auto_data_33_contigs
# contigs (>= 0 bp)	599	699	773	306
# contigs (>= 1000 bp)	406	490	561	125
# contigs (>= 5000 bp)	263	314	328	97
# contigs (>= 10000 bp)	171	185	185	77
# contigs (>= 25000 bp)	68	54	47	56
# contigs (>= 50000 bp)	16	9	6	35
Total length (>= 0 bp)	5532663	5525151	5513764	5564729
Total length (>= 1000 bp)	5483017	5461674	5440166	5529899
Total length (>= 5000 bp)	5094972	4995044	4830521	5468376
Total length (>= 10000 bp)	4437037	4061570	3793938	5323807
Total length (>= 25000 bp)	2774351	2051792	1675785	4954492
Total length (>= 50000 bp)	958477	512481	334781	4205478
# contigs	439	536	622	139
Largest contig	79017	73946	73948	358870
Total length	5506404	5495838	5484721	5540290
GC (%)	67.65	67.64	67.63	67.67
N50	25359	18597	15820	101530
N75	12241	9464	8452	55893
L50	68	87	101	17
L75	144	192	221	35
# N's per 100 kbp	0.00	0.00	0.00	418.75

Comandos del mejor ensamble

```
velveth VELVET31 31 -fastq -shortPaired -separate ../CUP_GRP1_1_pair.fastq ../CUP_GRP1_2_pair.fastq

velvetg VELVET31 -ins_length 250 -cov_cutoff auto -exp_cov auto -scaffolding no
```

Ensambles de SPAdes

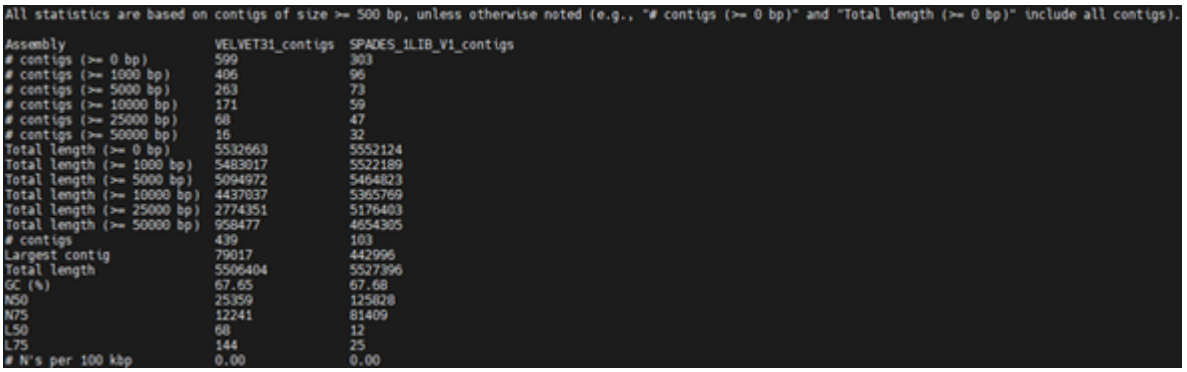
SPAdes es un conjunto de herramientas para ensamblar que permite llevar a cabo ensambles con lecturas cortas dada una lista de k-meros, donde generará una gráfica para cada k-mero y utilizando el conocimiento adquirido anteriormente de una gráfica resolverá las siguientes gráficas [2].

Se utilizó SPAdes genome assembler v3.15.3 para ensamblar nuevamente las lecturas cortas utilizando una lista de k-meros que fueran cercanos al mejor tamaño de k-mero obtenido con Velvet, y comparamos después mediante quast los resultados contra el ensamble de Velvet.

Para SPAdes utilizamos los siguientes parámetros:

Parámetro	Descripción
--careful	Intenta reducir el número de mismatches y deleciones cortas durante el ensamble
-1	Archivo con las lecturas forward pareadas
-2	Archivo con las lecturas reverse pareadas
-t	Número de threads a usar
-k	Lista de los tamaños de los k-meros a usar, separados por comas
-o	Directorio del output

Al comparar el ensamble de SPAdes contra el ensamble de Velvet, se obtuvieron mejores resultados con SPAdes.



All statistics are based on contigs of size >= 500 bp, unless otherwise noted (e.g., "# contigs (>= 0 bp)" and "Total length (>= 0 bp)" include all contigs).

Assembly	VELVET31_contigs	SPADES_1LIB_V1_contigs
# contigs (>= 0 bp)	599	303
# contigs (>= 1000 bp)	406	96
# contigs (>= 5000 bp)	263	73
# contigs (>= 10000 bp)	171	59
# contigs (>= 25000 bp)	68	47
# contigs (>= 50000 bp)	16	32
Total length (>= 0 bp)	5532663	5552124
Total length (>= 1000 bp)	5483017	5522189
Total length (>= 5000 bp)	5094972	5464823
Total length (>= 10000 bp)	4437037	5365769
Total length (>= 25000 bp)	2774351	5176403
Total length (>= 50000 bp)	958477	4654305
# contigs	439	103
Largest contig	79017	442996
Total length	5506404	5527396
GC (%)	67.65	67.68
N50	25359	125828
N75	12241	81409
L50	68	12
L75	144	25
# N's per 100 kbp	0.00	0.00

Comando de SPAdes

```
spades.py --careful -1 ../CUP_GRP1_1_pair.fastq -2 ../CUP_GRP1_2_pair.fastq -t 4 -k 21,31,33,35,41 -o SPADES_1LIB_V1
```

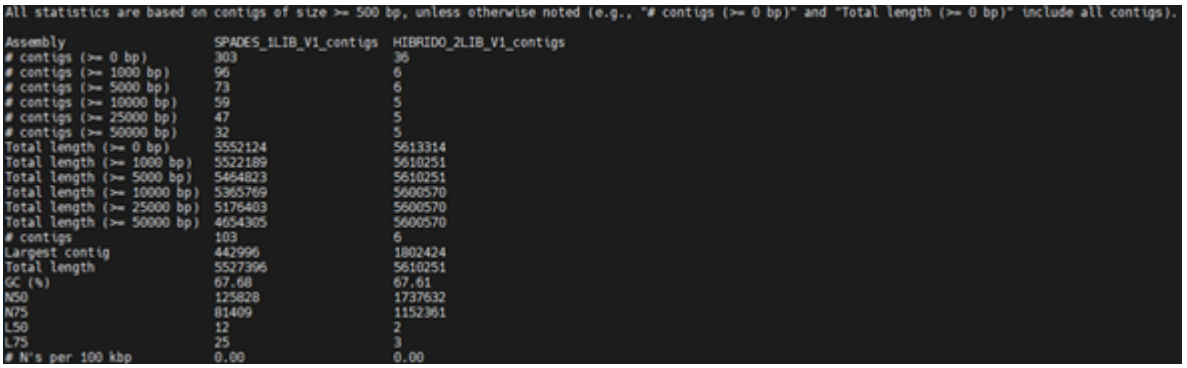
Ensamble híbrido de SPAdes

Dado que SPAdes permite realizar ensambles híbridos con lecturas cortas y lecturas largas [3], realizamos un ensamble híbrido usando nuevamente la lista de k-meros que fueran cercanos al mejor tamaño de k-mero obtenido con Velvet, y mediante un quast lo comparamos con los ensambles anteriores.

Los parámetros usados fueron los siguientes:

Parámetro	Descripción
--careful	Intenta reducir el número de mismatches y deleciones cortas durante el ensamble
--pacbio	Archivo de lecturas largas
-1	Archivo con las lecturas forward pareadas
-2	Archivo con las lecturas reverse pareadas
-t	Número de threads a usar
-k	Lista de los tamaños de los k-meros a usar, separados por comas
-o	Directorio del output

Al comparar el ensamble híbrido de SPAdes contra el ensamble anterior, se obtuvieron mejores resultados con el ensamble híbrido.



All statistics are based on contigs of size >= 500 bp, unless otherwise noted (e.g., "# contigs (>= 0 bp)" and "Total length (>= 0 bp)" include all contigs).

Assembly	SPADES_1LIB_V1_contigs	HIBRIDO_2LIB_V1_contigs
# contigs (>= 0 bp)	303	36
# contigs (>= 1000 bp)	96	6
# contigs (>= 5000 bp)	73	6
# contigs (>= 10000 bp)	59	5
# contigs (>= 25000 bp)	47	5
# contigs (>= 50000 bp)	32	5
Total length (>= 0 bp)	5552124	5613314
Total length (>= 1000 bp)	5522189	5610251
Total length (>= 5000 bp)	5464823	5610251
Total length (>= 10000 bp)	5365769	5600570
Total length (>= 25000 bp)	5176403	5600570
Total length (>= 50000 bp)	4654305	5600570
# contigs	103	6
Largest contig	442996	1802424
Total length	5527396	5610251
GC (%)	67.68	67.61
N50	125828	1737632
N75	81409	1152361
L50	12	2
L75	25	3
# N's per 100 kbp	0.00	0.00

Comando de SPAdes híbrido

```
spades.py --careful --pacbio ../../CUP_GRP1_canu.fastq  -1 ../CUP_GRP1_1_pair.fastq -2 ../CUP_GRP1_2_pair.fastq -t 4 -k 21,31,33,35,41 -o HIBRIDO_2LIB_V1
```

Ensamble de Canu

Una vez terminados los ensambles de lecturas cortas, se prosiguió a utilizar CANU v2.2 [7] para ensamblar las lecturas largas producto de secuenciación *PacBio* del archivo:

```
CUP_GRP1_canu.fastq
```

Dicho archivo contiene 56580 lecturas como se puede apreciar con el uso del comando *grep - c*:

```
(base) -bash-4.4$ ls
ANOTACION  CANU           CUP_GRP1_1_fastqc.html  CUP_GRP1_2.fq           CUP_GRP1_2_fastqc.zip
BOWTIE     CUP_GRP1_1.fq  CUP_GRP1_1_fastqc.zip   CUP_GRP1_2_fastqc.html  CUP_GRP1_canu.fastq
(base) -bash-4.4$ grep -c '@' CUP_GRP1_canu.fastq
56580
```

Para ensamblar dichas lecturas se creó el directorio "*CANU*" en el directorio principal de trabajo y se utilizó la siguiente línea de comando:

```
canu -p CANU_V2 -d CANU_V2 genomeSize=5500k -pacbio ../CUP_GRP1_canu.fastq
```

Se usó 5500 k como longitud estimada del genoma porque los ensambles de lecturas cortas daban una longitud muy similar (véanse el *quast* de *SPAdes* y el *quast híbrido*), por lo que se dedujo que era un buen parámetro. Nótese que se reporta el uso de la versión reciente de *canu* sin necesidad de ingresar al ambiente del servidor, pues la versión previa no dio resultados favorables. Terminado el ensamble, se obtuvo el siguiente directorio *CANU_V2*:

```
[(base) -bash-4.4$ ls CANU_V2/
CANU_V2.contigs.fasta              CANU_V2.seqStore.err            canu.out
CANU_V2.contigs.layout.readToTig   CANU_V2.seqStore.sh             correction
CANU_V2.contigs.layout.tigInfo     CANU_V2.trimmedReads.fasta.gz   trimming
CANU_V2.correctedReads.fasta.gz    CANU_V2.unassembled.fasta       unitigging
CANU_V2.report                     canu-logs
CANU_V2.seqStore                   canu-scripts
```

De donde se utilizó el resultado del ensamble directamente del archivo *CANU_V2.contigs.fasta* como un argumento del programa *quast* para comparar todos los ensamble realizados hasta el momento. Para ello se corrió *quast* de la siguiente manera dentro del directorio *CANU:*

```
quast CANU_V2/CANU_V2.contigs.fasta
../TRIMOMATIC/HIBRIDO/HIBRIDO_2LIB_V1/contigs.fasta
../TRIMOMATIC/SPADES/SPADES_1LIB_V1/contigs.fasta
```

Obteniéndose los siguientes resultados entre los tres mejores ensambles hasta el momento:

```
All statistics are based on contigs of size >= 500 bp, unless otherwise noted (e.g., "# contigs (>= 0 bp)" and "Total length (>= 0 bp)" include all contigs).
GC (%)                    67.59          67.61                  67.68
Assembly                  CANU_V2.contigs  HIBRIDO_2LIB_V1_contigs  SPADES_1LIB_V1_contigs
# contigs (>= 0 bp)       1              36                     303
# contigs (>= 1000 bp)    1              6                      96
# contigs (>= 5000 bp)    1              6                      73
# contigs (>= 10000 bp)   1              5                      59
# contigs (>= 25000 bp)   1              5                      47
# contigs (>= 50000 bp)   1              5                      32
Total length (>= 0 bp)    5641805        5613314                5552124
Total length (>= 1000 bp) 5641805        5610251                5522189
Total length (>= 5000 bp) 5641805        5610251                5464823
Total length (>= 10000 bp) 5641805       5600570                5365769
Total length (>= 25000 bp) 5641805       5600570                5176403
Total length (>= 50000 bp) 5641805       5600570                4654305
# contigs                 1              6                      103
Largest contig            5641805        1802424                442996
Total length              5641805        5610251                5527396
GC (%)                    67.59          67.61                  67.68
N50                       5641805        1737632                125828
N75                       5641805        1152361                81409
L50                       1              2                      12
L75                       1              3                      25
# N's per 100 kbp         0.00           0.00                   0.00
```

Como puede observarse, claramente el ensamble de *CANU* es la mejor opción, con el menor número de contigs, la mayor N50, la mayor N75, la menor L50 y la menor L75, pues solo ensambló un contig final. Hay que tener en consideración que dicho ensamble da como resultado un solo contig, cuando se esperaban dos, puesto que se sabía *a priori* que el genoma estaba constituido por dos cromosomas circulares. Empero, los resultados provistos por todos los ensambles mencionaban que ningún contig se circularizaba y que no se deducía un genoma circular. Por ello, se consideró que, a pesar de que el ensamble híbrido brindó 6 contigs, estos no se circularizarían, mientras que el ensamble de CANU dio un contig tampoco circularizado. En vista de que ambos ensambles fueron tomados en la misma circunstancia: lineal, se declara al de CANU como el mejor por las estadísticas mencionadas previamnte: número de contigs, L50, N50, L75 y N75.

No obstante, se sabe que *CANU* puede no tener lecturas de mucha confianza porque no pasa por un proceso de limpieza ni control de calidad *a priori*, como el caso de *SPAdes*. Debido a ello, se optó por analizar con la herramienta *BOWTIE* la validación de las lecturas largas de *CANU* con las lecturas cortas de los dos archivos originales.

6. **BOWTIE**

La versión utilizada fue *Bowtie 2 version 2.4.2 por Ben Langmead*.[8] El flujo de comandos de *bowtie2* fue el mismo para comparar tanto el ensamble híbrido como el de *CANU*.

CANU:

```
bowtie2-build --threads 4 ../../CANU/CANU_V2/CANU_V2.contigs.fasta CANUV2

bowtie2 --no-unal -p 4  -x DB/CANUV2 -1 ../TRIMOMATIC/CUP_GRP1_1_pair.fastq  -2
../TRIMOMATIC/CUP_GRP1_2_pair.fastq -S CANUV2.sam
```

```
413707 reads; of these:
  413707 (100.00%) were paired; of these:
    85885 (20.76%) aligned concordantly 0 times
    317875 (76.84%) aligned concordantly exactly 1 time
    9947 (2.40%) aligned concordantly >1 times
    ----
    85885 pairs aligned concordantly 0 times; of these:
      78398 (91.28%) aligned discordantly 1 time
    ----
    7487 pairs aligned 0 times concordantly or discordantly; of these:
      14974 mates make up the pairs; of these:
        3770 (25.18%) aligned 0 times
        4815 (32.16%) aligned exactly 1 time
        6389 (42.67%) aligned >1 times
99.54% overall alignment rate
```

Dando una tasa general de alineamiento del 99.54%

Híbrido:

```
bowtie2-build ../../TRIMOMATIC/HIBRIDO/HIBRIDO_2LIB_V1/contigs.fasta HIBRIDO

bowtie2 -p 4 --no-unal -x DB/HIBRIDO -1 ../TRIMOMATIC/CUP_GRP1_1_pair.fastq -2
../TRIMOMATIC/CUP_GRP1_2_pair.fastq -S HIBRIDOV1.sam
```

```
413707 reads; of these:
  413707 (100.00%) were paired; of these:
    86069 (20.80%) aligned concordantly 0 times
    320824 (77.55%) aligned concordantly exactly 1 time
    6814 (1.65%) aligned concordantly >1 times
    ----
    86069 pairs aligned concordantly 0 times; of these:
      79390 (92.24%) aligned discordantly 1 time
    ----
    6679 pairs aligned 0 times concordantly or discordantly; of these:
      13358 mates make up the pairs; of these:
        3726 (27.89%) aligned 0 times
        4939 (36.97%) aligned exactly 1 time
        4693 (35.13%) aligned >1 times
99.55% overall alignment rate
```

Dando una tasa general de alineamiento del 99.55%

El resultado de aplicar bowtie2 al ensamble de *CANU* dio una confianza de 99.54%, mientras que el híbrido dio 99.55%. Esto llevó a elegir el ensamble de *CANU* como el mejor ya que la diferencia en el número de contigs: 1 y 6, fue considerada significativa, mientras que los resultados brindados por *bowtie2* fueron prácticamente similares. Si bien, se esperaba que el ensamble final fuera de dos cromosomas y no uno, se considera que 1 contig de mayor longitud de dos cromosomas lo suficientemente semejantes y avalado en un >99% por lecturas cortas es mejor opción que el ensamble híbrido con 6 contigs. No obstante, se realizó un paso más para la última comparación de los dos mejores ensambles a través de la herramienta Artemis; para ello se crearon los correspondientes archivos BAM y SAM tanto de los *contigs* del ensamble híbrido como del de *CANU*. Se usó la siguiente estructura de comandos para ambos archivos:

CANU:

```
samtools view -b -S -o CANUV2.bam CANUV2.sam

samtools sort -o CANUV2.sorted.bam CANUV2.bam

samtools index -b CANUV2.sorted.bam
```

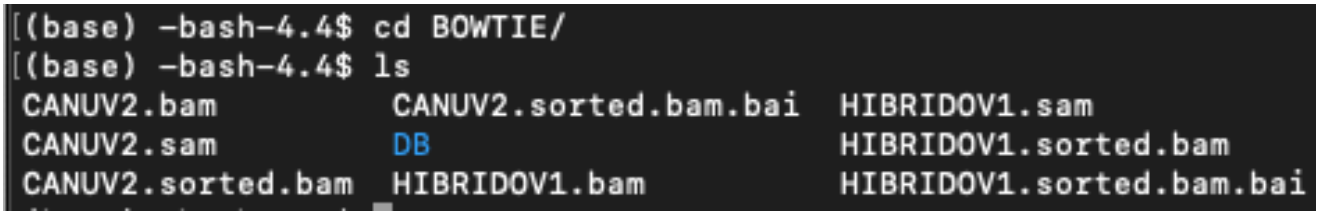
Híbrido:

```
samtools view -b -S -o HIBRIDOV1.bam HIBRIDOV1.sam

samtools sort -o HIBRIDOV1.sorted.bam HIBRIDOV1.bam

samtools index -b HIBRIDOV1.sorted.bam
```

Obteniéndose el siguiente directorio llamado “*BOWTIE*”, cuyos archivos *SAM* pueden ser analizados mediante *Artemis*:



7. **Artemis**

Artemis es un navegador genómico y una herramienta de anotación que permite visualizar las características de secuencia, datos de próxima generación y los resultados de análisis dentro del contexto de la secuencia, y también su traducción de seis marcos [4].

Se usó Artemis Release 18.1.0 para visualizar la cobertura de las lecturas cortas sobre el ensamble de CANU V2 y el ensamble híbrido de SPAdes.

Artemis CANU V2

Se obtuvo una buena cobertura sobre todo el ensamble con las lecturas cortas, excepto en los extremos, y no se observaron grandes desviaciones en la cobertura.

Artemis SPAdes híbrido

Se obtuvo una buena cobertura sobre todo el ensamble con las lecturas cortas, excepto en los extremos, pero se observaron zonas con mayor cobertura, lo que puede indicar que en esas zonas se encuentren las regiones repetidas del genoma colapsadas.

8. **Anotación**

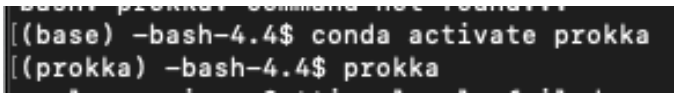
Una vez elegido el ensamble de *CANU* como el mejor ensamble, se prosiguió con la anotación funcional con el propósito de determinar los genes contenidos en el genoma secuenciado y posteriormente ensamblado. Se creó un directorio llamado "*ANOTACION*" en el directorio principal para trabajar, dentro otro directorio llamado "*GLIMMER*", en honor al software que se usa. Se utilizó primero el software *long-orfs* para extraer el número de genes, utilizando los parámetros *-l* para asumir genoma lineal y *-n* para excluir el header. Luego se corrió *extract* para leer el fasta con los *contigs* de *CANU* con la opción *-t* para excluir codones de paro, de esta manera se crea el archivo *run1.trai*n para entrenar al algoritmo en el siguiente paso en el que se usa *build-icm*. Se corrió *glimmer3* para hacer la predicción de genes con los parámetros *-l* asumiendo el cromosoma lineal *-o50* permitiendo el sobrelape default, *-g110* para una longitud mínima de gen, y *-t30* para poder de cómputo. Finalmente se volvió a correr *extract* con el archivo generado por *glimmer3*: *run1.predict*. Este flujo se expresa en las siguientes líneas de comando dentro del directorio "*ANOTACION/GLIMMER*":

```
long-orfs -l -n -t 1.15 ../../CANU/CANU_V2/CANU_V2.contigs.fasta run1.longorfs
extract -t ../../CANU/CANU_V2/CANU_V2.contigs.fasta run1.longorfs > run1.train
build-icm run1.icm < run1.train
glimmer3 -l -o50 -g110 -t30 ../../CANU/CANU_V2/CANU_V2.contigs.fasta run1.icm run1
extract -t ../../CANU/CANU_V2/CANU_V2.contigs.fasta
run1.predict>CANU_V2_genes.fasta
```

Los resultados del archivo final "*CANU_V2_genes.fasta*" se muestran a continuación (ignórese por el momento el directorio *PROKAA_CANUv2*):

Dando como resultado 2647 genes anotados:

Para llevar la anotación más allá, también se utilizó el programa *prokka 1.14.6* [9], con el objetivo de anotar a la bacteria, accediendo a través de un ambiente.



El uso de *prokka* se redujo a una línea que daba como archivo input el ensamble de *CANU* y el conocimiento previo de que el género de la bacteria es *Cupriavidus*.

```
prokka --outdir PROKKA_CANUv2 --prefix CANUv2 --locustag CANUv2 --genus Cupriavidus ../../CANU/CANU_V2/CANU_V2.contigs.fasta
```

Dando los siguientes resultados:

```
organism: Cupriavidus species strain
contigs: 1
bases: 5641808
CDS: 5024
rRNA: 12
tRNA: 65
tmRNA: 1
PROKKA_CANUv2/CANUv2.txt (END)
```

El archivo mostró un total de 5102 genes, poco menos que el doble del resultado anterior.

```
(prokka) -bash-4.4$ ls
CANUv2.err  CANUv2.ffn  CANUv2.fsa  CANUv2.gff  CANUv2.sqn  CANUv2.tsv
CANUv2.faa  CANUv2.fna  CANUv2.gbk  CANUv2.log  CANUv2.tbl  CANUv2.txt
(prokka) -bash-4.4$ grep ">" CANUv2.ffn | head -5
>CANUv2_00001 hypothetical protein
>CANUv2_00002 hypothetical protein
>CANUv2_00003 hypothetical protein
>CANUv2_00004 hypothetical protein
>CANUv2_00005 hypothetical protein
(prokka) -bash-4.4$ head CANUv2.ffn
>CANUv2_00001 hypothetical protein
ATGACAAGGGCCACGCTTGCACGAGCCGTCGCAACGCTAGTGGTGACCGCGAGCCTTGTT
GGCGCCTATTTCCTCGGCCATCGGCAGGGGCGCCACGGGGCACCATCGGCGCCGGCTGCA
CCAGCTGACGAGCCCAGTCAAAAGGCCGGCGATATCGACCCGAAGTCCGGCAAGCGCATT
CTCTATTGGCACGATCCGATGGTGCCAGGCCAGCGCTTTGACAAGCCCGGCAAATCGCCG
TTTATGGACATGCCGTTGGCCCCGGTATATGAGGAAGCAGGTGGCAGCGCGGCAGTGACC
ATCGACAGCCGCGTAGCGCAGAACCTCGGTATTCGAACCGCCGAGGTCAAGACCGGCCGC
CTGGAGTCGACCCTGCAAGTGCCAGGCAATGTCGCGATCAATGAGCGCGGGATCGAAGTG
ATCCAGGCCCGTGCGACCGGCTTTATCGAGCGCACATATGCCAGAACGACGCTGGACACC
GTGCGAAAGGGCCAGGCATTGGCGCAAATCTATGCGCCCGAATGGATAGCGGCCCAGGAA
(prokka) -bash-4.4$ grep -c ">" CANUv2.ffn
5102
```

9. **BLAST**

Se realizó un alineamiento de secuencias con *BLASTN 2.12.0+* de las cuatro secuencias de 16S rRNAs obtenidas contra la base de datos de refseq_rna para buscar la especie más cercana o la especie a la que le pertenecen esas secuencias. Se usaron secuencias de 16S rRNAs porque contienen tanto regiones altamente conservadas entre las distintas especies de bacterias como regiones muy variables.

De los resultados se obtuvo como mejor alineamiento a la misma secuencia parcial del 16S RNA ribosomal de **Cupriavidus metallidurans CH34**, que obtuvo los siguientes resultados de alineamiento para las 4 secuencias:

Description	Scientific Name	Common Name	Taxid	Max Score	Total Score	Query cover	E value	Per. Ident	Acc. Len	Accession
Cupriavidus metallidurans CH34 16S ribosomal RNA, partial sequence	Cupriavidus metallidurans CH34	NA	266264	2726	2726	99%	0.0	98.89%	1527	NR_074704.1
Cupriavidus metallidurans CH34 16S ribosomal RNA, partial sequence	Cupriavidus metallidurans CH34	NA	266264	2713	2713	99%	0.0	98.76%	1527	NR_074704.1
Cupriavidus metallidurans CH34 16S ribosomal RNA, partial sequence	Cupriavidus metallidurans CH34	NA	266264	2717	2717	99%	0.0	98.76%	1527	NR_074704.1
Cupriavidus metallidurans CH34 16S ribosomal RNA, partial sequence	Cupriavidus metallidurans CH34	NA	266264	2717	2717	99%	0.0	98.76%	1527	NR_074704.1

También hay que destacar que al hacer un blast con otras secuencias, estas se alinean con secuencias de otras especies de Cupriavidus, por lo cuál estos resultados no son concluyentes y requerirán de un análisis mucho más profundo para lograr determinar la especie.

10. **Conclusiones**

En conclusión, las herramientas de análisis bioinformático han representado una gran ventaja para el desarrollo de las ciencias genómicas, finalmente tanta información sería muy difícil de analizar si algo no nos sintetiza previamente la información obtenida, también nos son muy útiles para poder depurar la información obtenida, de igual manera se considera que los ensayos *in silico* permiten el ahorro de recursos, así como de tiempo ya que nos podemos centralizar en comprobar los ensayos más llamativos obtenidos a través del análisis y la síntesis computacional, finalmente la tecnología no ha hecho más que facilitar el trabajo, ahorrarnos tiempo y descubrir nuevos límites para la profundidad de nuestros estudios.

El procedimiento seguido es adecuado porque permite el uso fidedigno de los datos, su procesamiento, ensamble y posterior anotación. Es importante destacar que no existe un ensamble perfecto y sin fallas, pues siempre habrá lecturas de mejor o peor calidad, además de que los meros datos iniciales de secuenciación no están exentos de fallas. No obstante, un flujo de trabajo ordenado, que parte desde el análisis de calidad de los datos y su limpieza, que prueba con diferentes ensambladores y métodos de anotación y validación de resultados, tiene un gran potencial de brindar información valiosa y fiable. Se considera que el resultado de un contig es bueno, teniendo en cuenta que se usaron lecturas largas de secuenciación PacBio, y seguramente la razón por la que no se obtuvieron los dos contigs objetivo sea la información parcial, repetida o incompleta de los archivos originales, que no propiciaron la información necesaria para que los ensambladores circularizaran el genoma; pues la falta de circularización fue una constante a lo largo del uso de diversos ensambladores y parámetros.

11. **Referencias**

1. Zerbino D. R. (2010). Using the Velvet de novo assembler for short-read sequencing technologies. *Current protocols in bioinformatics*, *Chapter 11*, Unit–11.5. https://doi.org/10.1002/0471250953.bi1105s31
2. Prjibelski, A., Antipov, D., Meleshko, D., Lapidus, A., and Korobeynikov, A. (2020). Using SPAdes de novo assembler. *Curr. Protoc. Bioinformatics* 70:e102. doi: 10.1002/cpbi.102
3. Dmitry Antipov, Anton Korobeynikov, Jeffrey S. McLean, Pavel A. Pevzner, HYBRIDSPADES: an algorithm for hybrid assembly of short and long reads, *Bioinformatics*, Volume 32, Issue 7, 1 April 2016, Pages 1009–1015, https://doi.org/10.1093/bioinformatics/btv688
4. Tim Carver, Simon R. Harris, Matthew Berriman, Julian Parkhill, Jacqueline A. McQuillan, Artemis: an integrated platform for visualization and analysis of high-throughput sequence-based experimental data, *Bioinformatics*, Volume 28, Issue 4, 15 February 2012, Pages 464–469, https://doi.org/10.1093/bioinformatics/btr703
5. Vandamme, P; Coenye T. (Nov 2004). «Taxonomy of the genus Cupriavidus: a tale of lost and found». Int J Syst Evol Microbiol 54 (Pt 6): 2285-9. PMID 15545472. doi:10.1099/ijs.0.63247-0.
6. Pongsilp N, Nimnoi P, Lumyong S (2012) Genotypic diversity among rhizospheric bacteria of three legumes assessed by cultivation-dependent and cultivation-independent techniques.World J Microbiol Biotechnol 28(2):615–626. doi:10.1007/s11274-011-0855-7
7. Koren S, Walenz BP, Berlin K, Miller JR, Phillippy AM. Canu: scalable and accurate long-read assembly via adaptive k-mer weighting and repeat separation. Genome Research. (2017).
8. Langmead B, Salzberg S. Fast gapped-read alignment with Bowtie 2. *Nature Methods*. 2012, 9:357-359.
9. Seemann T. Prokka: rapid prokaryotic genome annotation. Bioinformatics. 2014;30: 2068 2069. doi:10.1093/bioinformatics/btu153 , https://www.ncbi.nlm.nih.gov/pubmed/24642063