Búsqueda de genes expresados diferencialmente en el tejido adiposo de *Cheirogaleus medius* durante la hibernación utilizando su genoma de referencia

Israel Blasco Ruano
Máster en Bioinformática y Bioestadística
Genómica comparativa y evolución

Nombre del consultor: José Luis Villanueva-Cañas
Nombre del PRA: Carles Ventura Royo
06/01/2020

Esta obra está sujeta a una licencia de Reconocimiento-NoComercial-SinObraDerivada 3.0 España de Creative Commons

FICHA DEL TRABAJO FINAL

Título del trabajo:	Genes expresados diferencialmente en el tejido adiposo de *Cheirogaleus medius* durante la hibernación utilizando su genoma de referencia
Nombre del autor:	Israel Blasco Ruano
Nombre del consultor:	José Luis Villanueva-Cañas
Nombre del PRA:	Carles Ventura Royo
Fecha de entrega:	06/01/2020
Titulación:	Máster en Bioinformática y Bioestadística
Área del trabajo final:	Genómica comparativa y evolución
Idioma del trabajo:	Castellano
Palabras clave:	Expresión génica diferencial, hibernación, lémur enano de cola gruesa.

Resumen del trabajo:

La hibernación es una de las estrategias más drásticas que emplean algunos mamíferos para enfrentarse a estaciones del año poco propicias. En la actualidad se cree que los cambios fisiológicos experimentados por estos mamíferos durante la hibernación están gobernados por variaciones en la expresión de ciertos genes, también presentes en mamíferos no hibernadores. Con el propósito de confirmar o desmentir esta teoría, es necesario investigar este fenómeno en especies filogenéticamente distantes en busca de patrones comunes.

Anteriormente se utilizó la secuenciación de ARN para investigar las variaciones en la expresión génica relacionadas con el letargo en un grupo de primates hibernadores formado por cuatro lémures enanos de cola gruesa de Madagascar (*Cheirogaleus medius*). El análisis bioinformático de los datos de la secuenciación de ARN se realizó en ausencia de un genoma de referencia. Se identificaron 90 genes con expresión diferencial cuando se compararon dos estados activos con el estado letargo.

Considerando que en la actualidad existen dos ensamblajes del genoma de esta especie, se han analizado los datos de la secuenciación de

ARN siguiendo una metodología diferente que ha permitido ampliar la lista de genes diferencialmente expresados con 29 nuevos genes. Estos se encuentran involucrados en procesos metabólicos, regulación biológica, transporte molecular, etc., necesarios para la supervivencia durante el letargo.

Abstract:

Hibernation is one of the most drastic strategies that some mammals use to face unfavorable seasons of the year. It is currently believed that the physiological changes experienced by these mammals during hibernation are ruled by variations in the expression of certain genes, also present in mammalian non-hibernators. In order to confirm or deny this theory, it is necessary to investigate this phenomenon in phylogenetically distant species in search of common patterns.

Previously, RNA sequencing was used to investigate variations in torpor-related gene expression in a group of hibernating primates consisting of four Madagascar thick-tailed dwarf lemurs (*Cheirogaleus medius*). Bioinformatic analysis of RNA sequencing data was performed in the absence of a reference genome. 90 genes with differential expression were identified when two active states were compared with the torpor state.

Considering that there are currently two genome assemblies of this species, RNA sequencing data has been analyzed following a different methodology that has allowed us to expand the list of differentially expressed genes with 29 new genes. These are involved in metabolic processes, biological regulation, molecular transport, etc., necessary for survival during torpor.

Índice

1 Introducción 1
1.1 Contexto y justificación del trabajo 1
1.2 Objetivos del trabajo 2
1.3 Enfoque y método seguido 3
1.4 Planificación del trabajo 5
1.4.1 Tareas 5
1.4.2 Calendario de planificación 6
1.5 Breve sumario de productos obtenidos 6
1.6 Breve descripción de los otros capítulos de la memoria 7
2 Resto de capítulos 8
2.1 Introducción 8
2.2 Materiales y métodos 13
2.2.1 Diseño experimental de la RNA-seq 13
2.2.1.1 Animales, régimen de vida y cuidados 13
2.2.1.2 Fechas de recolección de muestras experimentales 14
2.2.1.3 Extracción de muestras 15
2.2.1.4 Purificación del ARN y amplificación del transcriptoma completo 15
2.2.1.5 Preparación de la biblioteca y secuenciación con Illumina 16
2.2.2 Análisis bioinformático 17
2.2.2.1 Datos de la RNA-seq 18
2.2.2.2 Control de calidad 19
2.2.2.3 Filtración de lecturas de secuenciación 19
2.2.2.4 Transcriptoma de referencia 20
2.2.2.4.1 Alineamiento de las lecturas al genoma 21
2.2.2.4.2 Ensamblaje de los alineamientos 21
2.2.2.4.3 Proteínas homólogas en humanos 21
2.2.2.5 Estimación de la abundancia de los transcritos y genes diferencialmente expresados 22
2.2.2.5.1 Estimación de la abundancia de cada transcrito 22
2.2.2.5.2 Genes diferencialmente expresados 23
2.3 Resultados 23
2.3.1 Filtración de lecturas de secuenciación 23
2.3.2 Alineamiento de las lecturas 24
2.3.3 Transcritos ensamblados 28
2.3.4 Transcritos con proteínas homólogas en humanos 34
2.3.5 Identificación de genes diferencialmente expresados 35
2.4 Discusión 42
3 Conclusiones 46
3.1 Cumplimiento de los objetivos y seguimiento en la planificación 46
3.2 Perspectivas de futuro 48
4 Glosario y siglas 49
5 Bibliografía 52
6 Anexos 61

Lista de figuras

Figura 1. Calendario de planificación.
Figura 2. *Pipeline* bioinformático.
Figura 3. Porcentajes de alineamiento general a cada genoma.
Figura 4. Porcentajes de alineamientos apareados a cada genoma.
Figura 5. Porcentaje de alineamientos no apareados a cada genoma.
Figura 6. Curvas de densidad de distribuciones de la longitud de los transcritos ensamblados en cada individuo.
Figura 7. Curvas de densidad de distribuciones de la cobertura en cada individuo.
Figura 8. Curvas de densidad de distribuciones de los TPM en cada individuo.
Figura 9. Curvas de densidad de distribución de los FPKM en cada individuo.
Figura 10. Curva de densidad de distribución de la longitud de los transcritos fusionados.
Figura 11. Porcentaje de varianza explicado por cada componente principal.
Figura 12. Análisis de componentes principales.

Lista de tablas

Tabla 1. Información de los archivos SRA.
Tabla 2. Caracteres ASCII de calidad de secuenciación.
Tabla 3. Número de lecturas secuenciadas y de lecturas que superan la filtración con Trimmomatic.
Tabla 4. Resultados del alineamiento a los genomas ASM808673v1 y CheMed_v1_BIUU con Bowtie2.
Tabla 5. Número de transcritos ensamblados.
Tabla 6. Número de transcritos en función del porcentaje de la longitud de cobertura de su proteína homóloga humana.
Tabla 7. Número de genes diferencialmente expresados.
Tabla 8. Resultados del análisis de enriquecimiento funcional de los genes expresados en menor proporción durante el letargo.

1. Introducción

1.1 Contexto y justificación del trabajo

La hibernación es una de las estrategias que emplean algunos mamíferos, entre otros animales, para enfrentarse a estaciones del año poco propicias. La reducción de la temperatura corporal (Tc), del gasto de energía y de la pérdida de agua son algunos de los cambios fisiológicos más representativos que se experimentan en este estado [1]. En la actualidad, se cree que estos cambios están gobernados por variaciones en la expresión de ciertos genes, genes que también están presentes en mamíferos que no hibernan [2].

La herramienta actualmente más utilizada para realizar estudios de expresión génica es la secuenciación de ARN (RNA-seq). Esta tecnología nos permite obtener, a partir de una muestra biológica, las lecturas de las RNA-seq fragmentadas y sin orden. Para deducir de estas lecturas la abundancia de cada transcrito, es necesario alinearlas contra una referencia, que puede ser el genoma, transcriptoma, exoma, etc. En algunas especies no existe esa referencia y se debe utilizar un transcriptoma ensamblaje *de novo*, que consiste en crear una referencia a partir del solapamiento de las lecturas que hemos obtenido de la RNA-seq [3].

Entre los mamíferos se conocen catorce órdenes en los que hay algún miembro que hiberna [4]. Siendo una estrategia tan extendida, es posible que el mecanismo molecular sea el mismo en todos ellos. Una comparación de perfiles de expresión entre diferentes especies podría revelar cuáles son los genes más determinantes en el proceso de hibernación.

La primera aportación de un perfil de expresión génica en primates hibernadores es relativamente reciente [5]. Para la realización de este perfil se recogieron muestras de tejido adiposo blanco (WAT) en un grupo de lémures enanos de cola gruesa (*Cheirogaleus medius*), durante la hibernación y durante el periodo de actividad. El ARN extraído de estas muestras fue secuenciado y posteriormente analizado en ausencia de un genoma de referencia.

El lémur enano de cola gruesa malgache, *Cheirogaleus medius,* es uno de los pocos primates que manifiestan la hibernación. Hiberna durante siete meses al año, a pesar de que las temperaturas invernales superan los 30 °C. El lugar de origen del lémur

enano, Madagascar, alberga una de las faunas más inusuales, endémicas, diversas y amenazadas del mundo. La biodiversidad que alberga esta isla es de gran interés debido a su peculiar evolución, ya que ha permanecido aislada del resto del mundo durante millones de años [6]. Conocer mejor a este animal podría incrementar el interés y la concienciación de la población en su conservación. Además, es el mamífero hibernador más cercano evolutivamente al ser humano del que se han obtenido datos. Esta cercanía evolutiva podría abrir la puerta a la aplicación de estos conocimientos en el campo de la biomedicina. Actualmente ya se han reportado genes involucrados en el metabolismo de lípidos, la regulación del apetito y los ritmos biológicos. En el futuro, entender los mecanismos que conducen a la heterotermia controlada podría tener aplicaciones en cirugía [7], la regeneración de conexiones neurales perdidas durante la hibernación podría ser extrapolada a nuestra especie [8-10], la baja atrofia muscular durante la hibernación podría ser una solución en los casos de tratamientos que requieran de larga estancia hospitalaria [1] y quién sabe si la hibernación inducida algún día permitirá los viajes espaciales [11].

Durante este Trabajo de Fin de Máster (TFM) analizaremos de nuevo los datos de la RNA-seq anteriormente mencionada. La actual disponibilidad de la secuencia del genoma de referencia de *Cheirogaleus medius* nos permite practicar una metodología diferente que esperamos nos permita extraer una mayor cantidad de conocimiento, con menor cantidad de errores y todo ello con una inversión económica prácticamente nula.

1.2 Objetivos del trabajo

- Determinar la variación de la expresión génica durante la hibernación en las células del WAT del lémur enano de cola gruesa de Madagascar.
  - Desarrollar un *pipeline* bioinformático que implemente el análisis de datos de la RNA-seq usando un genoma como referencia.
  - Cuantificar la expresión génica en cada una de las muestras.
  - Compilar una lista con los genes diferencialmente expresados entre las dos condiciones.
- Valorar si existe relación entre los genes diferencialmente expresados en el lémur y los diferencialmente expresados en otras especies durante la hibernación.
  - Seleccionar las especies para la relación.

- Identificar los genes diferencialmente expresados en el lémur que tiene un comportamiento parecido en las otras especies durante la hibernación.

- Adquirir los conocimientos para el procesamiento de datos de una RNA-seq.

- Analizar las herramientas existentes para cada uno de los pasos del procesamiento de datos.

- Seleccionar las herramientas más convenientes para nuestro caso particular.

- Probar las herramientas seleccionadas.

- Establecer un orden en el procesamiento de los datos.

- Procesar los datos con las herramientas seleccionadas en el orden establecido.

- Interpretar los resultados obtenidos.

- Identificar posibles errores cometidos durante el procesamiento y subsanarlos.

1.3 Enfoque y método seguido

Durante este proyecto analizaremos los datos crudos de la RNA-seq que provienen de un proyecto anterior [5]. Por lo tanto, el diseño experimental viene predeterminado.

Los animales utilizados para este proyecto son cuatro lémures enanos de cola gruesa, dos machos y dos hembras entre los 8 y los 13 años de edad, nacidos y criados en cautividad. Se establecieron tres momentos de recogida de muestras, uno durante el periodo de hibernación y dos fuera de este periodo. La segunda muestra recogida durante el periodo de hiperfagia difiere de la primera en el paulatino descenso de las horas de luz a las que se vieron expuestos los individuos.

Durante el periodo que precede a la hibernación, los lémures acumulan grasa en el WAT que luego utilizan durante el periodo de letargo como reservorio de energía. Como este órgano es esencial durante la hibernación y se puede obtener una muestra con un perjuicio mínimo para el animal, fue el tejido elegido para la obtención del ARN. Esta muestra biológica fue recogida de la cola, que es donde la grasa se encuentra más concentrada.

Para evitar sesgos, la extracción de ARN de las muestras biológicas fueron realizadas el mismo día y siguiendo el mismo protocolo. El bajo rendimiento de la extracción de ARN debido al alto contenido lipídico de este tipo de muestras hizo necesaria una amplificación del transcriptoma completo a ADN complementario (ADNc) y depleción de ARN ribosómico (ARNr) sin pre-selección de ARN mensajero (ARNm).

Finalmente, la biblioteca para la secuenciación fue preparada con una distribución del tamaño de los insertos entre 206 y 256 pares de bases.

Illumina fue la tecnología utilizada para la secuenciación y se seleccionó el modo «rapid-run» con lecturas de extremos apareados de 150 pares de bases. Se juntaron las bibliotecas y se secuenciaron en dos carriles del secuenciador para evitar sesgos.

Hasta aquí, el diseño experimental para la obtención de los datos que utilizaremos.

Durante el análisis de los datos será necesario ir realizando algunos controles de calidad. El primero de estos controles de calidad se realizará sobre los datos crudos obtenidos de la secuenciación. De entre las herramientas que existen para este fin, FastQC es una de las más populares para aplicar sobre lecturas de Illumina [12]. FastQC se combina con el software Trimmomatic que descarta lecturas de baja calidad, recorta secuencias de los adaptadores y elimina bases de baja calidad para mejorar el mapeo [13].

Una vez preparados los datos, teniendo en cuenta la calidad de los mismos, podemos mapear las lecturas contra su genoma o contra el transcriptoma anotado [3]. Como nuestro proyecto está enfocado en la cuantificación, utilizaremos el transcriptoma como referencia.

La estimación de la abundancia de cada transcrito será el siguiente paso. Esta estimación está principalmente basada en el número de lecturas que son mapeadas a cada transcrito. Algoritmos como RSEM [14], eXpress [15], Sailfish [16] y Kallisto [17] podrían realizar esta labor, pero Kallisto es la más conveniente porque nos permite trabajar en un ordenador portátil sin consumir un tiempo excesivo, con una gran exactitud. Además, el análisis de expresión diferencial de los transcritos lo llevaremos a cabo con Sleuth, que es un paquete complementario para Kallisto.

En este momento estaremos preparados para un nuevo control de calidad. Se comprobará el contenido de CG y el sesgo de longitud de gen, por si es necesaria una normalización. También analizaremos la composición del biotipo de la muestra que es indicativo de la calidad de la purificación de ARN [3].

Finalmente, caracterizaremos las vías o las funciones moleculares de los genes diferencialmente expresados e identificaremos genes diferencialmente expresados en el lémur enano, también diferencialmente expresados en otras especies durante la hibernación. Para el primer objetivo, Bioconductor proporciona algunos paquetes de R que contienen recursos de anotación [18] y para el segundo, compararemos nuestra lista

de genes diferencialmente expresados con las lista publicadas de otras especies y representaremos en un diagrama de Venn los genes comunes y su mayor o menor abundancia durante la hibernación. De la misma manera, compararemos nuestros resultados con los obtenidos en la publicación original [5].

Como control de calidad global se realizara un análisis de componentes principales con las diferencias de expresión de los genes de cada muestra. De este gráfico esperaremos que las réplicas biológicas de las mismas condiciones se agrupen juntas [3].

1.4 Planificación del trabajo

1.4.1 Tareas

- Instalación y prueba de herramientas.
- Control de calidad de los datos crudos.
- Generación de un transcriptoma de referencia.
- Cuantificación de los transcritos expresados.
- Análisis de expresión diferencial de genes entre condiciones.
- Control de calidad global.
- Perfil funcional de los genes diferencialmente expresados.
- Identificación de genes que también están diferencialmente expresados en otras especies.

1.4.2 Calendario de planificación

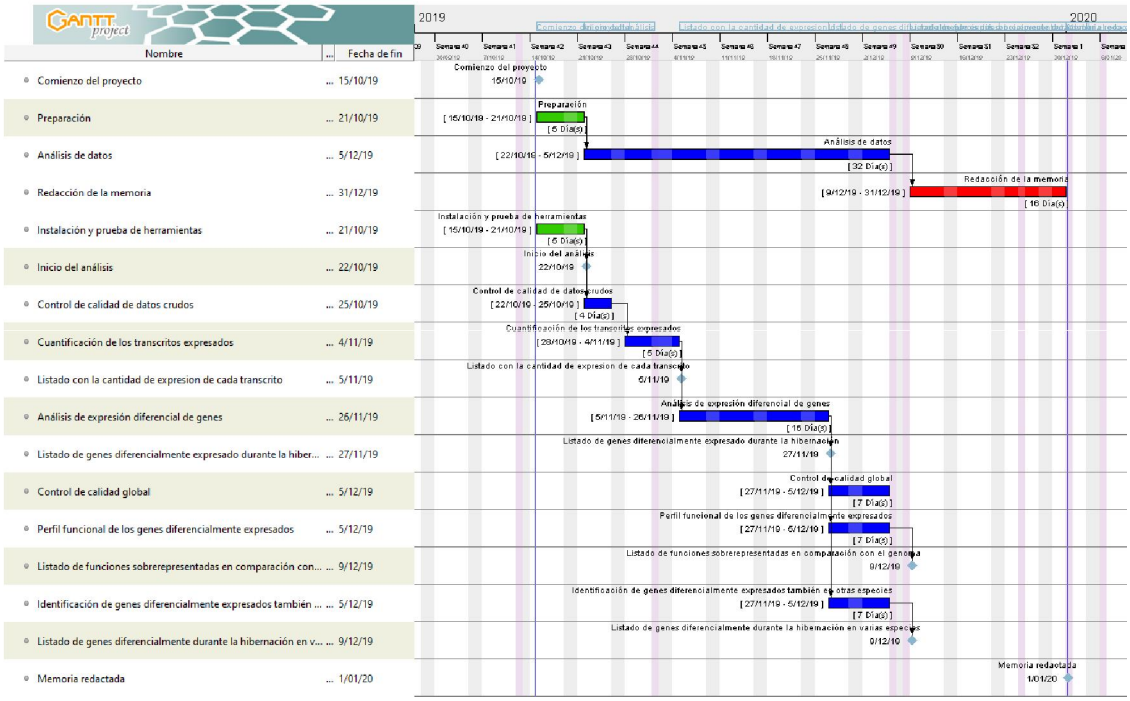

Figura 1. Calendario de planificación.

1.5 Breve sumario de productos obtenidos

Durante el desarrollo del TFM hemos creado los siguientes productos:

- Flujo de trabajo: un documento R Markdown que explica paso a paso el análisis de los datos de la RNA-seq, incluyendo el código r y bash utilizado. Este código puede ser ejecutado de principio a fin desde RStudio.
- Cuatro hojas de cálculo que incluyen las listas de los genes diferencialmente expresados encontrados en el análisis de datos de la RNA-seq.

- sleuth_significant_A1_Tp_wt: resultados obtenidos en la comparación de la expresión de los genes entre el estado activo 1 y el estado letargo.
- sleuth_significant_A2_Tp_wt: resultados obtenidos en la comparación de la expresión de los genes entre el estado activo 2 y el estado letargo.
- sleuth_significant_A1_A2_wt: resultados obtenidos en la comparación de la expresión de los genes entre el estado activo 1 y el estado activo 2.
- sleuth_significant_AA_Tp_wt: resultados obtenidos en la comparación de la expresión de los genes entre el estado activo (considerando el estado activo 1 y el estado activo 2 como réplicas del mismo estado) y el estado letargo.

1.6 Breve descripción de los otros capítulos de la memoria

El capítulo dos consta de los apartados siguientes:

- Introducción: contextualización en profundidad de los conceptos necesarios sobre la hibernación y la tecnología disponible para su estudio a nivel molecular.
- Materiales y métodos: información relacionada con los datos analizados y las propiedades de las herramientas utilizadas para el análisis.
- Resultados: descripción pormenorizada de los datos obtenidos en cada una de las etapas del análisis.
- Discusión: breve interpretación de los resultados obtenidos.

2. Resto de capítulos

2.1 Introducción

Tras millones de años de evolución, los seres vivos endotermos han establecido su Tc entre los 32 y los 42 ºC [1, 19]. Este rango de Tc es óptimo para facilitar una amplia variedad de reacciones metabólicas independientemente de la temperatura ambiental (Ta). Para minimizar la pérdida de calor y mantener constante su Tc, normalmente los endotermos aíslan su cuerpo y equilibran el calor perdido produciendo un calor interno proporcional [1]. La pérdida de calor se produce en función de la diferencia entre la Tc y la Ta. El coste termorregulatorio puede llegar a ser prohibitivo, especialmente a bajas temperaturas, para animales de menor tamaño y en situaciones de escasez de alimento [1].

Para afrontar estos desafíos energéticos, no todos los endotermos permanecen homeotérmicos, sino que entran en un estado de letargo que reduce controlada, sustancial y reversiblemente la Tc, la tasa metabólica, el consumo de agua, la frecuencia cardíaca y la respiratoria [1, 2, 20-22].

El letargo en las especies heterotérmicas puede ocurrir en momentos puntuales del día (letargo diario) o durante varios días en función de la estación del año (hibernación) [1]. Para sobrevivir en ambientes con grandes cambios estacionales, muchas especies de mamíferos exhiben una plasticidad fenotípica y etológica importante. La hibernación es una de sus estrategias adaptativas más extremas [2, 23-26].

La hibernación clásica en mamíferos consiste en largos periodos de letargo, interrumpidos regularmente por breves lapsos de normotemperatura conocidos como *interbouts arousals* (IBA) [1, 19]. La Tc y la periodicidad de los IBA puede variar dependiendo de la especie desde -3 ºC intercalados con IBA cada 3-4 semanas en la ardilla de tierra ártica hasta ~23 ºC sin IBA durante más de cuatro meses en tenrecs, o tener una Tc pasivamente fluctuante como el lémur enano de cola gruesa [1].

Se desconoce cuál es la función de los IBA. Se especula que los periodos de recalentamiento son necesarios para la reanudación de la transcripción [27] y la traducción [26] y la activación del sistema inmune [28]. También para proporcionar un tiempo de reemplazo, reposición y reparación molecular [29, 30], en el que se produce una eliminación de desechos metabólicos potencialmente tóxicos acumulados en la sangre durante el letargo [1].

En el pasado, la hibernación y el letargo diario se interpretaron como una respuesta adaptativa al frío pero, en la actualidad, se sabe que ocurren en muchos hábitats, desde el ártico hasta los trópicos. Parece que la escasez de alimentos debida a las estaciones secas en los trópicos [23, 24, 31] y las bajas temperaturas en el ártico son las dos principales razones que motivan la hibernación [1, 22, 32]. En algunos casos surge como respuesta predictiva a futuras carencias, como es el caso de la mulgara preñada que experimenta letargo diario para tener reservas durante la lactancia [1].

Los hibernadores conocidos se encuentran en cinco órdenes de mamíferos uterinos, así como en algunas especies de marsupiales y monotremas [2, 33]. Estas especies que hibernan están frecuentemente emparentadas con especies que no hibernan [2]. La significativa extensión de la hibernación, junto con la intercalada distribución filogenética de especies hibernadoras entre mamíferos, hace de ésta un sistema modelo ideal para probar la siguiente hipótesis: la base molecular de importantes eventos adaptativos durante la evolución involucra mecanismos dirigidos por regulación molecular de genes existentes y no la invención de genes nuevos [34].

El origen evolutivo de la hibernación es desconocido, cualquiera de las dos siguientes hipótesis puede explicar la naturaleza intercalada de la distribución de las especies hibernadoras existentes en los mamíferos. La primera afirma que el ancestro común de todos los mamíferos fue un hibernador, pero la habilidad de hibernar se perdió independientemente en los diferentes linajes [2]. La segunda postula que el ancestro de todos los mamíferos fue un no hibernador y la habilidad de hibernar fue adquirida independientemente en cada linaje [2]. En cualquiera de estos escenarios, actualmente se considera que un número pequeño de cambios en la regulación determina el fenotipo de hibernación, en lugar de la creación *de novo* o la pérdida de numerosas funciones bioquímicas específicas de la hibernación, codificadas por una colección de genes específicos de la hibernación [2]. Así, a la luz de estos datos, podemos suponer que la hibernación debe ser el resultado de una reprogramación de capacidades bioquímicas existentes en los mamíferos a través de la expresión diferencial de genes existentes [2, 34].

Los lémures de Madagascar comprenden un clado que contiene aproximadamente el 20% de todas las especies de primates e ilustra el impacto negativo de la actividad antropogénica en primates no humanos [35, 36], >90% de especies de lémur están actualmente en peligro de extinción según la evaluación de la Lista Roja de la Unión Internacional para la Conservación de la Naturaleza (IUCN).

El lémur enano (del género *Cheirogaleus*) pertenece a uno de los dos clados en el linaje primate que manifiesta tasas metabólicas deprimidas a través del letargo [5]. El lémur ratón, miembro de la familia Cheirogaleidae, es capaz de letargo, aunque el periodo de letargo en este animal es inferior a las 24 horas [5, 37, 38]. El otro clado de primates que expresa letargo son los Lorisiformes. Recientemente ha sido documentado que el loris lento pigmeo asiático muestra periodos de supresión metabólica sostenida en condiciones de semicautividad [39], mientras que referencias anecdóticas de letargo han sido también descritas en su linaje hermano, el galago, bajo demandas energéticas extremas [40].

Los lémures enanos son primates nocturnos de cuerpo pequeño (150-500 g) que experimentan hibernación obligatoria durante el invierno austral. Sin embargo, a diferencia de las especies que hibernan para adaptarse al frío, las especies de lémur enano viven en regiones subtropicales de Madagascar por lo que durante el letargo experimentan temperaturas corporales comparativamente más elevadas (~12-15 ºC) [23, 41-43].

En años recientes, este género ha experimentado revisiones taxonómicas sustanciales, pero cuatro especies se han mantenido generalmente bien resueltas: *Cheirogaleus sibreei*, *C. medius*, *C. crossleyi* y *C. major* [36]. Estas especies se distribuyen independientemente del hábitat, desde el altiplano hasta baja altitud, desde el árido oeste hasta el tropical este [44] y comparten el hábitat con más de una especie de lémur enano [45]. Todas las poblaciones experimentan hibernación estacional a pesar de habitar entornos climáticos drásticamente diferentes, donde algunas especies hibernan hasta siete meses al año [36]. Se sospecha que la hibernación es un fenotipo ancestral en el clado, aunque con variación posterior en su manifestación [23]. Por ejemplo, *Cheirogaleus sibreei* ocupa ambientes fríos a elevada altitud e hiberna hasta siete meses al año [46], mientras que las especies hermanas *Cheirogaleus major* y *Cheirogaleus crossleyi* ocupan hábitats a menor altitud e hibernan durante periodos inferiores, aproximadamente cuatro meses [46]. *Cheirogaleus medius* muestra un fenotipo de hibernación similar al de *Cheirogaleus sibreei*, hiberna hasta siete meses al año en los bosques caducifolios del oeste, donde el invierno es una estación de sequía severa asociada a una escasez de recursos [36].

El lémur enano de cola gruesa, *Cheirogaleus medius*, cuando ocupa un hibernáculo bien aislado, experimenta unos patrones de hibernación clásicos que incluyen los IBA característicos [5, 41]. Sin embargo, cuando hiberna en madrigueras arbóreas pobremente aisladas puede abandonar una defensa activa de su Tc y cambiar a un estado poiquilotérmico donde la Tc se adapta a la Ta sin la presencia de los IBA. Este

comportamiento contrasta con las otras especies que hibernan, excepto miembros de la familia Ursidae y Tenrecidae [47, 48]. Seguramente, la habilidad de adaptarse a la Ta en condiciones de aislamiento precario permite a este primate tropical entrar en hibernación, incluso a temperaturas fluctuantes (oscilaciones de más de 30 ºC en un solo día) y elevadas [5, 32, 49].

Durante el verano muchos mamíferos que hibernan experimentan hiperfagia y almacenan grasa en el WAT antes del comienzo de la hibernación [19]. El lémur enano puede superar el doble de su peso durante esta acumulación de reservas que utilizará como energía durante el largo periodo de inercia física [5, 50]. En cautividad, incluso en ausencia de detonantes ambientales como la escasez de comida y las bajas temperaturas, el lémur enano de cola gruesa exhibe ciclos de masa corporal estacionales [5]. Esta respuesta estacional sugiere la existencia de una especie de reloj molecular responsable de dicho comportamiento [5, 51, 52].

La hibernación puede ser vista como una serie de adaptaciones en varios tejidos y órganos que confieren al lémur enano la capacidad de sobrevivir a estaciones climáticas extremas [19, 49]. La suma de estas adaptaciones resulta en un alejamiento radical de la homeostasis fisiológica normal [19, 53, 54].

El WAT es un órgano especialmente informativo en el contexto del fenotipo de la hibernación dado que los depósitos de grasa y el metabolismo son aspectos cruciales de la supervivencia durante la hibernación. [5] Los hibernadores que almacenan grasa durante el verano cambian sus economías metabólicas de carbohidratos durante la estación activa a lípidos durante la estación de hibernación como un medio para conservar la energía [21, 24, 55].

Las investigaciones de cómo los ciclos de masa corporal anual son gobernados por mecanismos moleculares subyacentes al WAT han sido realizadas en múltiples linajes mamíferos usando acercamientos de genes candidatos [55-63] y, recientemente, usando secuenciación de última generación [64] en un subconjunto de modelos mamíferos que hibernan. Estos estudios constituyen las bases del paradigma de que la expresión diferencial de genes se correlaciona con los cambios fisiológicos a lo largo del año [5]. Aunque un estudio previo empleó un acercamiento longitudinal a la investigación de niveles de proteína en el suero sanguíneo del oso negro [65]. Este fue el primero en examinar los niveles de expresión de genes en tejido biológico del mismo animal en múltiples momentos del año [5]. Se constituye así un poderoso enfoque para el control de la variación interindividual que podría confundir el análisis de expresión génica diferencial [5].

Actualmente, se desconoce de qué modo los ciclos de masa corporal estacionales son regulados en el lémur enano, y si este se ajusta al paradigma convencional que afirma que los cambios en su economía metabólica ocurren en conjunción con variaciones en la expresión génica durante el transcurso del ciclo circanual [5]. Por eso, en el estudio de Faherty et al. [5], se usaron datos de expresión génica generada por un experimento de la RNA-seq, muestreados longitudinalmente en múltiples animales para examinar los controles genéticos detrás del cambio metabólico que conduce de un metabolismo basado en carbohidratos a uno basado en lípidos en el lémur enano. Se compararon sus resultados con datos publicados previamente de especies de mamíferos modelos para proporcionar una visión más amplia en la pregunta de si el fenotipo hibernación en mamíferos de parentesco distante es controlado por la misma vía reguladora [5].

La información transcriptómica es generalmente abundante en organismos modelos en los cuales la comunidad científica internacional ha concentrado el esfuerzo de investigación y los fondos, dejando de lado organismos que no son modelos [66]. Esta situación está cambiando drásticamente con la aparición y la generalización de tecnologías de secuenciación de ácido desoxirribonucleico (ADN) de *Next-Generation Sequencing* (NGS) que reducen enormemente el coste, la laboriosidad y el tiempo, proporcionando la oportunidad de realizar proyectos genómicos a gran escala y a menor coste para organismos que no son modelo [66].

Las tecnologías más económicas de NGS son aquellas que generan lecturas de secuencias cortas, y son el método de elección para la resecuenciación en organismos modelo (tecnología Illumina) [67]. En este caso el análisis es realizado mapeando las lecturas cortas sobre el genoma o transcriptoma de referencia. Este acercamiento ha sido recientemente utilizado para los perfiles de transcriptoma en un método llamado RNA-seq que se espera permita mayor rendimiento en el análisis del transcriptoma [68].

A pesar del rápido desarrollo de ensambladores capaces de manejar eficientemente cada vez más lecturas [69, 70], el ensamblaje *de novo* del transcriptoma sin una referencia conocida usando lecturas cortas es todavía difícil, aunque de gran utilidad [66]. La exactitud de la predicción de los transcritos y la cantidad de ellos dependen de los métodos utilizados, pero en cualquier caso es siempre inferior a los obtenidos con la ayuda del genoma de referencia.

Cuando se realizó el estudio del perfil de expresión génica en el lémur enano [5], todavía no se había secuenciado su genoma, por eso, los resultados que se obtuvieron están basados en un ensamblaje *de novo* del transcriptoma [5]. En este estudio aprovecharemos la reciente disponibilidad de la secuencia de este genoma [36] para

volver a analizar los datos de la RNA-seq. Se espera que el transcriptoma obtenido con ayuda del genoma de referencia sea de mayor tamaño y contenga un menor número de errores. Compararemos los rendimientos de estos dos métodos de trabajo. Intentaremos relacionar los cambios de expresión génica durante la hibernación con los cambios metabólicos y contrastaremos estos cambios con los de otras especies en busca de similitudes propias de un mecanismo molecular universal en la hibernación.

2.2 Materiales y métodos

2.2.1 Diseño experimental de la RNA-seq

En este apartado, contextualizaremos la situación de los animales en el momento de extracción de las muestras, así como los protocolos seguidos hasta la obtención de los datos analizados en este proyecto.

2.2.1.1 Animales, régimen de vida y cuidados

Fundado en 1966 y ubicado en Durham (EE.UU.), el Duke Lemur Center (DLC) es un centro especializado en la investigación no invasiva de los lémures. Alberga más de 200 ejemplares de 14 especies, la población más diversa de lémures fuera de su Madagascar natal. La gran experiencia y enorme variedad de especies e individuos que alberga el centro, permitió unas condiciones óptimas para la extracción longitudinal de las muestras biológicas que dieron origen a este y al estudio de Faherty et al. [5]. Estas muestras proceden de cuatro lémures enanos de cola gruesa, dos machos y dos hembras entre los 8 y 13 años de edad [5].

Para conseguir que estos lémures hibernaran durante los meses de enero y febrero, se procedió de la siguiente manera: en noviembre, cuando comienza en el DLC la estación de hibernación, se trasladó a los cuatro individuos a un hibernáculo especial donde fueron expuestos a una cantidad de horas de luz y temperatura cada vez más reducidas. Aunque se encontraban físicamente aislados, para mimetizar las condiciones de su hábitat natural, tenían acceso auditivo y visual a los demás individuos y contaban con unos tubos para anidar que imitan las madrigueras arbóreas utilizadas en libertad. La cantidad de calorías ingeridas se fue reduciendo hasta un 66%. La dieta previa al letargo

estaba compuesta por alimento para primates Purina®, frutas, verduras, gusanos de la harina y agua *ad libitum* [5].

2.2.1.2 Fechas de recolección de muestras experimentales

Se establecieron tres fechas de recolección de muestras, fechas representativas de tres momentos importantes en el ciclo de masa corporal anual: octubre, diciembre y febrero.

- Octubre es el mes previo al comienzo de la estación invernal. En este periodo, el comportamiento de los lémures enanos de cola gruesa no se distingue, a grandes rasgos, del comportamiento de cualquier otro lémur no hibernador. Por tanto, el perfil de expresión génica en este mes debería ser homólogo al de cualquier animal no hibernador.
- Diciembre es el mes previo al letargo estacional y supone el periodo en el que los animales en libertad acumulan la mayor cantidad de grasa posible para superar el invierno. Los cambios en la expresión génica que se producen en este periodo respecto al anterior podrían revelar cuáles son los genes involucrados en este comportamiento hiperfágico.
- Finalmente, febrero es uno de los meses en los que estos animales permanecen en estado de hibernación. Cabría esperar un perfil de expresión génica muy diferente a los anteriores el cual nos ayudara a comprender lo que sucede durante la hibernación a nivel celular.

Debido a las normas impuestas por el gobierno federal estadounidense, los individuos objeto de este estudio no tuvieron un acceso a alimentos *ad libitum* durante el mes de diciembre. Por este motivo, la única diferencia en las condiciones de muestreo entre octubre y diciembre fueron las horas diarias de luz, reducidas gradualmente de 11,5 a 9,5 horas [5]. La restricción en la dieta puede haber tenido consecuencias en la comparación de la expresión génica entre octubre y diciembre.

Durante un periodo de letargo ordinario, los lémures enanos de cola gruesa pueden experimentar una bajada de temperatura de hasta 15 ºC, una reducción en el ritmo cardíaco de hasta 3-5 latidos por minuto y episodios de respiraciones infrecuentes e irregulares [41, 71]. Para verificar el estado fisiológico en cada fecha de recolección de muestras, se registraron la Tc rectal y la masa corporal. Las muestras del WAT, recogidas

en febrero, fueron obtenidas 2-3 días después de que los animales comenzaran un periodo de letargo determinado por ausencia en la alimentación [5].

2.2.1.3 Extracción de muestras

La recogida de las muestras se realizó en el mismo horario (entre las 9 y las 11 a.m.), de esta manera se evitan fluctuaciones circadianas que pudieran verse reflejadas en los perfiles de expresión génica [5]. El área elegida para realizar la biopsia fue la sección de la cola, donde el almacenamiento de grasa es más concentrado [72]. Las muestras del WAT se extrajeron mediante técnicas de mínima invasión; para ello, los animales fueron anestesiados con ketamina (10 mg/kg, Ketaject®, Bioniche Teoranta, Irlanda) y midazolam (0,25 mg/kg, Bedford Laboratories, EE.UU.) por un veterinario del DLC. Se afeitó la zona y se limpió la piel tres veces, alternando una solución de clorhexidina y alcohol. Se insertó una aguja de biopsia Tru Cut 16G (Jorgensen Laboratories, EE.UU.) en el área elegida. El tejido obtenido fue depositado directamente en la solución de estabilización de ARN (Qiagen, EE.UU.). Las muestras se mantuvieron 4 ºC durante toda la noche y a la mañana siguiente se almacenaron a -20 ºC [5].

2.2.1.4 Purificación del ARN y amplificación del transcriptoma completo

La totalidad del WAT obtenido se utilizó para la extracción de ARN. Para minimizar la variabilidad entre muestras debida a los lotes, fueron procesadas el mismo día siguiendo un protocolo optimizado de TRIzol®, junto con el kit de tejido de microarray (Qiagen). Siguiendo este protocolo, el tejido congelado se homogeneizó brevemente en 500 µl de reactivo TRI (Ambion, EE.UU.) con un homogenizador de mano. Para la extracción se utilizaron 50 µl de BCP (1-bromo-3-cloropropano) y, posteriormente, se centrifugó la totalidad del volumen a 12000 x *g* durante 15 minutos a 4 ºC. Tras la centrifugación, el ARN queda retenido en la capa acuosa. Esta fue transferida a un nuevo tubo de centrifugación de 1,5 ml y se le añadió un volumen del 70% de EtOH. La totalidad de este volumen se cargó en columnas de filtración de RNeasy® (Qiagen) y se siguió el protocolo de acuerdo con las instrucciones del fabricante del kit, que incluyen un paso para eliminar cualquier residuo de ADN a través de DNasa en columna. La integridad y la concentración de ARN se evaluó con el bioanalizador Agilent 2100 (Santa Clara, EE.UU.) del Instituto de Ciencias Genómicas de la Universidad de Duke siguiendo las normas de

las instalaciones de microarray. Los valores del número de integridad del ARN (RIN) de los extractos estuvieron entre 3,5 y 6,7 y las concentraciones entre 2 y 24 ng/µl. Debido al alto contenido lipídico de las muestras del WAT la extracción de ARN fue insuficiente para los procesos posteriores de secuenciación. Por eso, antes de la secuenciación con Illumina, se amplificó el transcriptoma completo en todos los extractos.

Para evitar sesgos debidos a diferencias en las concentraciones de ARN, este fue diluido a una concentración de 2 ng/µl en un volumen total de 5 µl antes de la amplificación. El kit Ovation® RNA-Seq System V2 (San Carlos, EE.UU.), de acuerdo con las instrucciones del fabricante, permite preparar ADNc amplificado a partir del ARN total para que sea utilizado en la secuenciación posterior. Este proceso consiste en una amplificación isotérmica de *primer* único que amplifica el ARN total a ADNc y agota el ARNr sin preselección de ARNm. Las muestras de ADNc amplificado fueron posteriormente purificadas usando el kit RNeasy® MinElute™ Cleanup (Qiagen) de acuerdo con el protocolo del fabricante. Las concentraciones de ADNc tras la amplificación, se encontraban entre 347 y 461 ng/µl. Un análisis preliminar confirmó que la amplificación del transcriptoma integral no introducía sesgos significativos en frecuencias relativas de ARNm [73]. El ADNc amplificado se envió al recurso compartido de secuenciación de genoma de la Universidad de Duke para la preparación de la biblioteca y su posterior secuenciación.

2.2.1.5 Preparación de la biblioteca y secuenciación con Illumina

A partir del ADNc amplificado previamente, se prepararon las bibliotecas utilizando el kit de Illumina TruSeq®. Durante la preparación de una biblioteca, el ADNc es fragmentado y unos adaptadores son ligados a ambos extremos de los fragmentos. Estos adaptadores son unos oligos cuya secuencia sirve para unirse al secuenciador, diferenciar los fragmentos de diferentes muestras, diferenciar los extremos, permitir la unión de *primers*, etc. La distribución del tamaño de la biblioteca final fue determinada con el bioanalizador Agilent 2100 y el tamaño de los insertos estaban entre 206 y 246 pb.

Para la secuenciación de las bibliotecas se utilizó una plataforma Illumina HiSeq™ 2000 Sequencing System (San Diego, EE.UU.) haciendo uso del modo «rapid-run» con lecturas de 150 pb *paired-end* (PE). Los secuenciadores Illumina tienen una lámina de cristal dividida en carriles, llamada célula de flujo que está cubierta por dos tipos de oligos.

Estos oligos se encuentran anclados por uno de sus extremos a la célula de flujo, quedando el otro extremo libre. Cuando las bibliotecas se introducen en uno de los carriles de la célula de flujo, los adaptadores de los fragmentos se hibridan con los oligos de la superficie de la célula de flujo. Gracias a la acción de la polimerasa, se crea una copia complementaria de cada fragmento. Tras eliminar los fragmentos, se amplifican las copias complementarias mediante un proceso que se denomina amplificación de puente. Durante este proceso, las hebras de ADN ancladas en la célula de flujo se doblan y se hibridan con los adaptadores de su extremo libre a un oligo del otro tipo. Se crea una copia complementaria de la primera copia complementaria con ayuda de la polimerasa. Este proceso se repite hasta que obtenemos aproximadamente un millón de copias de cada fragmento. Tras la amplificación, se eliminan las copias complementarias al fragmento original y se comienza la secuenciación *forward*. Un *primer* se une al adaptador libre de cada copia para que, sobre este *primer*, la polimerasa introduzca los nucleótidos modificados complementarios. Cada uno de los cuatro tipos de nucleótidos está marcado con un fluoróforo que emite a diferente longitud de onda y durante la extensión del *primer* la fluorescencia de cada nucleótido es registrada como A, C, G o T (A: adenina; C: citosina; G: guanina; T, timina). En nuestro caso, este proceso se repitió 150 veces para obtener lecturas de secuencias de 150 nucleótidos de longitud. Finalmente, se secuencia la hebra *reverse* siguiendo los mismos pasos. Para evitar efectos de lote relacionados con el carril, el contenido de cada biblioteca fue dividido en dos y cada mitad fue secuenciada en un carril distinto.

2.2.2 Análisis bioinformático

Para realizar el análisis bioinformático, el dispositivo informático utilizado es un ordenador portátil con un procesador Intel® Core™ i3-3217U CPU @ 1.80GHz × 4 y una memoria RAM de 7,7GB.

Un esquema del *pipeline* computacional para la anotación del genoma del lémur enano de cola gruesa a partir de lecturas de la RNA-seq, el cálculo de los niveles de expresión y la determinación de genes diferencialmente expresados entre condiciones se muestran en la figura 2.

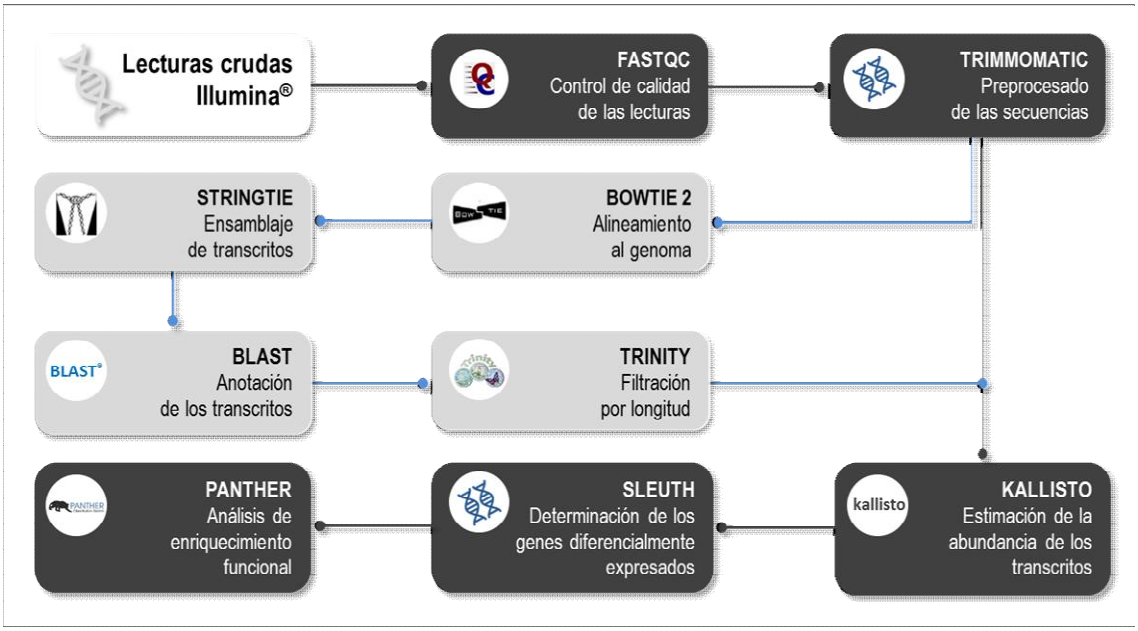

Figura 2. Pipeline bioinformático.

2.2.2.1 Datos de la RNA-seq

El *Sequence Read Archive* (SRA) es una base de datos donde los investigadores pueden depositar los resultados sin procesar de los experimentos de la RNA-seq basados en la tecnología *Next-Generation Sequencing* (NGS) para que sean accesibles a la comunidad científica [74]. Este acceso público y gratuito permite volver a analizar los datos sin la necesidad de repetir los experimentos.

Los datos de la secuenciación de las muestras de estos cuatro lémures enanos de cola gruesa fueron depositados en el SRA del NCBI con número de acceso SRP049327. Un resumen de la información más importante se encuentra representado en la tabla 1.

Tabla 1. Información de los archivos SRA.

Código	Muestra	Género	Estado	Fecha	Animal
SRR1634146	CM19	M	Active1	Octubre	Tanager
SRR1634147	CM20	F	Active1	Octubre	Quetzal
SRR1634148	CM21	M	Active1	Octubre	Osprey
SRR1634149	CM22	F	Active1	Octubre	Towhee
SRR1634152	CM23	M	Active2	Diciembre	Osprey
SRR1634169	CM24	F	Active2	Diciembre	Towhee
SRR1634266	CM25	M	Active2	Diciembre	Tanager
SRR1634268	CM27	F	Torpor	Febrero	Towhee
SRR1634269	CM28	F	Torpor	Febrero	Quetzal

Los archivos de datos están vinculados al código SRR y son accesibles gracias a la aplicación SRA Toolkit (versión 2.8.2). Este programa permite descargar los archivos vinculados a cada código SRR y validar su integridad.

Hemos descargado los archivos en formato FASTQ. Un archivo FASTQ contiene, por un lado, la secuencia de nucleótidos de cada lectura, donde cada nucleótido es representado por una letra (A, C, G, T) y, por otro lado, la calidad de la secuenciación de los nucleótidos es representada por una secuencia de caracteres American Standard Code for Information Interchange (ASCII), donde la posición de cada uno está vinculada a la posición de cada nucleótido (tabla 2).

Tabla 2. Caracteres ASCII de calidad de secuenciación de nucleótidos en un archivo FASTQ, siendo la menor calidad ! y la mayor calidad ~.

!"#$%&'()*+,-./0123456789:;<=>?@ABCDEFGHIJKLMNOPQRSTUVWXYZ[\]^_`abcdefghijklmnopqrstuvwxyz{|}~

Obtenemos dos archivos FASTQ por cada muestra secuenciada, uno contiene las lecturas de las secuencias *forward* y el otro las secuencias *reverse*.

2.2.2.2 Control de calidad

El control de calidad de los datos contenidos en los archivos FASTQ consiste en el análisis de la calidad de las secuencias, el contenido en CG, la presencia de los adaptadores, k-meros sobrerrepresentados y lecturas duplicadas. Este análisis permite la detección de errores de secuenciación, artefactos producidos durante la amplificación y contaminaciones. Hemos realizado todos estos análisis con FastQC (versión v0.11.5) que representa el resultado de estos análisis a través de tablas y gráficos.

2.2.2.3 Filtración de lecturas de secuenciación

La presencia de lecturas de baja calidad o la presencia de lecturas que contengan la secuencia de los adaptadores insertados para la secuenciación o, incluso, la presencia de lecturas que contengan bases secuenciadas con baja calidad en los extremos puede perjudicar el resultado del análisis. Por esta razón, hemos eliminado total o parcialmente algunas de las lecturas.

Atendiendo a los resultados obtenidos en el análisis de calidad, establecimos los siguientes tres parámetros de filtración:

- Si las tres primeras o las tres últimas bases de una lectura tenían una puntuación Phred inferior a 30 eran eliminadas.

- Si la calidad media de cuatro nucleótidos consecutivos empezando por el extremo 5' descendía por debajo de una puntuación Phred de 20, la lectura era cortada y la mitad que contenía el extremo 3' era eliminado del archivo FASTQ.

- Finalmente, las secuencias de los adaptadores que estaban presentes en las lecturas fueron eliminados por homología con las secuencias que contiene el archivo TruSeq3-PE.fa. Este archivo contiene la secuencia de los adaptadores que son insertados en los fragmentos de ADN necesarios para la secuenciación. Se eliminaron de las lecturas las secuencias que obtuvieron una puntuación de homología por encima de 30 en las lecturas palindrómicas (cuando la lectura apareada también contenía el adaptador) y por encima de 10 si no lo era.

Acabado este proceso de filtrado, todas las lecturas con una longitud inferior a 50 bases fueron eliminadas, ya que cuanto menor es la longitud de una lectura mayor es la probabilidad de que exista una secuencia igual en el genoma por casualidad.

Todos estos criterios de filtración fueron introducidos en el comando de la aplicación Trimmomatic (versión 0.39) [13].

2.2.2.4 Transcriptoma de referencia

El transcriptoma es el conjunto de moléculas de ARN (ácido ribonucleico) presentes en una célula o conjunto de células en un individuo en un momento dado. Para conseguir las secuencias de nucleótidos de estas moléculas, se suelen utilizar las anotaciones que la comunidad científica va haciendo sobre la secuencia del genoma. Cuando se publicó el estudio de Faherty et al. [5], el genoma del lémur enano de cola gruesa todavía no se había secuenciado; por eso, los autores tuvieron que ensamblar un transcriptoma *de novo* valiéndose únicamente de las lecturas obtenidas en la secuenciación de las muestras y anotando sus resultados por similitud con otras especies.

En nuestro caso, disponemos del genoma pero es tan reciente que todavía no tiene anotaciones, así que hemos alineado las lecturas contra el genoma, hemos ensamblado

los transcritos potenciales y, finalmente, hemos anotado los transcritos homólogos con proteínas caracterizadas en humanos.

2.2.2.4.1 Alineamiento de las lecturas al genoma

En la actualidad existen dos ensamblajes distintos del genoma del lémur enano de cola gruesa (ASM808673v1, CheMed_v1_BIUU). Teniendo en cuenta las estadísticas globales de ambos genomas, escogimos trabajar con ASM808673v1. Este genoma resultó tener su origen en un individuo híbrido lo que provocaba un bajo rendimiento en el mapeado de las lecturas. Aunque esto supuso una demora respecto al plan de trabajo previsto, aprovechamos este bajo rendimiento para comparar la eficiencia de HISAT2 (versión 2.1.0) [75, 76] y Bowtie2 (versión 2.3.4.1) [77] en estas condiciones tan desfavorables. Bowtie2 en modo «very-sensitive-local» produjo una mayor cantidad de alineamientos.

El alineamiento de las lecturas sobre el genoma CheMed_v1_BIUU con Bowtie2 produjo un alineamiento mucho más elevado que sobre el genoma ASM808673v1.

2.2.2.4.2 Ensamblaje de los tránscritos

El programa *StringTie* (versión 2.0.4) es un ensamblador de tránscritos a partir de datos de la RNA-seq rápido y eficiente que utiliza un método computacional que identifica entre un 30 y un 60% más transcritos que el siguiente mejor ensamblador actual (Cufflinks) [78].

Para ensamblar un conjunto de transcritos a partir de alineamientos en el genoma, utilizando *StringTie* es necesario ensamblar los tránscritos en cada muestra por separado, para posteriormente fusionarlos de manera no redundante.

2.2.2.4.3 Proteínas homólogas en humanos

Aprovechando que existe un gran número de proteínas caracterizadas en humanos y que la distancia evolutiva entre el ser humano y el lémur no es muy grande, compararemos las secuencias de nuestros transcritos potenciales con las secuencias de las proteínas anotadas en humanos (GRCh38.p13). Para ello hemos utilizado BlastX (versión 2.6.0+) [79] que, a partir de la traducción de una secuencia de nucleótidos en los

seis posibles marcos de lectura, compara estas secuencias traducidas con una base de datos de proteínas.

Hemos considerado que un tránscrito es homólogo a una proteína cuando su significancia estadística es inferior a un e-valor de 10^{-4}, la misma restricción que se utilizó en el artículo de Faherty et al. [5]. Hemos establecido en 10 el número máximo de resultados por tránscrito, porque BLAST usa un algoritmo heurístico que no garantiza el mejor resultado. Finalmente, hemos seleccionado una única proteína homóloga por tránscrito, siempre y cuando este suponga un 70% o más de la longitud de la proteína respecto a su homóloga humana. Para comprobar que se cumplía esta condición, hemos utilizado una de las herramientas de Trinity (versión 2.8.6) [80].

2.2.2.5 Estimación de la abundancia de los transcritos y genes diferencialmente expresados

Hemos eliminado del análisis a un individuo (Towhee) debido a que sufrió una significativa pérdida de peso antes del letargo, probablemente como consecuencia de la restricción en la comida mencionada en el apartado 2.2.1.2.

Para detectar diferencias entre los distintos estados (activo 1, activo 2 y letargo) hemos comparado la expresión de los genes por pares de estados teniendo en cuenta la variabilidad entre los individuos.

2.2.2.5.1 Estimación de la abundancia de cada transcrito

Para cuantificar la abundancia de los transcritos a partir de las lecturas, hemos utilizado Kallisto (versión 0.46.0) [17], un programa basado en la nueva idea de pseudoalineamiento para determinar rápidamente la compatibilidad de las lecturas con los transcritos, sin necesidad de alineamiento.

Kallisto puede ejecutarse en modo «bootstrap» lo que permite obtener estimaciones de incertidumbre para los niveles de expresión. La ejecución con *bootstraps* es obligatoria si se desea realizar un análisis de expresión diferencial de isoformas con Sleuth.

2.2.2.5.2 Genes diferencialmente expresados

Sleuth [81] es un paquete de R que se utiliza para el análisis de la expresión diferencial a partir de la cuantificación de los transcritos realizada con Kallisto. Si bien se podrían usar otros paquetes de expresión diferencial como limma o DESeq2 para analizar los datos obtenidos con Kallisto, Sleuth tiene en cuenta la variabilidad inherente en la cuantificación de la transcripción. Además, permite diseños experimentales con múltiples factores variables como la condición (activo 1, activo 2 y letargo) y el individuo (tres individuos por condición) en muestreos longitudinales.

2.3 Resultados

2.3.1 Filtración de lecturas de secuenciación

Los archivos FASTQ descargados de la SRA del NCBI contienen las secuencias de nucleótidos de las lecturas obtenidas en la RN-seq. El número de lecturas en cada archivo varía entre 22.369.995 y 29.043.051 (tabla 3). Todas las lecturas tienen una longitud de 151 nucleótidos y el porcentaje medio de CG por secuencia oscila entre un 41 y un 45%.

Tabla 3. Número de lecturas secuenciadas y de lecturas que superan la filtración con Trimmomatic por muestra.

Muestras	Núm.de lecturas secuenciadas	Núm.de lecturas tras la filtración
SRR1634146	28112230	20628770
SRR1634147	25225178	18549336
SRR1634148	26428476	19157147
SRR1634149	28515033	20821444
SRR1634152	28519876	20668848
SRR1634169	29904351	21889839
SRR1634266	29314627	20260747
SRR1634268	22796616	16439909
SRR1634269	23540903	17144009
SRR1634270	25908964	18710595
SRR1634272	22369995	15394381
SRR1634454	28635253	20922999

Tras la filtración con Trimmomatic, el número de secuencias por archivo se redujo entre un 36,1 y un 11,1%, conservándose entre 21.889.839 y 15.394.381 secuencias (tabla 3). El contenido en CG se mantuvo entre el 41 y el 45%.

2.3.2 Alineamiento de las lecturas

Como se ha explicado en el apartado 2.2.2.4.1 (véase pág. 21), hemos alineado las lecturas de RNA-seq a los dos genomas disponibles de lémur enano de cola gruesa. En la tabla 4 se encuentra representado un resumen de los resultados obtenidos.

Tabla 4. Resultados del alineamiento a los genomas ASM808673v1 y CheMed_v1_BIUU con Bowtie2.

ASM808673v1												
Muestras	Lecturas	Parejas	Lecturas apareadas alineadas					Lecturas que no se alinean apareadas	Lecturas no apareadas alineadas			Porcentaje de alineamiento general
			Concordantemente			Discordantemente	Concord. o discord.					
			Ninguna vez	Solo una vez	Más de una vez	Solo una vez	Ninguna vez		Ninguna vez	Solo una vez	Más de una vez	
SRR1634146	20628770	20628770(100.00%)	10620431(51.48%)	7361072(35.68%)	2647267(12.83%)	651766(6.14%)	9968665	19937330	17351866(87.03%)	1772892(8.89%)	812572(4.08%)	57.94%
SRR1634147	18549336	18549336(100.00%)	7440164(40.11%)	7715500(41.59%)	3393672(18.30%)	650834(8.75%)	6789330	13578660	10987479(80.92%)	1418097(10.44%)	1173084(8.64%)	70.38%
SRR1634148	16439909	16439909(100.00%)	6395073(38.90%)	7073253(43.02%)	2971583(18.08%)	539846(8.44%)	5855227	11710454	9415114(80.40%)	1239806(10.59%)	1055534(9.01%)	71.37%
SRR1634149	20821444	20821444(100.00%)	11546904(55.46%)	6743472(32.39%)	2531068(12.16%)	602598(5.22%)	10944306	21888612	19011166(86.85%)	1954979(8.93%)	922467(4.21%)	54.35%
SRR1634152	20668848	20668848(100.00%)	8408916(40.68%)	8770943(42.44%)	3488989(16.88%)	716855(8.52%)	7692061	15384122	12463534(81.02%)	1620847(10.54%)	1299741(8.45%)	69.85%
SRR1634169	21889839	21889839(100.00%)	8846479(40.41%)	9053927(41.36%)	3989433(18.23%)	763387(8.63%)	8083092	16166184	13011728(80.49%)	1710451(10.58%)	1444005(8.93%)	70.28%
SRR1634266	20260747	20260747(100.00%)	8795700(43.41%)	7882248(38.90%)	3582799(17.68%)	636652(7.24%)	8159048	16318096	13876742(85.04%)	1444206(8.85%)	997148(6.11%)	65.75%
SRR1634268	16439909	16439909(100.00%)	6395073(38.90%)	7073253(43.02%)	2971583(18.08%)	539846(8.44%)	5855227	11710454	9415114(80.40%)	1239806(10.59%)	1055534(9.01%)	71.37%
SRR1634269	17144009	17144009(100.00%)	4973682(29.01%)	8485826(49.50%)	3684501(21.49%)	588882(11.84%)	4384800	8769600	6722938(76.66%)	978273(11.16%)	1068389(12.18%)	80.39%
SRR1634270	18710595	18710595(100.00%)	6185931(33.06%)	9091798(48.59%)	3432866(18.35%)	638907(10.33%)	5547024	11094048	8852000(79.79%)	1160641(10.46%)	1081407(9.75%)	76.34%
SRR1634272	15394381	15394381(100.00%)	5713646(37.12%)	6692617(43.47%)	2988118(19.41%)	499481(8.74%)	5214165	10428330	8538897(81.88%)	1072661(10.29%)	816772(7.83%)	72.27%
SRR1634454	20922999	20922999(100.00%)	6409933(30.64%)	10324584(49.35%)	4188482(20.02%)	708387(11.05%)	5701546	11403092	9124262(80.02%)	1163200(10.20%)	1115630(9.78%)	78.20%

CheMed_v1_BIUU												
Muestras	Lecturas	Parejas	Lecturas apareadas alineadas					Lecturas que no se alinean apareadas	Lecturas no apareadas alineadas			Porcentaje de alineamiento general
			Concordantemente			Discordantemente	Concord. o discord.					
			Ninguna vez	Solo una vez	Más de una vez	Solo una vez	Ninguna vez		Ninguna vez	Solo una vez	Más de una vez	
SRR1634146	20628770	20628770(100.00%)	1184759(5.74%)	6475527(31.39%)	12968484(62.87%)	264706(22.34%)	920053	1840106	394944(21.46%)	287028(15.60%)	1158134(62.94%)	99.04%
SRR1634147	18549336	18549336(100.00%)	1137428(6.13%)	7193148(38.78%)	10218760(55.09%)	323675(28.46%)	813753	1627506	308918(18.98%)	280660(17.24%)	1037928(63.77%)	99.17%
SRR1634148	19157147	19157147(100.00%)	1065002(5.56%)	7485168(39.07%)	10606977(55.37%)	336767(31.62%)	728235	1456470	378558(25.99%)	234166(16.08%)	843746(57.93%)	99.01%
SRR1634149	20821444	20821444(100.00%)	1165176(5.60%)	6291589(30.22%)	13364679(64.19%)	251389(21.58%)	913787	1827574	466988(25.55%)	289903(15.86%)	1070683(58.58%)	98.88%
SRR1634152	20668848	20668848(100.00%)	1394911(6.75%)	8176099(39.56%)	11097838(53.69%)	433466(31.07%)	961445	1922890	358614(18.65%)	369528(19.22%)	1194748(62.13%)	99.13%
SRR1634169	21889839	21889839(100.00%)	1331901(6.08%)	8674155(39.63%)	11883783(54.29%)	421570(31.65%)	910331	1820662	337527(18.54%)	337115(18.52%)	1146020(62.95%)	99.23%
SRR1634266	20260747	20260747(100.00%)	1222159(6.03%)	7114414(35.11%)	11924174(58.85%)	339814(27.80%)	882345	1764690	515262(29.20%)	263345(14.92%)	986083(55.88%)	98.73%
SRR1634268	16439909	16439909(100.00%)	1045759(6.36%)	6855550(41.70%)	8538600(51.94%)	339196(32.44%)	706563	1413126	198655(14.06%)	297371(21.04%)	917100(64.90%)	99.40%
SRR1634269	17144009	17144009(100.00%)	1185692(6.92%)	8215250(47.92%)	7743067(45.16%)	442599(37.33%)	743093	1486186	228426(15.37%)	317140(21.34%)	940620(63.29%)	99.33%
SRR1634270	18710595	18710595(100.00%)	1285693(6.87%)	8706394(46.53%)	8718508(46.60%)	474039(36.87%)	811654	1623308	263592(16.24%)	345813(21.30%)	1013903(62.46%)	99.30%
SRR1634272	15394381	15394381(100.00%)	792693(5.15%)	5935158(38.55%)	8666530(56.30%)	272279(34.35%)	520414	1040828	223636(21.49%)	175966(16.91%)	641226(61.61%)	99.27%
SRR1634454	20922999	20922999(100.00%)	1357286(6.49%)	9803474(46.86%)	9762239(46.66%)	521332(38.41%)	835954	1671908	323009(19.32%)	329576(19.71%)	1019323(60.97%)	99.23%

El porcentaje de alineamiento general de las lecturas al genoma ASM808673v1 es muy inferior al obtenido en el alineamiento al genoma CheMed_v1_BIUU (figura 3), debido seguramente a que la secuencia del genoma ASM808673v1 proviene de un híbrido entre el lémur enano de cola gruesa y una especie cercana.

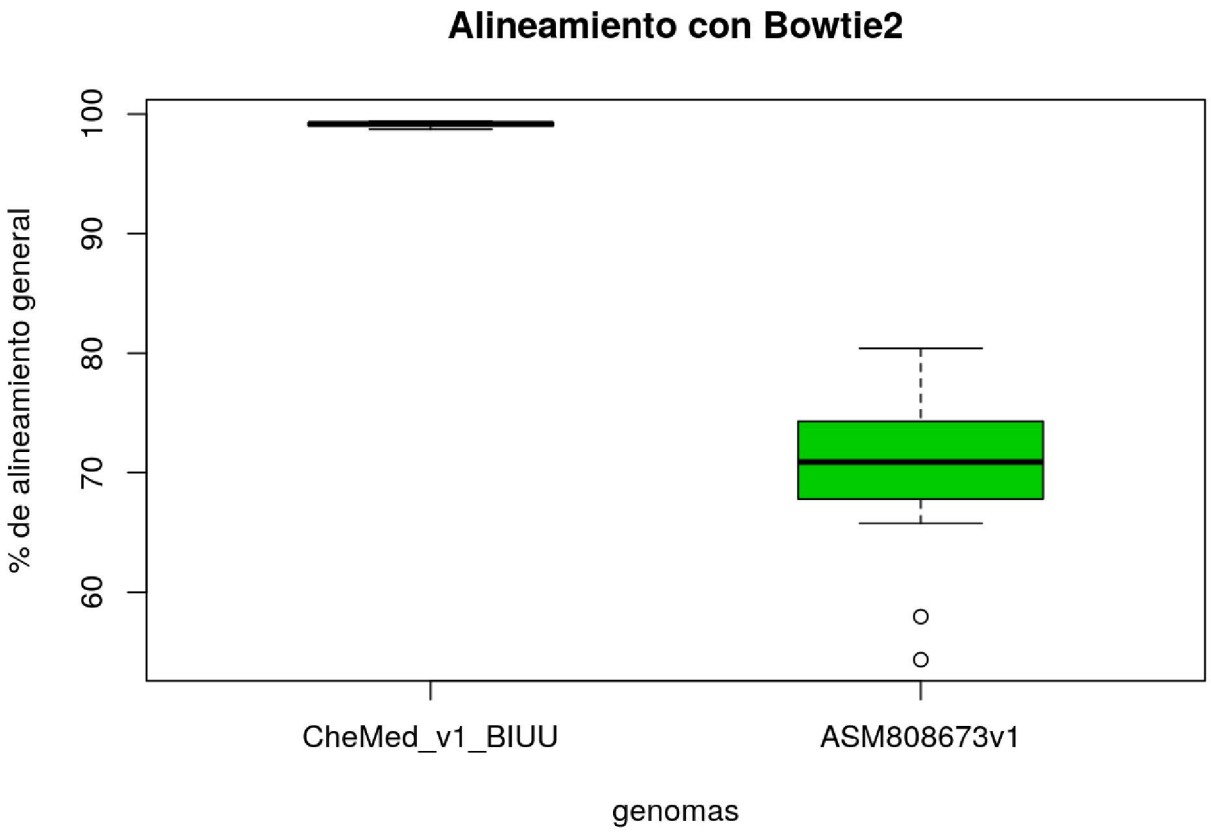

Figura 3. Diagrama de caja y bigotes que muestra los porcentajes de alineamiento general de las lecturas de las muestras a cada genoma.

Aunque el alineamiento general de las lecturas es muy diferente en los dos genomas, la diferencia en el porcentaje de lecturas apareadas que se alinean concordantemente solo una vez no es significativa (figura 4, a). Es decir, el porcentaje de lecturas apareadas que no se alinean concordantemente al genoma ASM808673v1 (figura 4, b) es mayor debido a un menor porcentaje de alineamientos concordantes múltiples (figura 4, c).

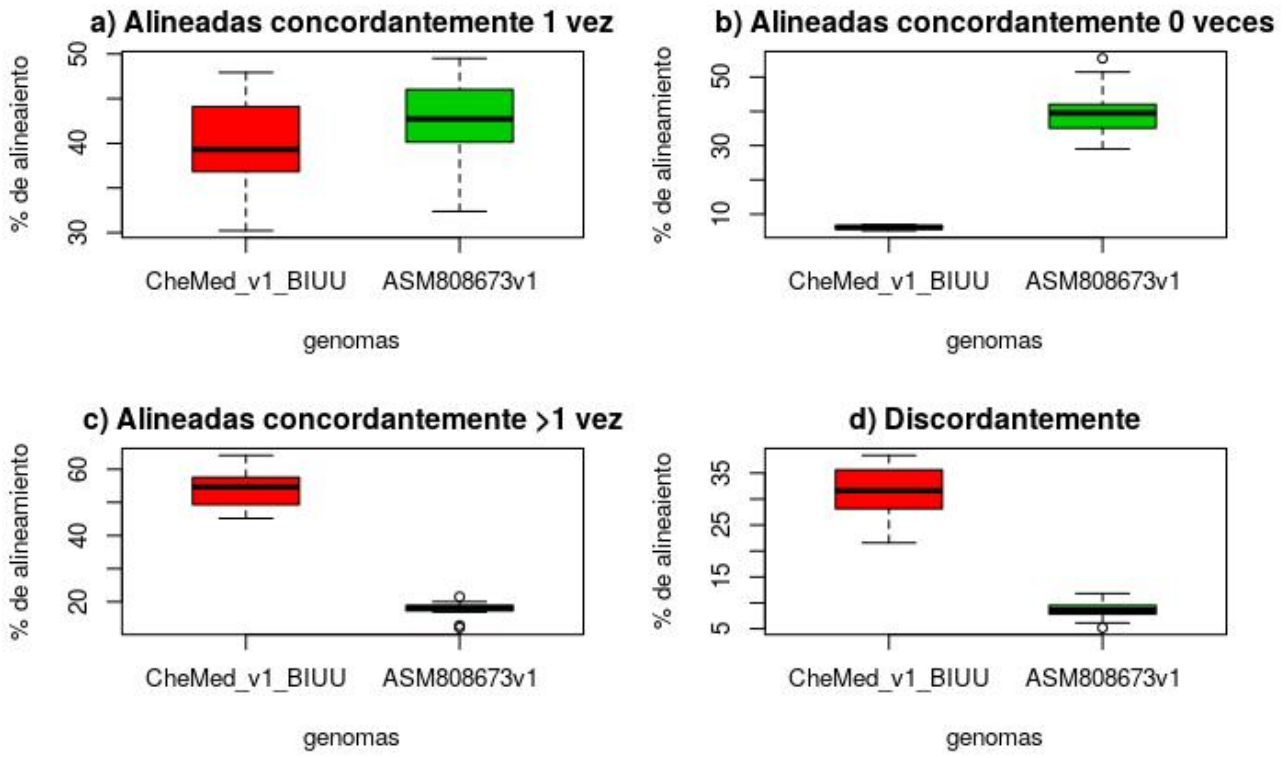

Figura 4. Porcentajes de alineamientos apareados a cada genoma. Diagrama de cajas y bigotes que muestra los porcentajes de las lecturas alineadas: a) concordantemente solo una vez; b) concordantemente cero veces; c) concordantemente más de una vez; d) discordantemente.

Por otro lado, también es inferior el porcentaje de alineamientos discordantes de las lecturas apareadas (figura 4, d) y el de lecturas no apareadas frente al genoma ASM808673v1 en comparación con el alineamiento frente al genoma CheMed_v1_BIUU (figura 5; a), debido principalmente a los alineamientos que se producen más de una vez (figura 5, b, c).

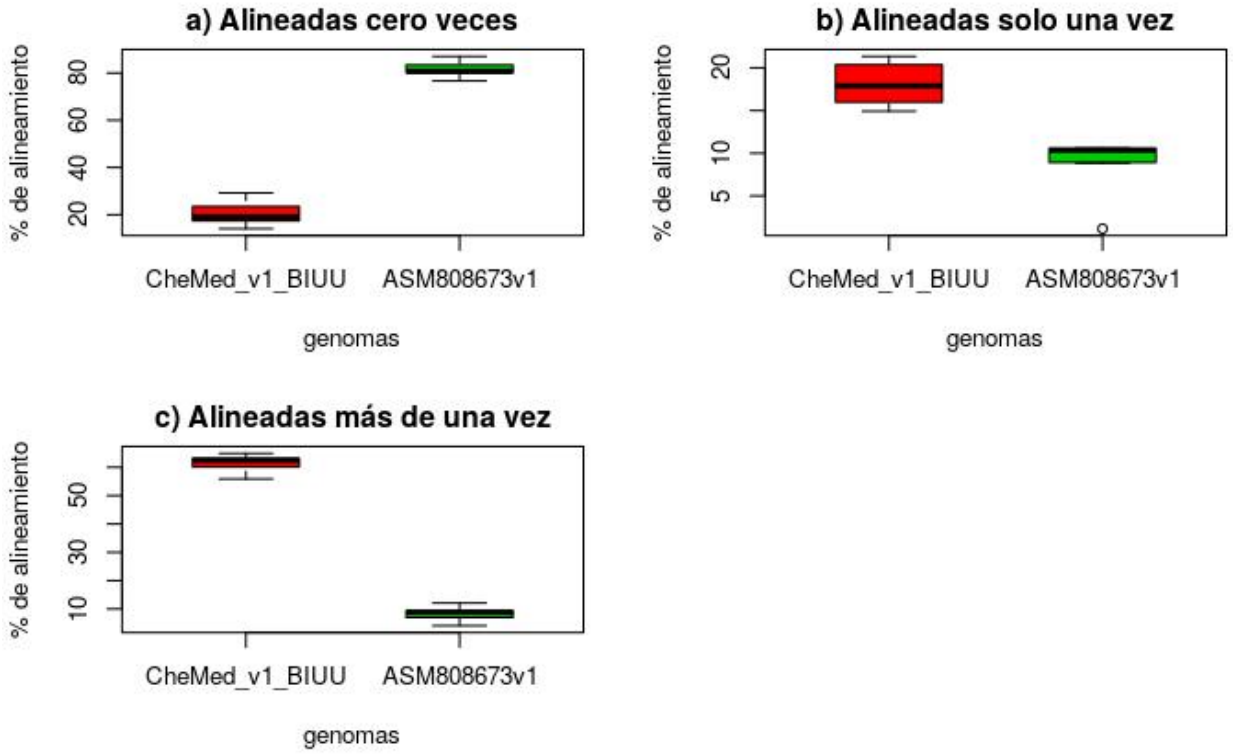

Figura 5. Porcentaje de alineamientos no apareados. Diagrama de cajas y bigotes que muestra los porcentajes de las lecturas que se alinean concordantemente cero veces alineadas por separado: a) cero veces; b) solo; c) más de una vez.

2.3.3 Transcritos ensamblados

Teniendo en cuenta los pobres resultados obtenidos en el alineamiento de las lecturas de RNA-seq al genoma ASM808673v1, hemos decidido ensamblar los transcritos a partir del alineamiento al genoma CheMed_v1_BIUU.

El número de transcritos obtenido por StringTie a partir de los datos de RNA-seq de cada una de las doce muestras está entre 13.871 y 65.851 (tabla 5).

Tabla 5. Número de transcritos ensamblados.

Muestra	Núm. de transcritos
SRR1634146	22303
SRR1634147	65851
SRR1634148	37106
SRR1634149	13871
SRR1634152	41402
SRR1634169	39491
SRR1634266	45908
SRR1634268	35018
SRR1634269	46747
SRR1634270	46799
SRR1634272	39268
SRR1634454	57329

La longitud de estos transcritos varía entre los 200 (mínimo número de nucleótidos que establece StringTie por defecto para los transcritos ensamblados) y los 122.135 nucleótidos. La distribución de las longitudes de estos transcritos está representada en la figura 6 a través de curvas de densidad que indican la proporción de los transcritos en función de su longitud, expresada como el logaritmo en base dos del número de nucleótidos.

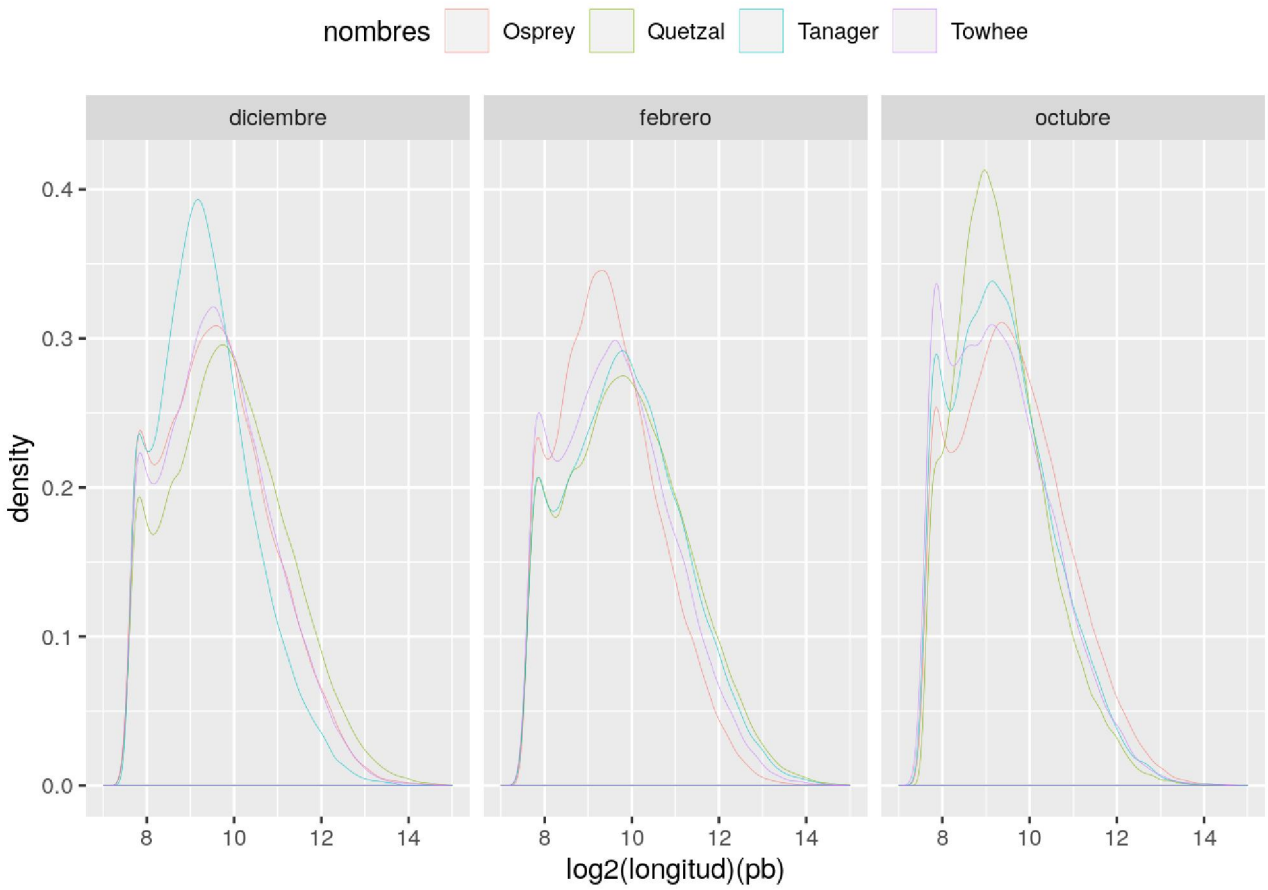

Figura 6. Curvas de densidad de distribuciones de la longitud de los transcritos ensamblados en cada individuo.

La distribución de las longitudes de los transcritos de una muestra sin degradación sigue una distribución normal pero, como podemos observar en la figura 6, la distribución de las longitudes de los transcritos es bimodal. Esto sugiere que hay dos poblaciones de transcritos: una de ellas estaría formada por los reconstruidos a partir del ARN íntegro (los transcritos de mayor longitud) y la otra, por los reconstruidos a partir del ARN degradado (los transcritos de menor longitud). La curva de densidad de las longitudes de los transcritos de la muestra de Quetzal recogida en octubre exhibe la menor proporción de ARN degradado. Usando esta curva como referencia, podemos esperar que la longitud más frecuente en los transcritos reconstruidos por StringTie a partir de ARN no degradado sea cercana a 512 nucleótidos (2^9) que corresponde al máximo valor en la curva de densidad de Quetzal.

La cobertura (*coverage*) es la media de las lecturas que se alinean a los nucleótidos de un transcrito. La cobertura de los transcritos reconstruidos por StringTie varía entre 0,63 y 204.255,22 lecturas y parece estar normalmente distribuida. El valor de cobertura más frecuente en todas las muestras es 4 lecturas (2^2), lo que supone más del 75% de los transcritos (figura 7).

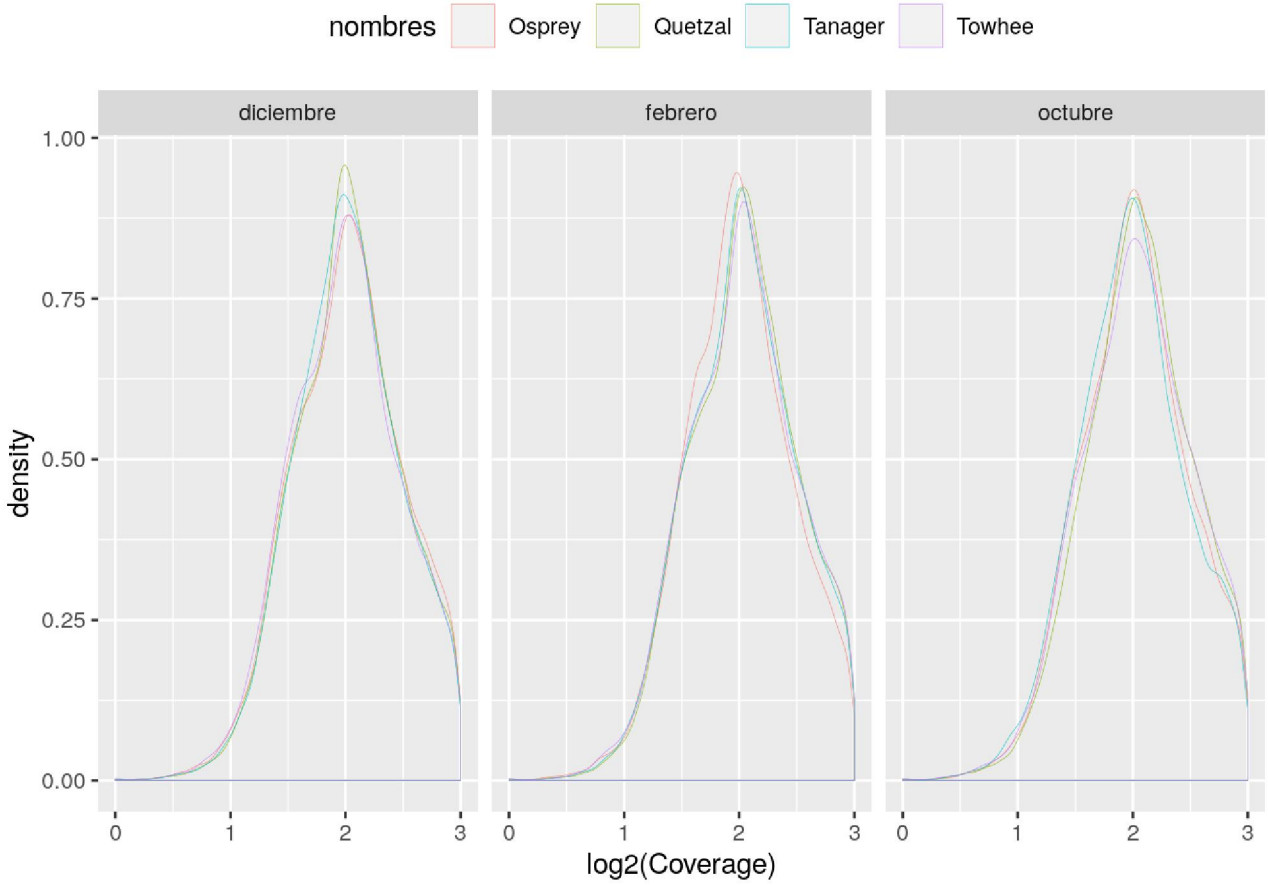

Figura 7. Curvas de densidad de distribuciones de la cobertura en cada individuo.

Los transcritos por millón (TPM) son una medida de la proporción de transcritos en una muestra de ARN. Para determinar los TPM de cada transcrito en cada muestra se utiliza la siguiente fórmula, en la cual X_i es el número de lecturas alineadas por base de transcritos y l_i es la longitud del transcrito en nucleótidos:

$$\mathrm{TPM}_i = \frac{X_i}{\widetilde{l}_i} \cdot \left(\frac{1}{\sum_j \frac{X_j}{\widetilde{l}_j}} \right) \cdot 10^6$$

Los fragmentos por kilobase de exón por millón de lecturas (FPKM) son los fragmentos secuenciados cuyas lecturas se alinean por cada mil bases del transcrito por cada $N/10^6$ fragmentos secuenciados. N es el número de fragmentos de ARN secuenciados en una muestra. Para determinar los FPKM de cada transcrito se utiliza la siguiente fórmula, en la cual X_i es el número de lecturas alineadas por base de transcritos, l_i es la longitud del transcrito en nucleótidos y N es el número de fragmentos de ARN secuenciados en una muestra:

$$\mathrm{FPKM}_i = \frac{X_i}{\left(\frac{\widetilde{l}_i}{10^3}\right)\left(\frac{N}{10^6}\right)} = \frac{X_i}{\widetilde{l}_i N} \cdot 10^9$$

Tanto los FPKM como los TPM miden las lecturas alineadas en un transcrito teniendo en cuenta la longitud de este último, aunque los FPKM tienen también en cuenta el número total de lecturas alineadas. Esto supone que, entre dos transcritos con el mismo número de lecturas alineadas, el transcrito de mayor longitud tendrá el valor de TPM o de FPKM menor. Los valores de TPM varían entre 2,6 y 370.677,4, mientras que los valores de FPKM varían entre 0,81 y 72.721,04. Como podemos observar en las figuras 8 y 9, la distribución de los TPM y los FPKM de los transcritos ensamblados es bimodal. Una de las modas refleja la población de transcritos ensamblados a partir del ARN degradado (la moda de menor valor en TPM y FPKM) y la otra, la población de transcritos ensamblados a partir del ARN íntegro (la moda de mayor valor en TPM y FPKM).

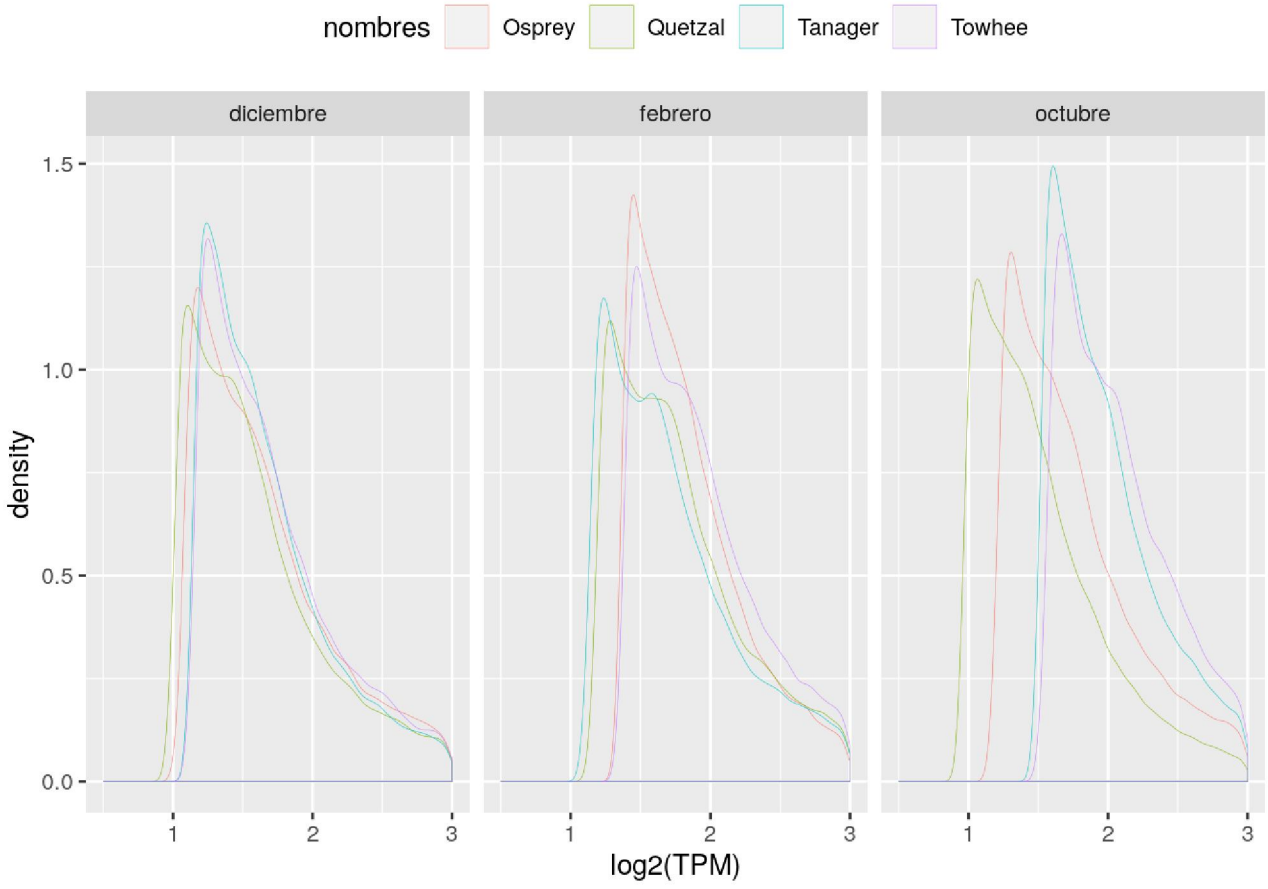

Figura 8. Curvas de densidad de distribuciones de los TPM en cada individuo.

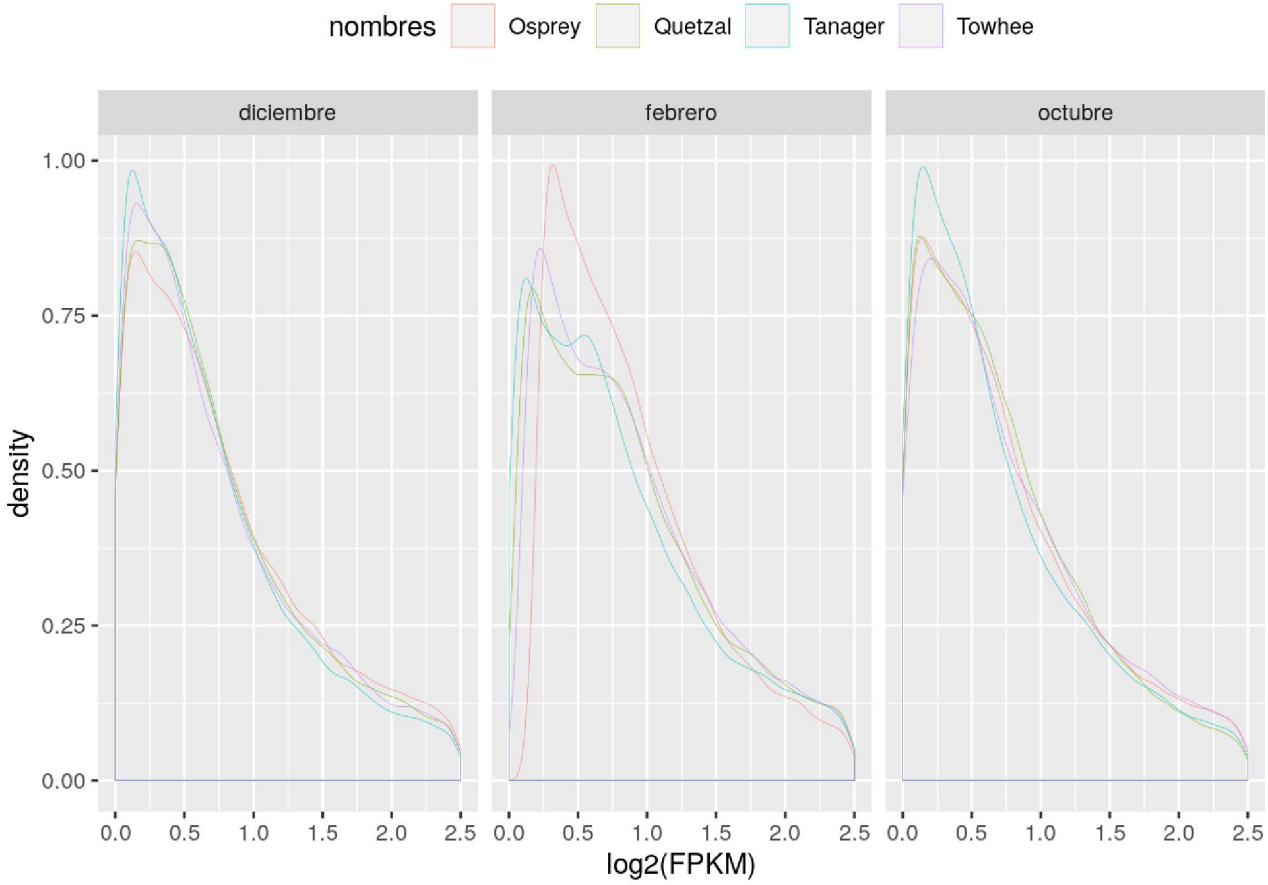

Figura 9. Curvas de densidad de distribución de los FPKM en cada individuo.

Los transcritos ensamblados en cada muestra por separado son posteriormente fusionados de manera no redundante con StringTie. El número de transcritos que StringTie ha obtenido de esta fusión asciende a 115.050. Las longitudes de estos transcritos varían entre 200 y 142.148 nucleótidos. Como podemos observar en la figura 10, la distribución de las longitudes de los transcritos, aunque es bimodal, parece reflejar una menor proporción de transcritos ensamblados a partir del ARN degradado que las muestras por separado. La longitud más frecuente en los transcritos fusionados por StringTie es aproximadamente 512 nucleótidos (2^9) que corresponde al máximo en la gráfica (como ocurría en la muestra de Quetzal recogida en octubre).

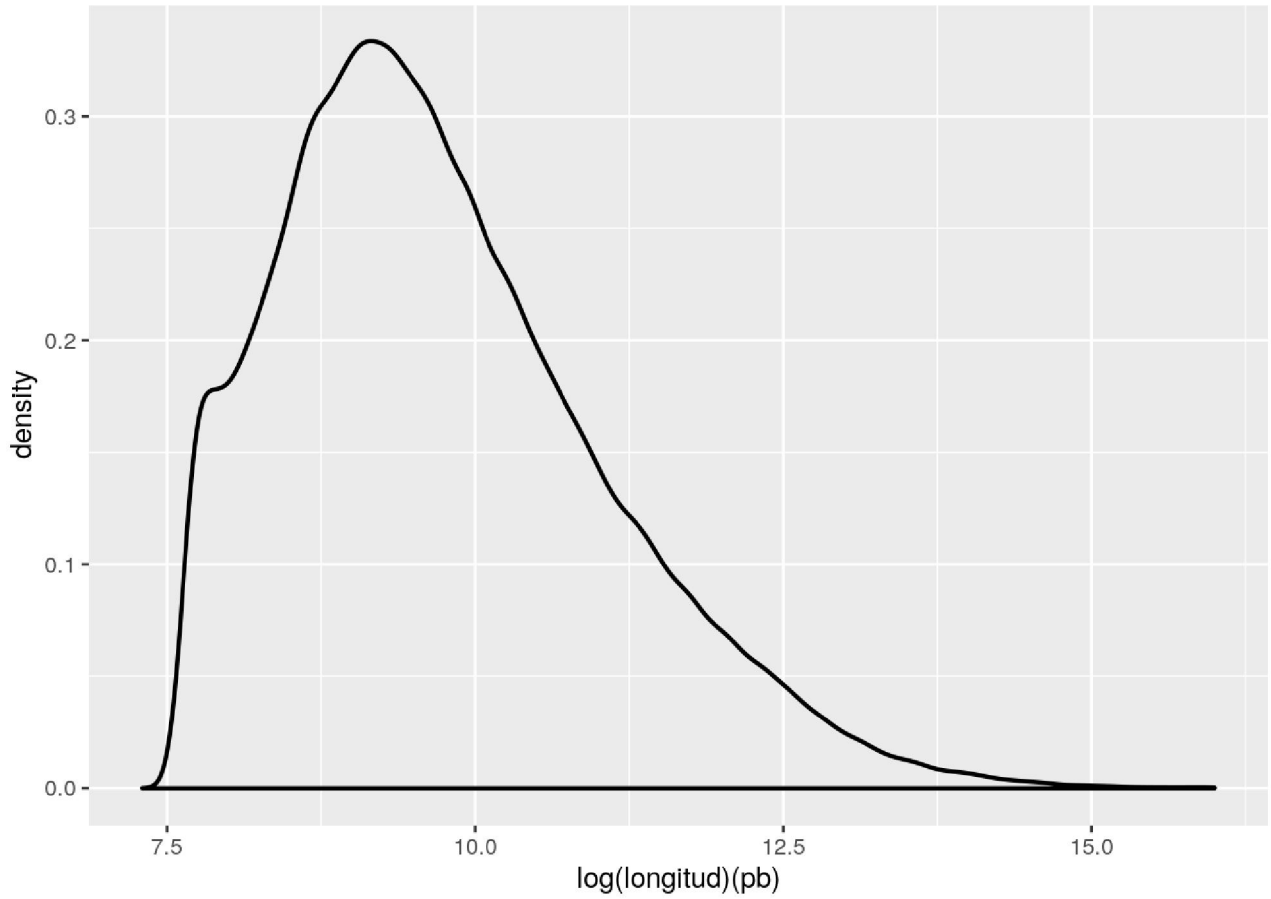

Figura 10. Curva de densidad de distribución de la longitud de los transcritos fusionados.

2.3.4 Transcritos con proteínas homólogas en humanos

De los 115.050 transcritos obtenidos por StringTie solo nos hemos interesado por aquellos que tenían una proteína homóloga caracterizada en humanos. Esto supone descartar 58.885 transcritos potenciales, dado que desconocemos la función que desempeña la proteína codificada.

Una de las medidas para evaluar la calidad del ensamblaje de un transcriptoma es examinar el número de transcritos ensamblados que son completos o casi completos. Se puede hacer este análisis alineando los transcritos ensamblados con los transcritos del transcriptoma de un organismo relacionado (en este caso, el conjunto de las secuencias de las proteínas humanas conocidas) para determinar el número de proteínas (una por transcrito como máximo) con la mayor coincidencia que se alinean en un porcentaje de su longitud.

Tabla 6. Número de transcritos en función del porcentaje de la longitud de cobertura de su proteína homóloga humana.

Porcentaje de cobertura	Transcritos en cada tramo	Transcritos con porcentaje ≥
100	884	884
90	413	1297
80	444	1741
70	511	2252
60	758	3010
50	1019	4029
40	1701	5730
30	2965	8695
20	6234	14929
10	8149	23078

Vemos que hay 2.252 proteínas que son representadas por transcritos de longitud casi completa, >70% de cobertura de alineamiento (tabla 6).

De los 56.165 transcritos anotados, solo aquellos cuya longitud suponía un ≥70% con respecto a su proteína homóloga han sido considerados. Esto supone un transcriptoma de 2.531 secuencias, de las cuales 791 son isoformas de un mismo gen o no disponemos de su identificación *Ensembl* para proteínas.

El transcriptoma del artículo de Faherty et al. [5] tiene 9.540 transcritos mientras que el nuestro tiene 1.720. Ambos transcriptomas tienen 606 transcritos en común.

2.3.5 Identificación de genes diferencialmente expresados

Para determinar la relación que existe entre las muestras en función del estado fisiológico o el individuo al que pertenecen, hemos realizado un análisis de componentes principales. El primer componente representa prácticamente la totalidad de la variabilidad, dejando muy poco porcentaje de variabilidad al segundo componente (figura 11).

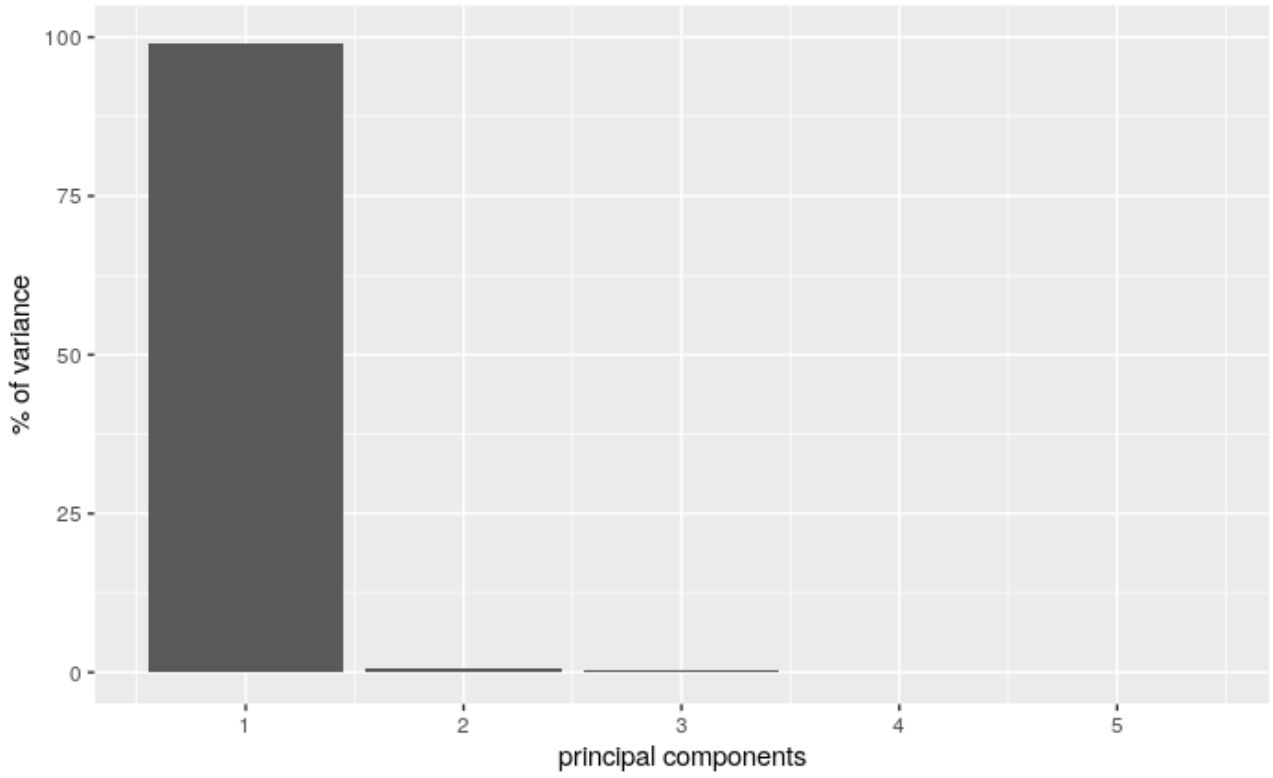

Figura 11. Porcentaje de varianza explicado por cada componente principal.

Mediante la representación gráfica de este análisis (figura 12), hemos comprobado que no existe una separación clara de las muestras por estado fisiológico o por individuo, aunque parece evidente que, tomando las muestras de cada individuo, éstas se separan de menor a mayor a lo largo del eje PC1 en los estados fisiológicos activo 1, activo 2 y letargo.

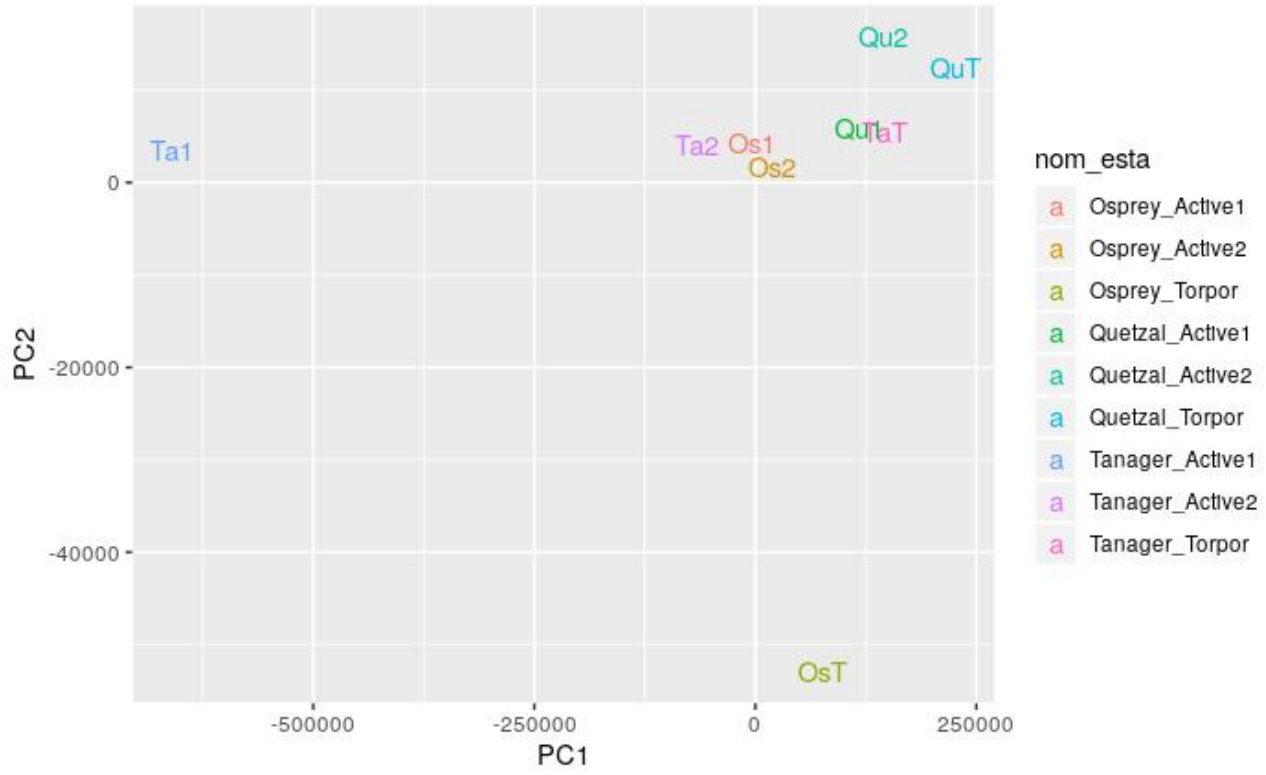

Figura 12. Análisis de componentes principales.

Las muestras de Quetzal se agrupan en la parte superior derecha del gráfico, sin embargo, las de Tanager se extienden a lo largo del eje PC1, mientras que las de Osprey lo hacen a lo largo del eje PC2. Encontramos que tanto la muestra de Tanager (activo 1) como la muestra de Osprey (letargo) parecen atípicas. Si estas muestras formaran parte de un estudio con un mayor número de réplicas, podrían ser eliminadas del análisis pero en este caso no lo hemos hecho para evitar la pérdida de poder estadístico.

Hemos estimado los niveles de expresión por pseudoalineamiento de las lecturas a las secuencias de los transcritos seleccionados y, posteriormente, hemos realizado el análisis de expresión diferencial de los genes con Sleuth. Esta herramienta permite determinar qué genes están diferencialmente expresados entre los pares de condiciones fisiológicas (activo 1 con activo 2, activo 2 con letargo, activo 1 con letargo) controlando la variabilidad propia de cada individuo.

El modelo reducido es el que se encarga de las variables que hay que tener en cuenta durante el análisis y lo hemos ajustado con el nombre de los animales (nombre). El modelo completo incluye todas las variables (nombre + estado). Aunque hemos comparado el modelo reducido con el modelo completo utilizando la prueba de máxima verosimilitud y la prueba de Wald, solo hemos tenido en cuenta los resultados obtenidos con esta última.

Tabla 7. Número de genes diferencialmente expresados entre pares de condiciones fisiológicas, agrupados en upregulated y downregulated.

	Genes diferencialmente expresados								
		Activo1		Activo2		Letargo		Activo	
Parejas de estados	Totales	*Upregulated*	*Downregulated*	*Upregulated*	*Downregulated*	*Upregulated*	*Downregulated*	*Upregulated*	*Downregulated*
Activo1 versus Letargo	155	96	59	-	-	59	96	-	-
Activo2 versus Letargo	89	-	-	60	29	29	60	-	-
Activo versus Letargo	117	-	-	-	-	38	79	79	38
Activo1 versus Activo2	14	8	6	6	8	-	-	-	-

Para controlar el *False Discovery Rate* se han considerado genes diferencialmente expresados aquellos con un valor $q \leq 0{,}05$.

La tabla 7 muestra un resumen del número de genes diferencialmente expresados descubiertos y si se encontraban más o menos expresados en la comparación.

De los 2.531 transcritos que componen el transcriptoma (1.720 genes distintos) hemos encontrado 14 genes diferencialmente expresados en el estado activo 1 comparado con el estado activo 2; 155 en la comparación del estado activo 1 con el estado letargo y 89 en la comparación del estado activo 2 con el estado letargo. Entre los genes diferencialmente expresados en la confrontación de los estados activos y letargo, hemos encontrado 33 transcritos coincidentes.

Asumiendo que los estados fisiológicos activo 1 y activo 2 son estados fisiológicos activos, también hemos realizado el análisis de expresión diferencial de genes suponiendo activo 1 y activo 2 como réplicas del mismo estado obteniendo 117 genes diferencialmente expresados en la comparación con el estado letargo.

Un análisis de enriquecimiento funcional con el sistema de clasificación Panther no revela ningún resultado estadísticamente significante entre los genes comparativamente más expresados durante el letargo. Sin embargo, entre los genes expresados en menor medida durante el letargo, sí encontramos un enriquecimiento positivo en procesos biológicos como la repuesta celular a aldehídos y a sustancias inorgánicas, pero también en actividades que se realizan en las mitocondrias como el transporte de electrones del citocromo c al oxígeno, el transporte de protones acoplado a la síntesis de ATP, el transporte de electrones de NADH a la ubiquinona y el ensamblaje del complejo I de la cadena respiratoria. Estos resultados nos hacen suponer que durante el letargo alguno de estos procesos se ven afectados (tabla 8).

Ningún proceso biológico se encontró sobrerrepresentado en los genes diferencialmente expresados entre activo 1 y activo 2.

Cuando comparamos los resultados obtenidos en los 606 transcritos comunes en los transcriptomas del actual análisis y el obtenido en el artículo de Faherty et al. [5], encontramos que entre los estados activos y el estado letargo existen 13 genes que son

considerados diferencialmente expresados en ambos análisis, 5 genes son considerados diferencialmente expresados únicamente en el análisis de Faherty et al. [5] y 65 genes son considerados diferencialmente expresados únicamente en el nuestro, llegando hasta 71 genes si incluimos la comparación del estado letargo con los estados activos tomados como replicas.

Atendiendo únicamente a los genes comunes de ambos transcriptomas que han sido considerados diferencialmente expresados entre el estado activo 1 y el letargo, existen 9 genes que han sido considerados diferencialmente expresados únicamente por Faherty et al. [5], 49 genes considerados diferencialmente expresados únicamente por nosotros y 6 genes en los que coinciden ambos análisis.

Cuando comparamos el estado activo 2 y el letargo encontramos 4 genes considerados diferencialmente expresados únicamente por Faherty et al. [5], 34 genes considerados diferencialmente expresados en el presente análisis y 5 genes coincidentes.

El único gen considerado diferencialmente expresado por Faherty et al. [5] en la comparación de los estados activo 1 y activo 2 no se encuentra entre los genes comunes de ambos transcriptomas, sin embargo, 8 de los 15 genes encontrados como diferencialmente expresados en nuestro estudio sí forman parte de este grupo.

Tabla 8. Resultados del análisis de enriquecimiento funcional de los genes expresados en menor proporción durante el letargo usando como referencia el genoma humano completo para procesos biológicos.

GO biological process complete	Homo sapiens - REFLIST (20996)	Client Text Box Input (135)	Client Text Box Input (expected)	Client Text Box Input (over/under)	Client Text Box Input (fold Enrichment)	Client Text Box Input (raw P-value)	Client Text Box Input (FDR)
cellular response to aldehyde (GO:0110096)	10	3	.06	+	46.66	6.96E-05	3.16E-02
mitochondrial electron transport, cytochrome c to oxygen (GO:0006123)	20	5	.13	+	38.88	4.74E-07	4.43E-04
aerobic electron transport chain (GO:0019646)	20	5	.13	+	38.88	4.74E-07	4.18E-04
mitochondrial ATP synthesis coupled proton transport (GO:0042776)	21	5	.14	+	37.03	5.83E-07	4.64E-04
ATP synthesis coupled proton transport (GO:0015986)	25	5	.16	+	31.11	1.24E-06	9.38E-04
energy coupled proton transport, down electrochemical gradient (GO:0015985)	25	5	.16	+	31.11	1.24E-06	8.95E-04
mitochondrial ATP synthesis coupled electron transport (GO:0042775)	90	14	.58	+	24.19	4.63E-15	1.84E-11
oxidative phosphorylation (GO:0006119)	116	18	.75	+	24.13	4.81E-19	7.66E-15
ATP synthesis coupled electron transport (GO:0042773)	91	14	.59	+	23.93	5.31E-15	1.41E-11
mitochondrial electron transport, NADH to ubiquinone (GO:0006120)	49	7	.32	+	22.22	6.64E-08	8.80E-05
ATP biosynthetic process (GO:0006754)	35	5	.23	+	22.22	5.43E-06	3.60E-03
respiratory electron transport chain (GO:0022904)	110	15	.71	+	21.21	2.72E-15	1.44E-11
cristae formation (GO:0042407)	31	4	.20	+	20.07	7.15E-05	3.16E-02
purine ribonucleoside triphosphate biosynthetic process (GO:0009206)	46	5	.30	+	16.90	1.83E-05	1.04E-02
purine nucleoside triphosphate biosynthetic process (GO:0009145)	47	5	.30	+	16.55	2.02E-05	1.11E-02
NADH dehydrogenase complex assembly (GO:0010257)	66	7	.42	+	16.50	4.26E-07	4.84E-04
mitochondrial respiratory chain complex I assembly (GO:0032981)	66	7	.42	+	16.50	4.26E-07	4.52E-04
aerobic respiration (GO:0009060)	80	8	.51	+	15.55	9.42E-08	1.15E-04
purine ribonucleoside triphosphate metabolic process (GO:0009205)	61	6	.39	+	15.30	4.41E-06	3.05E-03
ribonucleoside triphosphate biosynthetic process (GO:0009201)	52	5	.33	+	14.95	3.17E-05	1.68E-02
cellular respiration (GO:0045333)	160	15	1.03	+	14.58	4.19E-13	7.40E-10
electron transport chain (GO:0022900)	183	17	1.18	+	14.45	1.20E-14	2.73E-11

ribonucleoside triphosphate metabolic process (GO:0009199)	67	6	.43	+	13.93	7.29E-06	4.64E-03
ATP metabolic process (GO:0046034)	205	18	1.32	+	13.66	4.71E-15	1.50E-11
purine nucleoside triphosphate metabolic process (GO:0009144)	69	6	.44	+	13.52	8.53E-06	5.22E-03
mitochondrial respiratory chain complex assembly (GO:0033108)	99	8	.64	+	12.57	4.32E-07	4.29E-04
nucleoside triphosphate biosynthetic process (GO:0009142)	64	5	.41	+	12.15	8.00E-05	3.44E-02
energy derivation by oxidation of organic compounds (GO:0015980)	230	17	1.48	+	11.50	3.87E-13	7.70E-10
mitochondrial transmembrane transport (GO:1990542)	92	6	.59	+	10.14	3.96E-05	2.03E-02
nucleoside triphosphate metabolic process (GO:0009141)	93	6	.60	+	10.03	4.19E-05	2.08E-02
proton transmembrane transport (GO:1902600)	140	9	.90	+	10.00	4.97E-07	4.16E-04
generation of precursor metabolites and energy (GO:0006091)	419	24	2.69	+	8.91	8.10E-16	6.44E-12
mitochondrion organization (GO:0007005)	443	19	2.85	+	6.67	1.25E-10	1.98E-07
mitochondrial transport (GO:0006839)	217	8	1.40	+	5.73	1.01E-04	4.13E-02
oxidation-reduction process (GO:0055114)	964	26	6.20	+	4.19	6.92E-10	1.00E-06
drug metabolic process (GO:0017144)	497	12	3.20	+	3.76	1.06E-04	4.22E-02
response to inorganic substance (GO:0010035)	540	13	3.47	+	3.74	5.58E-05	2.61E-02
phosphorylation (GO:0016310)	1306	23	8.40	+	2.74	1.13E-05	6.66E-03
cellular metabolic process (GO:0044237)	7671	73	49.32	+	1.48	4.62E-05	2.23E-02
metabolic process (GO:0008152)	8428	77	54.19	+	1.42	9.39E-05	3.93E-02

2.4 Discusión

Se cree que la hibernación en mamíferos está condicionada por cambios en la expresión de genes, también presentes en mamíferos no hibernadores.

Con el fin de dilucidar qué genes están involucrados en la hibernación, en 2016 Faherty et al. realizaron una RNA-seq a partir de muestras del WAT recogidas en tres individuos, en tres condiciones fisiológicas diferentes [5].

El análisis de los datos de la RNA-seq requiere de un transcriptoma al cual alinear o pseudoalinear las lecturas para estimar la abundancia de cada transcrito, abundancia proporcional al número de lecturas alineadas o pseudoalineadas. En ausencia del genoma de referencia, Faherty et al. ensamblaron un transcriptoma *de novo* a partir de las lecturas de la RNA-seq [5]. Este método tiene ciertas limitaciones las cuales impiden el ensamblaje de un gran número de transcritos: se obtuvieron 9.540 aunque se cree que existen entre 25.000 y 30.000 genes que codifican proteínas en el genoma humano [82, 83].

Con la intención de obtener un transcriptoma que contuviera un mayor número de transcritos, hemos alineado las lecturas al genoma para, posteriormente, ensamblar los transcritos. El porcentaje de alineamiento general al ensamblaje ASM808673v1 del genoma del *Cheirogaleus medius* fue muy inferior al esperado debido, seguramente, a que este genoma fue secuenciado a partir del ADN de un individuo híbrido entre lémur enano de cola gruesa y una especie cercana. Sin embargo, del alineamiento de las lecturas al ensamblaje CheMed_v1_BIUU, hemos obtenido un porcentaje general acorde a las expectativas. Alrededor de un 99% de las lecturas se alinearon al genoma CheMed_v1_BIUU, pero solo se ensamblaron 2.531 transcritos, un número muy inferior a lo esperado e inferior a los 9.540 obtenidos en el ensamblado *de novo*.

Una posible explicación de este bajo rendimiento se encuentra en el ensamblado del genoma CheMed_v1_BIUU. Cuando se genera la secuencia de un genoma, las lecturas de ADN son primeramente ensambladas en *contigs*, lecturas que se solapan entre sí formando una secuencia consenso. Los *scaffolds* son creados encadenando *contigs* para conseguir una continuidad en la secuencia del genoma. Los *contigs* están separados por secuencias

desconocidas hecho que conlleva una pérdida de información genómica. Además las secuencias de los *scaffolds* que flanquean los espacios vacíos suelen ser de baja calidad, es decir, cuánto mayor es el numero de *contigs* y *scaffolds* en un genoma, más fragmentado se encuentra y más espacios en blanco contiene. El ensamblaje CheMed_v1_BIUU del genoma está formado por 547.715 *scaffolds* y 555.751 *contigs* lo que supone unas cifras muy elevadas teniendo en cuenta que el ensamblaje ASM808673v1 contiene 3.231 *scaffolds* y 130.903 *contigs* y que el genoma humano contiene únicamente 1.520 *scaffolds* y *contigs*. Este exceso de espacios vacíos en la secuencia del genoma CheMed_v1_BIUU podría ser el responsable de que ≥50% de los transcritos ensamblados supongan ≤10% de la longitud de su proteína homóloga en humanos.

A pesar del escaso número de transcritos anotados en el genoma, hemos conseguido ampliar la lista de genes diferencialmente expresados durante la hibernación del lémur enano de cola gruesa con 179 nuevos genes.

Además, hemos encontrado un enriquecimiento funcional en los genes expresados en menor proporción durante el letargo relacionado con la actividad mitocondrial que no fueron obtenidos en el estudio original [5]. Esta menor expresión de ciertos genes mitocondriales, sugiere una afectación en la actividad mitocondrial durante la hibernación que podría estar relacionada con la disminución en la Tc y la reducción en el consumo de oxígeno [84].

También hemos encontrado que todos los genes con actividad peptidasa expresados diferencialmente están regulados a la baja. Esto podría ser debido a que el proceso de transcripción se ve frenado durante el letargo [27] y sugiere que las células del WAT realizan un escaso remplazo de proteínas durante la hibernación, seguramente para ahorrar energía.

Dentro de este grupo de genes cabe destacar la proteína *GABA type A receptor-associated* (GABARAP-a) que está relacionada con la actividad autofágica en respuesta a la inanición [85]. La autofagia es un mecanismo celular que permite obtener recursos energéticos a partir de la degradación de los componentes celulares. En condiciones de inanición, como la hibernación, se esperaría ver sobreexpresado este gen, como ocurre universalmente cuando se cesa la aportación de alimentos. Sin embargo, se encuentra

regulado a la baja. Este comportamiento probablemente es debido a que los adipocitos disponen de una fuente de energía alternativa a la peptídica.

Otro gen con función peptidasa que merece ser mencionado es *Proteasome Subunit Alpha 3*, sobreexpresado en el corazón del oso negro durante la hibernación [86] pero poco expresado en el WAT del lémur enano de cola gruesa, seguramente por el mismo motivo que GABARAP-a.

La expresión de genes con actividad hidrolasa se ha visto aumentada durante la hibernación. Dentro de este grupo, *G-protein coupled receptor 33* actúa como señal intracelular en casos de asfixia [87] hecho que podría estar relacionado con la baja disponibilidad de oxígeno durante el letargo y *Proteinase-activated receptor 1* que desempeña alguna función en el desarrollo vascular permitiendo un transporte más eficiente de nutrientes desde el WAT.

Durante la hibernación, los mamíferos dejan de utilizar los carbohidratos como fuente principal de energía para utilizar la grasa almacenada en el WAT [21, 24, 55]. Esta afirmación coincide con la disminución en la expresión génica de los genes encargados de la síntesis de ácidos grasos (*Acyl-CoA synthetase short-chain family member 3 mitochondrial* o *Alpha-1,3-galactosyltransferase 2*) y con el aumento de la expresión de genes con actividad hidrolítica de ácidos grasos (*Sialidase-3*). Respecto al metabolismo de los carbohidratos, *Sorbitol dehydrogenase*, relacionada con la oxidación de varios alcoholes de azúcar, se encuentra expresada a la baja.

Los genes que expresan *Nuclear factor interleukin-3-regulated protein* y *SRR1-like protein* están relacionados con la regulación de los ritmos circadianos. La expresión diferencial de estos genes podría estar relacionada con la variación en la cantidad de horas de luz a la que son expuestos los individuos.

Muchos de los genes que hemos resuelto como diferencialmente expresados en este análisis fueron también analizados pero determinados como no expresados diferencialmente por Faherty et al. [5]. Esta aparente incongruencia entre los resultados de ambos análisis podría estar relacionada con la diferencia en la cantidad de transcritos que se han analizado ya que, cuanto mayor es el número de genes analizados en una comparación múltiple, más probabilidades existen de determinar genes diferencialmente expresados por casualidad y, por eso, el *p* valor ajustado es más restrictivo haciendo,

consecuentemente, más improbable determinar un gen como diferencialmente expresado.

3. Conclusiones

En la búsqueda de genes expresados diferencialmente en cualquier organismo es indispensable disponer en primer lugar, de los datos de la RNA-seq de las muestras de referencia y, en segundo lugar, de un transcriptoma de referencia en el cual esten representados los genes de interés.

Asimismo, el contenido en los datos de la RNA-seq dependen del individio y de su condición fisiológica así que un estudio longitudinal amplio y el establecimiento de unas condiciones estrictas en la extracción de las muestras son determinantes, más si no se dispone de un gran número de réplicas.

Del mismo modo, la cantidad de transcritos ensamblados, a partir de los datos de la RNA-seq, depende en gran medida de las características del ensamblaje del genoma de referencia y de la mínima degradación del ARN.

En la búsqueda de genes expresados diferencialmente de *Cheirogaleus medius* durante la hibernación, algunas de estas premisas no se han producido, hecho que se ha ido constatando a lo lago de la elaboración de este TFM.

Se selecionó el WAT debido a su papel fundamental en el metabolismo del lémur de cola gruesa de Madagascar. Sin embargo, de los 2.531 transcritos ensamblados obtenidos (≥70% de la longitud de su proteína homóloga en humanos) menos de 300 están relacionados con alguna actividad metabólica, lo que sugiere que el número de transcritos es reducido.

Se disponía de dos ensamblajes de referencia, uno híbrido y el otro con una cantidad considerable de espacios vacíos. En ambos casos, las secuencias de los transcritos ensamblados obtenidas eran mayoritariamente incompletas y representaban un porcentaje muy bajo respecto a las proteínas homólogas en humanos.

3.1 Cumplimiento de los objetivos y seguimiento en la planificación

En este contexto, han aparecido numerosos imprevistos que han hecho rediseñar tanto los objetivos como la planificación y el uso de herramientas.

Al principio, asumimos que el ensamblaje del genoma ASM808673v1 disponía de las anotaciones de los genes, como suele ser habitual. Conseguir un transcriptoma de referencia a partir de las anotaciones de un genoma es relativamente fácil y requiere de poco tiempo. Sin embargo, la ausencia de estas anotaciones nos obligó a asumir un nuevo objetivo específico: generar un transcriptoma a partir de los datos de la RNA-seq de las muestras.

Para alinear las lecturas al genoma, decidimos instalar una nueva herramienta, HISAT2, que supuso una ralentización considerable del ordenador portátil utilizado en este TFM. También requirió de una inversión de tiempo en el aprendizaje de la herramienta.

Tras obtener un reducido porcentaje de alineamientos generales con HISAT2, decidimos utilizar una segunda aplicación, Bowtie2. Aunque obtuvimos mejores resultados, todavía estaban muy por debajo de lo esperado, así que decidimos indagar sobre el origen del genoma. Mientras, ensamblamos los transcritos con StringTie y los anotamos por similitud utilizando BlastX. Obtuvimos un transcriptoma de 2.102 transcritos, un número muy inferior a los 9.540 conseguidos en el artículo de Faherty et al. [5]. Para entonces descubrimos que el ADN empleado en la secuenciación del genoma ASM808673v1 provenía de un individuo híbrido. Decidimos alinear las lecturas de una de las muestras al genoma CheMed_v1_BIUU con Bowtie2 que, aunque consumía una mayor cantidad de tiempo, había demostrado producir mejores resultados que HISAT2.

Dado el buen rendimiento obtenido, decidimos utilizar CheMed_v1_BIUU. A pesar de los buenos porcentajes de alineamientos generales obtenidos con este genoma, el número de transcritos obtenidos fue de 2.531.

En este paso es donde recuperamos los objetivos iniciales. Conseguir un transcriptoma ha supuesto una inversión de tiempo no prevista y el incumplimiento del segundo de los objetivos que consistía en valorar la relación entre los genes diferencialmente expresados en el lémur enano de cola gruesa y los diferencialmente expresados en otras especies durante la hibernación.

3.2 Perspectivas de futuro

El objetivo final del estudio de los perfiles de expresión génica durante la hibernación es comprender qué cambios a nivel celular están relacionados con los cambios fisiológicos producidos durante el letargo.

Se desconoce cuáles son los genes que condicionan estos cambios fisiológicos, por eso lo ideal sería poder estudiarlos todos, pero solo los genes que tienen representación en el transcriptoma de referencia son los analizados.

Para incrementar el número de genes representados en el transcriptoma sería necesario disponer de un genoma con una calidad que permita ensamblar los transcritos con mayor integridad, un mayor número de muestras por individuo o un mayor número de individuos.

Comparar los genes diferencialmente expresados durante la hibernación en el lémur enano de cola gruesa con los diferencialmente expresados en otras especies podría confirmar o desmentir la teoría que afirma que los mecanismos moleculares implicados en la hibernación son universales.

4. Glosario y siglas

***Ad libitum**:* a gusto, a voluntad.

ADN: ácido desoxirribonucleico. Nombre químico de la molécula que contiene la información genética en todos los seres vivos.

ADNc: ADN complementario. Molécula de ADN de doble cadena, en la que una de sus hebras constituye una secuencia totalmente complementaria al ARN mensajero a partir del cual se ha sintetizado.

Alinear datos: desplazar datos de suerte que tomen una posición particular respecto a uno o unos puntos de referencia, líneas o marcas del soporte de almacenamiento.

ARN: ácido ribonucleico. Molécula similar a la de ADN. A diferencia del ADN, el ARN es de cadena sencilla.

ARNm: ARN mensajero. Molécula de ácido ribonucleico que transmite la información genética del ADN para ser traducida durante la síntesis de proteínas.

ARNr: ARN ribosómico. Diferentes moléculas de ARN que forma parte de los ribosomas y es esencial para la síntesis proteica en todos los seres vivos.

ASCII: *American Standard Code for Information Interchange*. Código de caracteres que se basa en el alfabeto latino y sirve para codificar en bits y bytes los distintos caracteres alfanuméricos.

Circadiano, na: perteneciente o relativo a un período de aproximadamente 24 horas. Se aplica especialmente a ciertos fenómenos biológicos que ocurren rítmicamente alrededor de la misma hora, como la sucesión de vigilia y sueño.

Clado: agrupación que contiene un antepasado común y todos los descendientes (vivos y extintos) de ese antepasado.

Cuantificación: cálculo de cuántos fragmentos se unen a cada zona del genoma.

Depleción: disminución de cualquier líquido, en particular la sangre.

Endotermos: animales vertebrados que generan calor interno a través de la combustión química de los alimentos para mantener una temperatura corporal adecuada.

Ensamblaje *de novo*: proceso de descifrar la secuencia genómica a partir de pequeños fragmentos de ADN en el cual solo se utiliza la información obtenida de la secuenciación para reconstruir el genoma en cuestión, sin conocimiento *a priori* de la organización del mismo.

Ensamblar: unir, montar, convertir una codificación a lenguaje máquina.

Exoma: parte del genoma formado por los exones, los fragmentos de ADN que se transcriben para dar lugar a las proteínas.

Fenotipo: expresión del genotipo en función de un determinado ambiente.

Filogenia: historia evolutiva de un organismo o de un grupo de especies.

Fluoróforo: componente de una molécula que hace que ésta sea fluorescente. Grupo funcional de la molécula que absorberá energía de una longitud de onda específica y la volverá a emitir en otra determinada de mayor longitud de onda.

Formato FASTQ: formato con información sobre las secuencias biológicas y el correspondiente índice de calidad para cada base.

Heterotérmico, ca: de temperatura diferente.

Hiperfagia: aumento de la ingesta de comida.

Homeostasis: propiedad de una célula, individuo o población para mantener su equilibrio dinámico y, en caso de disturbio, restaurarlo por sus propios mecanismos reguladores.

Homeotérmico, ca: dícese de los animales cuya temperatura se mantiene constante, con independencia de la temperatura del ambiente.

Isoforma: una de las distintas formas de la misma proteína.

k-meros: todas las sub-secuencias de longitud k dentro de una secuencia.

Lenguaje R: lenguaje de programación con un enfoque al análisis estadísticos y gráficos.

Mapeado: asociación de cada fragmento de ADNc a la zona del genoma a la que corresponde.

Microarray de ADN: tecnología que permite el análisis comparativo y simultáneo de cómo se expresan cientos de genes en un solo experimento.

NGS: *Next-Generation Sequencing.* Conjunto de técnicas de secuenciación genética que mejoran el proceso de secuenciación original.

Phred: originalmente era un programa de *basecallin*g, pero ahora es sobretodo conocido como la escala del valor de calidad.

Pipeline: línea que implementa la tunelización de varios protocolos.

Poiquilotermia: incapacidad de regulación de la temperatura del cuerpo, por lo que esta varía de acuerdo con la temperatura ambiental.

Ritmo circanual: variación periódica de un parámetro biológico cuyo ciclo completo es un año.

RNA-seq: secuenciación de ácido ribonucleico. La aplicación de cualquier técnica de secuenciación de nueva generación al estudio de ARN.

Secuenciación *paired-end*: secuenciación de los dos extremos de un fragmento de ADN.

SRA: *Sequence Read Archive.* Base de datos de bioinformática que proporciona un repositorio público para los datos de secuenciación de ADN.

Transcriptoma: conjunto de moléculas de RNAm presentes en una célula, tejido u organismo en un estado determinado.

Transcrito: molécula de ARN obtenida a partir de una hebra complementaria de ADN.

WAT: tejido adiposo blanco. Compartimento de almacenamiento de energía principal del cuerpo de mamífero.

5. Bibliografía

1. GEISER, Fritz. Hibernation. *Current Biology*, 2013, vol. 23, no 5, p. R188-R193.
2. SRERE, Hilary K.; WANG, L. C.; MARTIN, Sandra L. Central role for differential gene expression in mammalian hibernation. *Proceedings of the National Academy of Sciences*, 1992, vol. 89, no 15, p. 7119-7123.
3. CONESA, Ana, et al. A survey of best practices for RNA-seq data analysis. *Genome biology*, 2016, vol. 17, no 1, p. 13.
4. DOS REIS, Mario, et al. Phylogenomic datasets provide both precision and accuracy in estimating the timescale of placental mammal phylogeny. *Proceedings of the Royal Society B: Biological Sciences*, 2012, vol. 279, no 1742, p. 3491-3500.
5. FAHERTY, Sheena L., et al. Gene expression profiling in the hibernating primate, Cheirogaleus medius. *Genome biology and evolution*, 2016, vol. 8, no 8, p. 2413-2426.
6. KRAUSE, David W., et al. Late Cretaceous terrestrial vertebrates from Madagascar: implications for Latin American biogeography. *Annals of the Missouri Botanical Garden*, 2006, p. 178-208.
7. POLDERMAN, Kees H. Application of therapeutic hypothermia in the ICU: opportunities and pitfalls of a promising treatment modality. Part 1: Indications and evidence. *Intensive care medicine*, 2004, vol. 30, no 4, p. 556-575.
8. CHRISTINA, G., et al. Ubiquitous and temperature-dependent neural plasticity in hibernators. *Journal of Neuroscience*, 2006, vol. 26, no 41, p. 10590-10598.
9. GARNER, C. C., et al. Synaptic protein dynamics in hibernation. *The Journal of neuroscience: the official journal of the Society for Neuroscience*, 2007, vol. 27, no 1, p. 84-92.
10. MAGARIÑOS, Ana María, et al. Rapid and reversible changes in intrahippocampal connectivity during the course of hibernation in

European hamsters. *Proceedings of the National Academy of Sciences*, 2006, vol. 103, no 49, p. 18775-18780.

11. CERRI, Matteo, et al. Hibernation for space travel: Impact on radioprotection. *Life sciences in space research*, 2016, vol. 11, p. 1-9.
12. ANDREWS, Simon. FastQC: a quality control tool for high throughput sequence data. Version 0.11. 2. *Babraham Institute, Cambridge, UK http://www. bioinformatics. babraham. ac. uk/projects/fastqc*, 2014.
13. BOLGER, Anthony M.; LOHSE, Marc; USADEL, Bjoern. Trimmomatic: a flexible trimmer for Illumina sequence data. *Bioinformatics*, 2014, vol. 30, no 15, p. 2114-2120.
14. LI, Bo; DEWEY, Colin N. RSEM: accurate transcript quantification from RNA-Seq data with or without a reference genome. *BMC bioinformatics*, 2011, vol. 12, no 1, p. 323.
15. ROBERTS, Adam; PACHTER, Lior. Streaming fragment assignment for real-time analysis of sequencing experiments. *Nature methods*, 2013, vol. 10, no 1, p. 71.
16. ANDERS, Simon; PYL, Paul Theodor; HUBER, Wolfgang. HTSeq—a Python framework to work with high-throughput sequencing data. *Bioinformatics*, 2015, vol. 31, no 2, p. 166-169.
17. BRAY, Nicolas L., et al. Near-optimal probabilistic RNA-seq quantification. *Nature biotechnology*, 2016, vol. 34, no 5, p. 525.
18. HUBER, Wolfgang, et al. Orchestrating high-throughput genomic analysis with Bioconductor. *Nature methods*, 2015, vol. 12, no 2, p. 115.
19. ANDREWS, Matthew T. Molecular interactions underpinning the phenotype of hibernation in mammals. *Journal of Experimental Biology*, 2019, vol. 222, no 2, p. jeb160606.
20. BARNES, Brian M. Freeze avoidance in a mammal: body temperatures below 0 degree C in an Arctic hibernator. *Science*, 1989, vol. 244, no 4912, p. 1593-1595.
21. CAREY, Hannah V.; ANDREWS, Matthew T.; MARTIN, Sandra L. Mammalian hibernation: cellular and molecular responses to depressed metabolism and low temperature. *Physiological reviews*, 2003, vol. 83, no 4, p. 1153-1181.

22. HELDMAIER, Gerhard; ORTMANN, Sylvia; ELVERT, Ralf. Natural hypometabolism during hibernation and daily torpor in mammals. *Respiratory physiology & neurobiology*, 2004, vol. 141, no 3, p. 317-329.

23. FAHERTY, Sheena L., et al. Transcriptomics in the wild: Hibernation physiology in free-ranging dwarf lemurs. *Molecular ecology*, 2018, vol. 27, no 3, p. 709-722.

24. BOYER, Bert B.; BARNES, Brian M. Molecular and metabolic aspects of mammalian hibernation: expression of the hibernation phenotype results from the coordinated regulation of multiple physiological and molecular events during preparation for and entry into torpor. *Bioscience*, 1999, vol. 49, no 9, p. 713-724.

25. RUF, Thomas; GEISER, Fritz. Daily torpor and hibernation in birds and mammals. *Biological Reviews*, 2015, vol. 90, no 3, p. 891-926.

26. VAN BREUKELEN, Frank; MARTIN, Sandra L. Translational initiation is uncoupled from elongation at 18 C during mammalian hibernation. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 2001, vol. 281, no 5, p. R1374-R1379.

27. VAN BREUKELEN, F.; MARTIN, S. Reversible depression of transcription during hibernation. *Journal of Comparative Physiology B*, 2002, vol. 172, no 5, p. 355-361.

28. PRENDERGAST, Brian J., et al. Periodic arousal from hibernation is necessary for initiation of immune responses in ground squirrels. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 2002, vol. 282, no 4, p. R1054-R1062.

29. D'ALESSANDRO, Angelo, et al. Comfortably numb and back: plasma metabolomics reveals biochemical adaptations in the hibernating 13-lined ground squirrel. *Journal of proteome research*, 2016, vol. 16, no 2, p. 958-969.

30. WIERSMA, Marit, et al. Torpor-arousal cycles in Syrian hamster heart are associated with transient activation of the protein quality control

system. *Comparative Biochemistry and Physiology Part B: Biochemistry and Molecular Biology*, 2018, vol. 223, p. 23-28.

31. MCKECHNIE, Andrew E.; MZILIKAZI, Nomakwezi. Heterothermy in Afrotropical mammals and birds: a review. *Integrative and comparative biology*, 2011, vol. 51, no 3, p. 349-363.
32. DAUSMANN, Kathrin H. Measuring body temperature in the field—evaluation of external vs. implanted transmitters in a small mammal. *Journal of Thermal Biology*, 2005, vol. 30, no 3, p. 195-202.
33. LYMAN, C. P. Willis JS, Malan A, Wang LCH. Hibernation and Torpor in Mammals and Birds. *New York: Academic*, 1982, p. 1-301.
34. WILSON, Allan C.; CARLSON, Steven S.; WHITE, Thomas J. Biochemical evolution. *Annual review of biochemistry*, 1977, vol. 46, no 1, p. 573-639.
35. BURNS, Stephen J., et al. Rapid human-induced landscape transformation in Madagascar at the end of the first millennium of the Common Era. *Quaternary Science Reviews*, 2016, vol. 134, p. 92-99.
36. WILLIAMS, Rachel C., et al. Conservation genomic analysis reveals ancient introgression and declining levels of genetic diversity in Madagascar's hibernating dwarf lemurs. *Heredity*, 2020, vol. 124, no 1, p. 236-251.
37. ORTMANN, S., et al. Spontaneous daily torpor in Malagasy mouse lemurs. *Naturwissenschaften*, 1997, vol. 84, no 1, p. 28-32.
38. SCHMID, Jutta. Daily torpor in free-ranging gray mouse lemurs (Microcebus murinus) in Madagascar. *International Journal of Primatology*, 2001, vol. 22, no 6, p. 1021-1031.
39. RUF, Thomas, et al. Hibernation in the pygmy slow loris (Nycticebus pygmaeus): multiday torpor in primates is not restricted to Madagascar. *Scientific reports*, 2015, vol. 5, p. 17392.
40. NOWACK, Julia; MZILIKAZI, Nomakwezi; DAUSMANN, Kathrin H. Torpor as an emergency solution in Galago moholi: heterothermy is triggered by different constraints. *Journal of Comparative Physiology B*, 2013, vol. 183, no 4, p. 547-556.

41. DAUSMANN, Kathrin H., et al. Physiology: hibernation in a tropical primate. *Nature*, 2004, vol. 429, no 6994, p. 825.

42. BLANCO, Marina B.; RAHALINARIVO, Vololonirina. First direct evidence of hibernation in an eastern dwarf lemur species (Cheirogaleus crossleyi) from the high-altitude forest of Tsinjoarivo, central-eastern Madagascar. *Naturwissenschaften*, 2010, vol. 97, no 10, p. 945-950.

43. BLANCO, Marina B., et al. Underground hibernation in a primate. *Scientific reports*, 2013, vol. 3, p. 1768.

44. GOODMAN, Steven M.; BENSTEAD, Jonathan P. *Natural history of Madagascar*. University of Chicago Press, 2003.

45. BLANCO, Marina B., et al. Discovery of sympatric dwarf lemur species in the high-altitude rain forest of Tsinjoarivo, eastern Madagascar: implications for biogeography and conservation. *Folia Primatologica*, 2009, vol. 80, no 1, p. 1-17.

46. BLANCO, Marina B., et al. Tropical heterothermy is “cool”: The expression of daily torpor and hibernation in primates. *Evolutionary Anthropology: Issues, News, and Reviews*, 2018, vol. 27, no 4, p. 147-161.

47. TØIEN, Øivind, et al. Hibernation in black bears: independence of metabolic suppression from body temperature. *Science*, 2011, vol. 331, no 6019, p. 906-909.

48. LOVEGROVE, Barry G.; LOBBAN, Kerileigh D.; LEVESQUE, Danielle L. Mammal survival at the Cretaceous–Palaeogene boundary: metabolic homeostasis in prolonged tropical hibernation in tenrecs. *Proceedings of the Royal Society B: Biological Sciences*, 2014, vol. 281, no 1796, p. 20141304.

49. DAUSMANN, K. H.; GLOS, J.; HELDMAIER, G. Energetics of tropical hibernation. *Journal of Comparative Physiology B*, 2009, vol. 179, no 3, p. 345-357.

50. FIETZ, Joanna; DAUSMANN, K. H. Big is beautiful: fat storage and hibernation as a strategy to cope with marked seasonality in the fat-tailed

dwarf lemur (Cheirogaleus medius). En *Lemurs*. Springer, Boston, MA, 2006. p. 97-110.

51. PERRET, M.; AUJARD, F. Daily hypothermia and torpor in a tropical primate: synchronization by 24-h light-dark cycle. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 2001, vol. 281, no 6, p. R1925-R1933.

52. PERRET, M.; AUJARD, F.; VANNIER, G. Influence of daylength on metabolic rate and daily water loss in the male prosimian primate Microcebus murinus. *Comparative Biochemistry and Physiology Part A: Molecular & Integrative Physiology*, 1998, vol. 119, no 4, p. 981-989.

53. GEISER, Fritz. Metabolic rate and body temperature reduction during hibernation and daily torpor. *Annu. Rev. Physiol.*, 2004, vol. 66, p. 239-274.

54. VAN BREUKELEN, Frank; MARTIN, Sandra L. The hibernation continuum: physiological and molecular aspects of metabolic plasticity in mammals. *Physiology*, 2015, vol. 30, no 4, p. 273-281.

55. BUCK, Michael J.; SQUIRE, Teresa L.; ANDREWS, Matthew T. Coordinate expression of the PDK4 gene: a means of regulating fuel selection in a hibernating mammal. *Physiological genomics*, 2002, vol. 8, no 1, p. 5-13.

56. WILSON, BRUCE E.; DEEB, S.; FLORANT, G. L. Seasonal changes in hormone-sensitive and lipoprotein lipase mRNA concentrations in marmot white adipose tissue. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 1992, vol. 262, no 2, p. R177-R181.

57. HERMINGHUYSEN, D., et al. Measurement and seasonal variations of black bear adipose lipoprotein lipase activity. *Physiology & behavior*, 1995, vol. 57, no 2, p. 271-275.

58. BOYER, Bert B., et al. Differential regulation of uncoupling protein gene homologues in multiple tissues of hibernating ground squirrels. *American*

Journal of Physiology-Regulatory, Integrative and Comparative Physiology, 1998, vol. 275, no 4, p. R1232-R1238.

59. BAUER, Vernon W., et al. Expression of a chimeric retroviral-lipase mRNA confers enhanced lipolysis in a hibernating mammal. *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 2001, vol. 281, no 4, p. R1186-R1192.

60. DEMAS, Gregory E., et al. Photoperiodic regulation of gene expression in brown and white adipose tissue of Siberian hamsters (Phodopus sungorus). *American Journal of Physiology-Regulatory, Integrative and Comparative Physiology*, 2002, vol. 282, no 1, p. R114-R121.

61. EDDY, Sean F.; STOREY, Kenneth B. Up-regulation of fatty acid-binding proteins during hibernation in the little brown bat, Myotis lucifugus. *Biochimica et Biophysica Acta (BBA)-Gene Structure and Expression*, 2004, vol. 1676, no 1, p. 63-70.

62. KABINE, Mostafa, et al. Peroxisome proliferator-activated receptors as regulators of lipid metabolism; tissue differential expression in adipose tissues during cold acclimatization and hibernation of jerboa (Jaculus orientalis). *Biochimie*, 2004, vol. 86, no 11, p. 763-770.

63. EDDY, Sean F.; MORIN, Pier; STOREY, Kenneth B. Cloning and expression of PPARγ and PGC-1α from the hibernating ground squirrel, Spermophilus tridecemlineatus. *Molecular and cellular biochemistry*, 2005, vol. 269, no 1, p. 175-182.

64. HAMPTON, Marshall; MELVIN, Richard G.; ANDREWS, Matthew T. Transcriptomic analysis of brown adipose tissue across the physiological extremes of natural hibernation. *PloS one*, 2013, vol. 8, no 12, p. e85157.

65. CHOW, Brian A., et al. Serum immune-related proteins are differentially expressed during hibernation in the American black bear. *PLoS One*, 2013, vol. 8, no 6, p. e66119.

66. SURGET-GROBA, Yann; MONTOYA-BURGOS, Juan I. Optimization of de novo transcriptome assembly from next-generation sequencing data. *Genome research*, 2010, vol. 20, no 10, p. 1432-1440.

67. PORRECA, Gregory J., et al. Multiplex amplification of large sets of human exons. *Nature methods*, 2007, vol. 4, no 11, p. 931.

68. MORTAZAVI, Ali, et al. Mapping and quantifying mammalian transcriptomes by RNA-Seq. *Nature methods*, 2008, vol. 5, no 7, p. 621.

69. ZERBINO, Daniel R.; BIRNEY, Ewan. Velvet: algorithms for de novo short read assembly using de Bruijn graphs. *Genome research*, 2008, vol. 18, no 5, p. 821-829.

70. SIMPSON, Jared T., et al. ABySS: a parallel assembler for short read sequence data. *Genome research*, 2009, vol. 19, no 6, p. 1117-1123.

71. KRYSTAL, Andrew D., et al. The relationship of sleep with temperature and metabolic rate in a hibernating primate. *PloS one*, 2013, vol. 8, no 9, p. e69914.

72. FIETZ, J., et al. White adipose tissue composition in the free-ranging fat-tailed dwarf lemur (Cheirogaleus medius; Primates), a tropical hibernator. *Journal of Comparative Physiology B*, 2003, vol. 173, no 1, p. 1-10.

73. FAHERTY, Sheena L., et al. Evaluating whole transcriptome amplification for gene profiling experiments using RNA-Seq. *BMC biotechnology*, 2015, vol. 15, no 1, p. 65.

74. WHEELER, David L., et al. Database resources of the national center for biotechnology information. *Nucleic acids research*, 2006, vol. 35, no suppl_1, p. D5-D12.

75. SIRÉN, J.; VÄLIMÄKI, N.; MÄKINEN, V. HISAT2-fast and sensitive alignment against general human population. *IEEE/ACM Trans Comput Biol Bioinforma*, 2014, vol. 11, p. 375-88.

76. KIM, Daehwan; LANGMEAD, Ben; SALZBERG, Steven L. HISAT: a fast spliced aligner with low memory requirements. *Nature methods*, 2015, vol. 12, no 4, p. 357.

77. LANGMEAD, Ben; SALZBERG, Steven L. Fast gapped-read alignment with Bowtie 2. *Nature methods*, 2012, vol. 9, no 4, p. 357.

78. PERTEA, Mihaela, et al. StringTie enables improved reconstruction of a transcriptome from RNA-seq reads. *Nature biotechnology*, 2015, vol. 33, no 3, p. 290.

79. CAMACHO, Christiam, et al. BLAST+: architecture and applications. *BMC bioinformatics*, 2009, vol. 10, no 1, p. 421.

80. HENSCHEL, Robert, et al. Trinity RNA-Seq assembler performance optimization. En *Proceedings of the 1st Conference of the Extreme Science and Engineering Discovery Environment: Bridging from the eXtreme to the campus and beyond*. ACM, 2012. p. 45.

81. PIMENTEL, Harold, et al. Differential analysis of RNA-seq incorporating quantification uncertainty. *Nature methods*, 2017, vol. 14, no 7, p. 687.

82. CAWLEY, Simon, et al. Unbiased mapping of transcription factor binding sites along human chromosomes 21 and 22 points to widespread regulation of noncoding RNAs. *Cell*, 2004, vol. 116, no 4, p. 499-509.

83. STORZ, Gisela. An expanding universe of noncoding RNAs. *Science*, 2002, vol. 296, no 5571, p. 1260-1263.

84. STAPLES, James F. Metabolic suppression in mammalian hibernation: the role of mitochondria. *Journal of Experimental Biology*, 2014, vol. 217, no 12, p. 2032-2036.

85. TACHIBANA, Shin-Ichiro, et al. The Autophagy-Related Protein GABARAP Is Induced during Overwintering in the Bean Bug (Hemiptera: Alydidae). *Journal of economic entomology*, 2019.

86. FEDOROV, Vadim B., et al. Modulation of gene expression in heart and liver of hibernating black bears (Ursus americanus). *BMC genomics*, 2011, vol. 12, no 1, p. 171.

87. COX-LIMPENS, Kimberly EM, et al. Fetal asphyctic preconditioning alters the transcriptional response to perinatal asphyxia. *BMC neuroscience*, 2014, vol. 15, no 1, p. 67.

6. Anexos

Búsqueda de genes expresados diferencialmente en el tejido adiposo de *Cheirogaleus medius* durante la hibernación utilizando su genoma de referencia

Israel Blasco Ruano

Introducción

Los datos que analizaremos a continuación pertenecen a la secuenciación de ácido ribonucleico (ARN) extraído de tejido adiposo blanco (WAT) en un grupo formado por cuatro lémures enanos de cola gruesa. La extracción del tejido se realizó en tres momentos distintos: octubre, diciembre y febrero.

- Octubre es el mes previo al comienzo de la estación invernal. En este periodo, el comportamiento de los lémures enanos de cola gruesa no se distingue, a grandes rasgos, del comportamiento de cualquier otro lémur no hibernador. Por tanto, el perfil de expresión génica en este mes debería ser homólogo al de cualquier animal no hibernador.
- Diciembre es el mes previo al letargo estacional y supone el periodo en el que los animales en libertad acumulan la mayor cantidad de grasa posible para superar el invierno. Cambios en la expresión génica respecto al período anterior podrían revelar cuáles son los genes involucrados en este comportamiento hiperfágico.
- Febrero es uno de los meses en el que estos animales permanecen en estado de hibernación. Cabría esperar un perfil de expresión génica muy diferente a los anteriores, que nos ayudara a comprender lo que sucede durante la hibernación a nivel celular.

Muestras y ajustes de entorno

Ajustes de entorno y herramientas

Antes de descargar los archivos y herramientas necesarios para comenzar el análisis, crearemos un conjunto de directorios de trabajo. Los archivos que contienen los datos de las muestras, del genoma, etc., los almacenaremos en el subdirectorio `datos`, los resultados obtenidos en `resultados` y las herramientas necesarias en `herramientas`.

```
mkdir datos resultados herramientas
```

Las herramientas necesarias para llevar a cabo este proyecto son:

- **SRA toolkit**: aplicación necesaria para obtener los archivos con las secuencias de las lecturas de las muestras que hay que analizar.
- **FastQC**: control de calidad de los datos crudos y comprobación de que las secuencias de los adaptadores de las lecturas han sido eliminadas correctamente.
- **MultiQC**: agrupa los resultados obtenidos por FastQC para ver la globalidad de los resultados obtenidos en el control de calidad en gráficos que contienen todos los datos.
- **Trimmomatic**: elimina las secuencias de los adaptadores y recorta las lecturas en los extremos para eliminar nucleótidos con baja probabilidad de ser correctos.
- **BowTie2**: alinea las lecturas de secuenciación al genoma de referencia.
- **SamTools**: sirve para explorar, procesar y manipular archivos en formato SAM y BAM.
- **StringTie**: agrupa las lecturas que se solapan en la alineación con este programa para crear un transcriptoma sobre el que contar las lecturas en cada transcrito para cada muestra.
- **BlastX**: herramienta que puntúa la similitud entre secuencias.
- **Trinity**: aplicación que compara la longitud de la secuencia de una proteína con la longitud de la secuencia del transcrito homólogo potencial.
- **Bedtools**: produce archivos de secuencias FASTA a partir del genoma en un archivo FASTA y las anotaciones de los transcritos en un archivo GFF.
- **Kallisto**: realiza la medición de la abundancia de cada transcrito.
- **Sleuth**: un paquete de R utilizado para la detección de genes expresados diferencialmente a partir de las estimaciones realizadas por Kallisto.

La obtención de estas herramientas y su posterior almacenamiento en el subdirectorio `herramientas`, se realizará con los comandos facilitados en los manuales correspondientes.

Para instalar **SRA toolkit** se utilizará el siguiente comando.

```
cd herramientas
wget 'ftp-trace.ncbi.nlm.nih.gov/sra/sdk/2.10.0/sratoolkit.2.10.0-centos_linux64.tar.gz'
tar -vzxf sratoolkit.2.10.0-centos_linux64.tar.gz
cd sratoolkit.2.10.0-centos_linux64/bin
mv * ../..
```

Para instalar **FastQC** se utilizará el siguiente comando.

```
cd herramientas
wget https://www.bioinformatics.babraham.ac.uk/projects/fastqc/fastqc_v0.11.8.zip
unzip fastqc_v0.11.8.zip
cd FastQC
mv * ..
```

Para instalar **MultiQC** se utilizará el siguiente comando.

```
pip install multiqc
```

Para instalar **Trimmomatic** se utilizará el siguiente comando.

```
cd herramientas
wget 'www.usadellab.org/cms/uploads/supplementary/Trimmomatic/Trimmomatic-0.39.zip'
unzip 'Trimmomatic-0.39.zip'
cd Trimmomatic-0.39
mv * ..
```

Para instalar **Bowtie2** se utilizará el siguiente comando.

```
cd herramientas
wget 'https://sourceforge.net/projects/bowtie-bio/files/bowtie2/2.3.5.1/bowtie2-2.3.5.1-linux-x86_64.zip'
unzip bowtie2-2.3.5.1-linux-x86_64.zip
cd bowtie2-2.3.5.1-linux-x86_64
mv * ..
```

Para instalar **Samtools** se utilizará el siguiente comando.

```
cd herramientas
mkdir src
git clone 'https://github.com/samtools/htslib'
git clone 'https://github.com/samtools/samtools'
cd samtools
make
mv samtools ../..
```

Para instalar **StringTie** se utilizará el siguiente comando.

```
cd herramientas
git clone 'https://github.com/gpertea/stringtie'
cd stringtie
make release
mv * ..
```

Para descargar la aplicación **BLAST** se utilizará el siguiente comando.

```
cd herramientas
wget 'ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/2.9.0/ncbi-blast-2.9.0+-x64-linux.tar.gz'
tar -vzxf ncbi-blast-2.9.0+-x64-linux.tar.gz
cd ncbi-blast-2.9.0+/bin
mv * ../..
```

Para instalar **Trinity** se utilizará el siguiente comando.

```
cd herramientas
wget 'https://github.com/trinityrnaseq/trinityrnaseq/releases/download/v2.8.6/trinityrnaseq-v2.8.6.FULL.tar.gz'
tar -vzxf trinityrnaseq-v2.8.6.FULL.tar.gz
```

Para instalar **Bedtools** se utilizará el siguiente comando.

```
cd herramientas
wget https://github.com/arq5x/bedtools2/releases/download/v2.29.1/bedtools-2.29.1.tar.gz
tar -zxvf bedtools-2.29.1.tar.gz
cd bedtools2
make
```

Para instalar **Kallisto** se utilizará el siguiente comando.

```
cd herramientas
conda install kallisto
```

Para instalar Sleuth se utilizará el siguiente comando.

```
source("http://bioconductor.org/biocLite.R")
biocLite("rhdf5")

install.packages("devtools")

devtools::install_github("pachterlab/sleuth")
```

Descarga de los archivos necesarios

Los archivos que contienen los datos de la secuenciación de ARN de cada una de las muestras son públicos y se encuentran disponibles en la página web de National Center for Biotechnology Information (NCBI). Este conjunto de datos pertenece al proyecto PRJNA264073 y consiste en doce archivos de lecturas *paired-end* (PE).

El nombre de los archivos de las muestras, junto con toda su información, se encuentra en la página web:

https://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP049327&o=acc_s%3Aa.

Para descargar estos archivos, se utilizará **SRA Toolkit**. El siguiente comando sirve para descargar los archivos, validarlos, convertirlos en formato fastq y dividirlos en un archivo *forward* (_1) y otro *reverse* (_2). Estos archivos se almacenarán en el directorio `datos`.

```
muestras=("SRR1634146" "SRR1634147" "SRR1634148" "SRR1634149" "SRR1634152" "SRR1634169"
"SRR1634266" "SRR1634268"
"SRR1634269" "SRR1634270" "SRR1634272" "SRR1634454")
cd datos
for i in ${muestras[*]}
do
  prefetch $i && vdb-validate $i && fastq-dump --split-files $i
done
```

Se descargará la secuencia del genoma ASM808673v1 de *Cheirogaleus medius* del *File Transfer Protocol* (FTP) del NCBI y se almacenará en la carpeta `datos`.

```
cd datos
wget
'ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/008/086/735/GCA_008086735.1_ASM808673v1/GCA_00808673
5.1_ASM808673v1_genomic.fna.gz'
gzip -d GCA_008086735.1_ASM808673v1_genomic.fna.gz
```

La secuencia del genoma CheMed_v1_BIUU de *Cheirogaleus medius* lo descargaremos del sitio FTP de NCBI y lo guardaremos en la carpeta `datos`.

```
cd datos
wget
'ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCA/004/024/725/GCA_004024725.1_CheMed_v1_BIUU/GCA_00402
4725.1_CheMed_v1_BIUU_genomic.fna.gz'
gzip -d GCA_004024725.1_CheMed_v1_BIUU_genomic.fna.gz
```

Se descargarán las secuencias de aminoácidos de las proteínas humanas (GRCh38.p13).

```
cd datos
wget
ftp://ftp.ncbi.nlm.nih.gov/genomes/all/GCF/000/001/405/GCF_000001405.39_GRCh38.p13/GCF_000001405
.39_GRCh38.p13_protein.faa.gz
```

Instalación y carga de paquetes de R

Además de las herramientas que se utilizarán en la línea de comandos, será necesario instalar y cargar los siguientes paquetes de R.

```
libraries <- c("class", "gmodels",  "knitr", "ggplot2")
check.libraries <- is.element(libraries, installed.packages()[, 1])==FALSE
libraries.to.install <- libraries[check.libraries]
if (length(libraries.to.install!=0)) {
  install.packages(libraries.to.install)
}

libraries_bioconductor <- c("Biostrings", "GenomicRanges", "sleuth")
check.libraries_bioconductor <- is.element(libraries_bioconductor, installed.packages()[,
1])==FALSE
libraries.to.install_bioconductor <- libraries_bioconductor[check.libraries_bioconductor]
if (length(libraries.to.install_bioconductor!=0)) {
  if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
  BiocManager::install(libraries.to.install_bioconductor)
}

success <- sapply(c(libraries, libraries_bioconductor),require, quietly = FALSE,  character.only
= TRUE)
```

```
if(length(success) != length(c(libraries, libraries_bioconductor))) {stop("A package failed to
return a success in require() function.")}
```

Definición del experimento

A través de la siguiente tabla, se definirán todos los archivos `fastq` y las comparaciones de las muestras.

```
Código <- c("SRR1634146", "SRR1634147", "SRR1634148", "SRR1634149", "SRR1634152", "SRR1634169",
"SRR1634266",
            "SRR1634268", "SRR1634269", "SRR1634270", "SRR1634272", "SRR1634454")
Muestra <- c(paste("CM",seq(19,25), sep = ""),paste("CM",seq(27,30), sep = ""), paste("CM", 26,
sep = ""))
Género <- c(rep(c("M","F"),4), "F", "M", "M", "F")
Estado <- c(rep("Active1", 4), rep("Active2", 3), rep("Torpor", 4), "Active2")
Fecha <- c(rep("Octubre", 4), rep("Diciembre", 3), rep("Febrero", 4), "Diciembre")
Animal <- c("Tanager", "Quetzal", "Osprey", "Towhee", "Osprey", "Towhee", "Tanager", "Towhee",
"Quetzal",
            "Tanager", "Osprey",  "Quetzal")
targets<-data.frame(Código , Muestra , Género , Estado, Fecha, Animal, stringsAsFactors = F)
rownames(targets)<-1:12
kable(targets)
# write.table(targets, "datos/targets.txt", sep="\t")
```

Preprocesado de las lecturas

Control de calidad de los datos crudos

Una forma sencilla de hacer un control de calidad en datos de RNA-seq antes de su procesamiento puede llevarse a cabo con **FastQC**. Esta herramienta realiza un análisis que ofrece una impresión rápida de problemas que deben tenerse en cuenta antes de realizar cualquier análisis posterior. Se almacenarán los archivos con los datos del análisis en `resultados`.

```
mkdir resultados/FastQC_Crudos
cd resultados/FastQC_Crudos
muestras_1_2=("SRR1634146_1.fastq" "SRR1634147_1.fastq" "SRR1634148_1.fastq"
"SRR1634149_1.fastq"
"SRR1634152_1.fastq" "SRR1634169_1.fastq" "SRR1634266_1.fastq" "SRR1634268_1.fastq"
"SRR1634269_1.fastq"
"SRR1634270_1.fastq" "SRR1634272_1.fastq" "SRR1634454_1.fastq" "SRR1634146_2.fastq"
"SRR1634147_2.fastq"
"SRR1634148_2.fastq" "SRR1634149_2.fastq" "SRR1634152_2.fastq" "SRR1634169_2.fastq"
"SRR1634266_2.fastq"
"SRR1634268_2.fastq" "SRR1634269_2.fastq" "SRR1634270_2.fastq" "SRR1634272_2.fastq"
"SRR1634454_2.fastq")
for i in ${muestras_1_2[*]}
do
  fastqc ../../datos/$i
done
```

Se descomprimirán los archivos creados por **FastQC**.

```
cd resultados/FastQC_Crudos
unzip '*fastqc.zip'
```

Veamos los gráficos del análisis de control de calidad de **FastQC**.

```
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634146_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634146_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634147_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634147_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634148_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634148_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634149_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634149_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634152_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634152_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634169_1_fastqc/Images",
```

```
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634169_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634266_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634266_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634268_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634268_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634269_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634269_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634270_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634270_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634272_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634272_2_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634454_1_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Crudos/SRR1634454_2_fastqc/Images",
pattern = ".png", full.names = T ))
```

Se podrán incluir todos los datos de control de calidad obtenidos por **FastQC** en un solo informe HTML utilizando **MultiQC**.

```
multiqc resultados/FastQC_Crudos
mv *multiqc* resultados
```

Para ver dicho informe, se clicará aquí y se abrirá en el navegador.

Filtración y recorte de los extremos de las lecturas

Para eliminar los adaptadores utilizados para la secuenciación, se utilizará **Trimmomatic.** Establecemos los siguientes tres parámetros de filtración:

- Si las tres primeras o las tres últimas bases de una lectura tienen una puntuación Phred inferior a 30, serán eliminadas.
- Si la calidad media de cuatro nucleótidos consecutivos empezando por el extremo 5' desciende por debajo de una puntuación Phred de 20, la lectura será cortada y la mitad que contenía el extremo 3' será eliminada del archivo FASTQ.
- Si las secuencias de los adaptadores están presentes en las lecturas serán eliminados cuando sean similares a las secuencias que contienen el archivo TruSeq3-PE.fa. Este archivo contiene las secuencias de los adaptadores que son insertados en los fragmentos de ADN necesarios para la secuenciación. Se eliminarán de las lecturas las secuencias que obtengan una puntuación de similaridad por encima de 30 en las lecturas palindrómicas (cuando la lectura apareada también contenía el adaptador) y por encima de 10 si no lo son.

```
mkdir resultados/Trimmomatic
cd resultados/Trimmomatic
muestras=("SRR1634146" "SRR1634147" "SRR1634148" "SRR1634149" "SRR1634152" "SRR1634169"
"SRR1634266" "SRR1634268"
"SRR1634269" "SRR1634270" "SRR1634272" "SRR1634454")
for i in ${muestras[*]}
do
  java -jar ../../herramientas/Trimmomatic-0.39/trimmomatic-0.39.jar PE ../../datos/$i\_1.fastq
../../datos/$i\_2.fastq $i\_f_p.fastq $i\_f_u.fastq $i\_r_p.fastq $i\_r_u.fastq
ILLUMINACLIP:../../herramientas/Trimmomatic-0.39/adapters/TruSeq3-PE-2.fa:2:30:10 LEADING:3
TRAILING:3 SLIDINGWINDOW:4:20 MINLEN:50
done
```

Para comprobar los cambios que ha producido el algoritmo **Trimmomatic** en la calidad de las lecturas, se utilizará **FastQC.**

```
mkdir resultados/FastQC_Trimmomatic
cd resultados/FastQC_Trimmomatic
muestras_f_r=("SRR1634146_f_p.fastq" "SRR1634147_f_p.fastq" "SRR1634148_fp.fastq"
"SRR1634149_f_p.fastq"
"SRR1634152_f_p.fastq" "SRR1634169_f_p.fastq" "SRR1634266_f_p.fastq" "SRR1634268_f_p.fastq"
"SRR1634269_f_p.fastq"
"SRR1634270_f_p.fastq" "SRR1634272_f_p.fastq" "SRR1634454_f_p.fastq" "SRR1634146_r_p.fastq"
"SRR1634147_r_p.fastq"
"SRR1634148_r_p.fastq" "SRR1634149_r_p.fastq" "SRR1634152_r_p.fastq" "SRR1634169_r_p.fastq"
```

```
"SRR1634266_r_p.fastq"
"SRR1634268_r_p.fastq" "SRR1634269_r_p.fastq" "SRR1634270_r_p.fastq" "SRR1634272_r_p.fastq"
"SRR1634454_r_p.fastq"
"SRR1634146_f_u.fastq" "SRR1634147_f_u.fastq" "SRR1634148_f_u.fastq" "SRR1634149_f_u.fastq"
"SRR1634152_f_u.fastq"
"SRR1634169_f_u.fastq" "SRR1634266_f_u.fastq" "SRR1634268_f_u.fastq" "SRR1634269_f_u.fastq"
"SRR1634270_f_u.fastq"
"SRR1634272_f_u.fastq" "SRR1634454_f_u.fastq" "SRR1634146_r_u.fastq" "SRR1634147_r_u.fastq"
"SRR1634148_r_u.fastq"
"SRR1634149_r_u.fastq" "SRR1634152_r_u.fastq" "SRR1634169_r_u.fastq" "SRR1634266_r_u.fastq"
"SRR1634268_r_u.fastq"
"SRR1634269_r_u.fastq" "SRR1634270_r_u.fastq" "SRR1634272_r_u.fastq" "SRR1634454_r_u.fastq")
for i in ${muestras_f_r[*]}
do
  fastqc ../Trimmomatic/$i
done
```

Se descomprimirán los archivos creados por **FastQC**.

```
cd resultados/FastQC_Trimmomatic
unzip '*fastqc.zip'
```

Veamos los gráficos del análisis de control de calidad de **FastQC** tras la filtración con **Trimmomatic**.

```
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634146_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634146_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634147_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634147_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634148_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634148_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634149_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634149_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634152_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634152_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634169_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634169_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634266_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634266_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634268_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634268_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634269_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634269_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634270_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634270_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634272_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634272_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634454_f_p_fastqc/Images",
pattern = ".png", full.names = T ))
include_graphics(list.files(path = "resultados/FastQC_Trimmomatic/SRR1634454_r_p_fastqc/Images",
pattern = ".png", full.names = T ))
```

Alineamientos

Mapeado de lecturas con Bowtie2

Tras comprobar que la herramienta **HISAT2** produce un rendimiento muy bajo de alineamientos, se opta por utilizar **BowTie2**. Se crea un índice para el genoma de referencia.

Índice con el genoma **ASM808673v1**.

```
mkdir resultados/ASM808673v1
bowtie2-build datos/GCA_008086735.1_ASM808673v1_genomic.fna resultados/ASM808673v1/GCAbowtie
```

Índice con el genoma **CheMed_v1_BIUU**.

```
mkdir resultados/CheMed_v1_BIUU
bowtie2-build datos/GCA_004024725.1_CheMed_v1_BIUU_genomic.fna resultados/CheMed_v1_BIUU/genoma2
```

Se alinean las lecturas de cada muestra contra el genoma de referencia.

Contra el genoma **ASM808673v1**.

```
muestras=("SRR1634146" "SRR1634147" "SRR1634148" "SRR1634149" "SRR1634152" "SRR1634169"
"SRR1634266" "SRR1634268"
"SRR1634269" "SRR1634270" "SRR1634272" "SRR1634454")
for i in ${muestras[*]}
do
  bowtie2 -p 2 -x resultados/ASM808673v1/GCAbowtie -1 resultados/Trimmomatic/$i\_f_p.fastq -2
resultados/Trimmomatic/$i\_r_p.fastq -S  resultados/ASM808673v1/$i\_bowtie2_sensitive.sam --met-
file Bowtie2_metrics_sensitive.txt --sensitive-local 2>
resultados/ASM808673v1/$i\very_sensitive_file.log
done
```

Contra el genoma **CheMed_v1_BIUU**.

```
muestras=("SRR1634146" "SRR1634147" "SRR1634148" "SRR1634149" "SRR1634152" "SRR1634169"
"SRR1634266" "SRR1634268"
"SRR1634269" "SRR1634270" "SRR1634272" "SRR1634454")
for i in ${muestras[*]}
do
  bowtie2 -p 2 -x resultados/CheMed_v1_BIUU/genoma2 -1 resultados/Trimmomatic/$i\_f_p.fastq -2
resultados/Trimmomatic/$i\_r_p.fastq -S
resultados/CheMed_v1_BIUU/$i\_bowtie2_very_sensitive_g2.sam --met-file
resultados/CheMed_v1_BIUU/Bowtie2_metrics_very_sensitive.txt --very-sensitive-local 2>
resultados/CheMed_v1_BIUU/$i\_very_sensitive_file.log
done
```

Veamos los resultados comparados de los alineamientos en los dos genomas.

```
# Resultados ASM808673v1
alineamientosI <- list()
for (i in targets$Código) {
  alineamientosI[[i]] <- read.delim(paste("resultados/ASM808673v1/", i,
"very_sensitive_file.log", sep = ""), header=FALSE, stringsAsFactors = F)
}

reads <- c()
paired <- c()
p_aligned_concordantly_0_times <- c()
p_aligned_concordantly_exactly_1_time <- c()
p_aligned_concordantly_more_1_times <- c()
p_aligned_discordantly_1_time <- c()
p_aligned_0_times_concordantly_or_discordantly <- c()
mates_make_up_the_pairs <- c()
m_aligned_0_times <- c()
m_aligned_exactly_1_time <- c()
m_aligned_more_1_times <- c()
overall_alignment_rate <- c()
for (i in 1:12) {
  reads[i] <- unlist(strsplit(alineamientosI[[i]][1,], " "))[1]
  paired[i] <- paste(unlist(strsplit(alineamientosI[[i]][2,], " "))[3:4], collapse = "")
  p_aligned_concordantly_0_times[i] <- paste(unlist(strsplit(alineamientosI[[i]][3,], "
"))[5:6], collapse = "")
  p_aligned_concordantly_exactly_1_time[i] <- paste(unlist(strsplit(alineamientosI[[i]][4,], "
"))[5:6], collapse = "")
  p_aligned_concordantly_more_1_times[i] <- paste(unlist(strsplit(alineamientosI[[i]][5,], "
"))[5:6], collapse = "")
  p_aligned_discordantly_1_time[i] <- paste(unlist(strsplit(alineamientosI[[i]][8,], " "))[7:8],
collapse = "")
  p_aligned_0_times_concordantly_or_discordantly[i]  <-
```

```
unlist(strsplit(alineamientosI[[i]][10,], " "))[5]
  mates_make_up_the_pairs[i] <- unlist(strsplit(alineamientosI[[i]][11,], " "))[7]
  m_aligned_0_times[i] <- paste(unlist(strsplit(alineamientosI[[i]][12,], " "))[9:10], collapse
= "")
  m_aligned_exactly_1_time[i] <- paste(unlist(strsplit(alineamientosI[[i]][13,], " "))[9:10],
collapse = "")
  m_aligned_more_1_times[i] <- paste(unlist(strsplit(alineamientosI[[i]][14,], " "))[9:10],
collapse = "")
  overall_alignment_rate[i] <- unlist(strsplit(alineamientosI[[i]][15,], " "))[1]
}
ASM808673v1 <- data.frame(reads, paired,
  p_aligned_concordantly_0_times, p_aligned_concordantly_exactly_1_time,
p_aligned_concordantly_more_1_times,
  p_aligned_discordantly_1_time, p_aligned_0_times_concordantly_or_discordantly,
mates_make_up_the_pairs,
  m_aligned_0_times, m_aligned_exactly_1_time, m_aligned_more_1_times, overall_alignment_rate,
stringsAsFactors = F)

rownames(ASM808673v1) <- targets$Código


# Resultados alineamiento CheMed_v1_BIUU
alineamientosII <- list()
for (i in targets$Código) {
  alineamientosII[[i]] <- read.delim(paste("resultados/CheMed_v1_BIUU/", i,
"_very_sensitive_file.log", sep = ""), header=FALSE, stringsAsFactors = F)
}

reads <- c()
paired <- c()
p_aligned_concordantly_0_times <- c()
p_aligned_concordantly_exactly_1_time <- c()
p_aligned_concordantly_more_1_times <- c()
p_aligned_discordantly_1_time <- c()
p_aligned_0_times_concordantly_or_discordantly <- c()
mates_make_up_the_pairs <- c()
m_aligned_0_times <- c()
m_aligned_exactly_1_time <- c()
m_aligned_more_1_times <- c()
overall_alignment_rate <- c()
for (i in 1:12) {
  reads[i] <- unlist(strsplit(alineamientosII[[i]][1,], " "))[1]
  paired[i] <- paste(unlist(strsplit(alineamientosII[[i]][2,], " "))[3:4], collapse = "")
  p_aligned_concordantly_0_times[i] <- paste(unlist(strsplit(alineamientosII[[i]][3,], "
"))[5:6], collapse = "")
  p_aligned_concordantly_exactly_1_time[i] <- paste(unlist(strsplit(alineamientosII[[i]][4,], "
"))[5:6], collapse = "")
  p_aligned_concordantly_more_1_times[i] <- paste(unlist(strsplit(alineamientosII[[i]][5,], "
"))[5:6], collapse = "")
  p_aligned_discordantly_1_time[i] <- paste(unlist(strsplit(alineamientosII[[i]][8,], "
"))[7:8], collapse = "")
  p_aligned_0_times_concordantly_or_discordantly[i]  <-
unlist(strsplit(alineamientosII[[i]][10,], " "))[5]
  mates_make_up_the_pairs[i] <- unlist(strsplit(alineamientosII[[i]][11,], " "))[7]
  m_aligned_0_times[i] <- paste(unlist(strsplit(alineamientosII[[i]][12,], " "))[9:10], collapse
= "")
  m_aligned_exactly_1_time[i] <- paste(unlist(strsplit(alineamientosII[[i]][13,], " "))[9:10],
collapse = "")
  m_aligned_more_1_times[i] <- paste(unlist(strsplit(alineamientosII[[i]][14,], " "))[9:10],
collapse = "")
  overall_alignment_rate[i] <- unlist(strsplit(alineamientosII[[i]][15,], " "))[1]
}

CheMed_v1_BIUU <- data.frame(reads, paired,
  p_aligned_concordantly_0_times, p_aligned_concordantly_exactly_1_time,
p_aligned_concordantly_more_1_times,
  p_aligned_discordantly_1_time, p_aligned_0_times_concordantly_or_discordantly,
mates_make_up_the_pairs,
  m_aligned_0_times, m_aligned_exactly_1_time, m_aligned_more_1_times, overall_alignment_rate,
stringsAsFactors = F)

rownames(CheMed_v1_BIUU) <- targets$Código

# Tablas con los alineamientos de las lecturas a los dos genomas
```

```
write.csv( ASM808673v1, "resultados/bowtie2_ASM808673v1.txt")
write.csv( CheMed_v1_BIUU, "resultados/bowtie2_CheMed_v1_BIUU.txt")
```

A continuación se preparan los datos para el *boxplot* de las lecturas apareadas alineadas concordantemente solo una vez.

```
# CheMed_v1_BIUU
pce1_CheMed_v1_BIUU<- c()
for (i in 1:12) {
  pce1_CheMed_v1_BIUU[i] <-
paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$p_aligned_concordantly_exactly_1_time,
"%")[[i]][1], ""))[9:13], collapse = "")
}
# ASM808673v1
pce1_ASM808673v1<- c()
for (i in 1:11) {
  pce1_ASM808673v1[i] <-
paste(unlist(strsplit(strsplit(ASM808673v1$p_aligned_concordantly_exactly_1_time, "%")[[i]][1],
""))[9:13], collapse = "")
}
pce1_ASM808673v1[12] <-
paste(unlist(strsplit(strsplit(ASM808673v1$p_aligned_concordantly_exactly_1_time, "%")[[12]][1],
""))[10:14], collapse = "")
```

A continuación se preparan los datos para el *boxplot* de las lecturas apareadas alineadas concordantemente cero veces.

```
# CheMed_v1_BIUU
pc0_CheMed_v1_BIUU<- c()
for (i in c(1:10, 12)) {
  pc0_CheMed_v1_BIUU[i] <-
paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$p_aligned_concordantly_0_times, "%")[[i]][1],
""))[9:12], collapse = "")
}
pc0_CheMed_v1_BIUU[11] <-
paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$p_aligned_concordantly_0_times, "%")[[11]][1],
""))[8:11], collapse = "")

# ASM808673v1
pc0_ASM808673v1<- c()
for (i in c(2, 3,5:12)) {
  pc0_ASM808673v1[i] <-
paste(unlist(strsplit(strsplit(ASM808673v1$p_aligned_concordantly_0_times, "%")[[i]][1],
""))[9:13], collapse = "")
}
for (i in c(1, 4)) {
  pc0_ASM808673v1[i] <-
paste(unlist(strsplit(strsplit(ASM808673v1$p_aligned_concordantly_0_times, "%")[[i]][1],
""))[10:14], collapse = "")
}
```

A continuación se preparan los datos para el *boxplot* de las lecturas apareadas alineadas concordantemente más de una vez.

```
# CheMed_v1_BIUU
pcm1_CheMed_v1_BIUU<- c()
for (i in 1:7) {
  pcm1_CheMed_v1_BIUU[i] <-
paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$p_aligned_concordantly_more_1_times, "%")[[i]][1],
""))[10:14], collapse = "")
}

for (i in 8:12) {
  pcm1_CheMed_v1_BIUU[i] <-
paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$p_aligned_concordantly_more_1_times, "%")[[i]][1],
""))[9:13], collapse = "")
}

# ASM808673v1
pcm1_ASM808673v1<- c()
for (i in 1:12) {
  pcm1_ASM808673v1[i] <-
paste(unlist(strsplit(strsplit(ASM808673v1$p_aligned_concordantly_more_1_times, "%")[[i]][1],
""))[9:13], collapse = "")
}
```

A continuación se preparan los datos para el *boxplot* de las lecturas apareadas alineadas discordantemente.

```
# CheMed_v1_BIUU
pd1_CheMed_v1_BIUU<- c()
for (i in 1:12) {
  pd1_CheMed_v1_BIUU[i] <-
paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$p_aligned_discordantly_1_time, "%")[[i]][1],
""))[8:12], collapse = "")
}

# ASM808673v1
pd1_ASM808673v1<- c()
for (i in 1:12) {
  pd1_ASM808673v1[i] <-
paste(unlist(strsplit(strsplit(ASM808673v1$p_aligned_discordantly_1_time, "%")[[i]][1],
""))[8:11], collapse = "")
}
```

A continuación se preparan los datos para el *boxplot* de las lecturas no apareadas alineadas cero veces.

```
# CheMed_v1_BIUU
m0_CheMed_v1_BIUU<- c()
for (i in c(1:12)) {
  m0_CheMed_v1_BIUU[i] <- paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$m_aligned_0_times,
"%")[[i]][1], ""))[8:12], collapse = "")
}

# ASM808673v1
m0_ASM808673v1<- c()
for (i in c(3, 8:12)) {
  m0_ASM808673v1[i] <- paste(unlist(strsplit(strsplit(ASM808673v1$m_aligned_0_times,
"%")[[i]][1], ""))[9:13], collapse = "")
}
for (i in c(1:2, 4:7)) {
  m0_ASM808673v1[i] <- paste(unlist(strsplit(strsplit(ASM808673v1$m_aligned_0_times,
"%")[[i]][1], ""))[10:14], collapse = "")
}
```

A continuación se preparan los datos para el *boxplot* de las lecturas no apareadas alineadas solo una vez.

```
# CheMed_v1_BIUU
m1_CheMed_v1_BIUU<- c()
for (i in c(1:12)) {
  m1_CheMed_v1_BIUU[i] <-
paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$m_aligned_exactly_1_time, "%")[[i]][1],
""))[8:12], collapse = "")
}

# ASM808673v1
m1_ASM808673v1<- c()
for (i in c(2:3, 5:6, 8, 10:12)) {
  m1_ASM808673v1[i] <- paste(unlist(strsplit(strsplit(ASM808673v1$m_aligned_exactly_1_time,
"%")[[i]][1], ""))[9:13], collapse = "")
}
for (i in c(1, 4, 7, 9)) {
  m1_ASM808673v1[i] <- paste(unlist(strsplit(strsplit(ASM808673v1$m_aligned_exactly_1_time,
"%")[[i]][1], ""))[9:12], collapse = "")
}
```

A continuación se preparan los datos para el *boxplot* de las lecturas no apareadas alineadas más de una vez.

```
# CheMed_v1_BIUU
mm1_CheMed_v1_BIUU<- c()
for (i in c(3, 7:9, 11)) {
  mm1_CheMed_v1_BIUU[i] <- paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$m_aligned_more_1_times,
"%")[[i]][1], ""))[8:12], collapse = "")
}
for (i in c(1:2, 4:6, 10)) {
  mm1_CheMed_v1_BIUU[i] <- paste(unlist(strsplit(strsplit(CheMed_v1_BIUU$m_aligned_more_1_times,
"%")[[i]][1], ""))[9:13], collapse = "")
}

# ASM808673v1
mm1_ASM808673v1<- c()
for (i in 1:12) {
  mm1_ASM808673v1[i] <- paste(unlist(strsplit(strsplit(ASM808673v1$m_aligned_more_1_times,
```

```
"%")[[i]][1], ""))[9:12], collapse = "")
}
for (i in c(1, 4, 7, 11)) {
  mm1_ASM808673v1[i] <- paste(unlist(strsplit(strsplit(ASM808673v1$m_aligned_more_1_times,
"%")[[i]][1], ""))[8:11], collapse = "")
}
```

Gráficos *Boxplot* con los porcentajes de los alineamientos tal y como aparecerán en la memoria.

```
# Boxplot de alineamientos generales
boxplot(as.numeric(unlist(strsplit(CheMed_v1_BIUU$overall_alignment_rate, "%"))),
as.numeric(unlist(strsplit(ASM808673v1$overall_alignment_rate, "%"))), names =
c("CheMed_v1_BIUU", "ASM808673v1"), col = 2:3, ylab = "% de alineamiento", xlab = "genomas",
main = "Alineamiento general",boxwex = 0.5)

# Boxplots de los alineamientos apareados
par(mfrow=c(2,2))
boxplot(as.numeric(pce1_CheMed_v1_BIUU), as.numeric(pce1_ASM808673v1), names =
c("CheMed_v1_BIUU", "ASM808673v1"), col = 2:3, ylab = "% de alineaiento", xlab = "genomas", main
= "a) Alineadas concordantemente 1 vez",boxwex = 0.5)
boxplot(as.numeric(pc0_CheMed_v1_BIUU), as.numeric(pc0_ASM808673v1), names = c("CheMed_v1_BIUU",
"ASM808673v1"), col = 2:3, ylab = "% de alineamiento", xlab = "genomas", main = "b) Alineadas
concordantemente 0 veces",boxwex = 0.5)
boxplot(as.numeric(pcm1_CheMed_v1_BIUU), as.numeric(pcm1_ASM808673v1), names =
c("CheMed_v1_BIUU", "ASM808673v1"), col = 2:3, ylab = "% de alineamiento", xlab = "genomas",
main = "c) Alineadas concordantemente >1 vez",boxwex = 0.5)
boxplot(as.numeric(pd1_CheMed_v1_BIUU), as.numeric(pd1_ASM808673v1), names = c("CheMed_v1_BIUU",
"ASM808673v1"), col = 2:3, ylab = "% de alineaiento", xlab = "genomas", main = "d)
Discordantemente",boxwex = 0.5)

# Boxplots de los alineamientos no apareados
par(mfrow=c(2,2))
boxplot(as.numeric(m0_CheMed_v1_BIUU), as.numeric(m0_ASM808673v1), names = c("CheMed_v1_BIUU",
"ASM808673v1"), col = 2:3, ylab = "% de alineamiento", xlab = "genomas", main = "a) Alineadas
cero veces",boxwex = 0.5)
boxplot(as.numeric(m1_CheMed_v1_BIUU), as.numeric(m1_ASM808673v1), names = c("CheMed_v1_BIUU",
"ASM808673v1"), col = 2:3, ylab = "% de alineamiento", xlab = "genomas", main = "b) Alineadas
solo una vez",boxwex = 0.5)
boxplot(as.numeric(mm1_CheMed_v1_BIUU), as.numeric(mm1_ASM808673v1), names = c("CheMed_v1_BIUU",
"ASM808673v1"), col = 2:3, ylab = "% de alineamiento", xlab = "genomas", main = "c) Alineadas
más de una vez",boxwex = 0.5)
```

Generación de un transcriptoma a partir de datos RNA-seq

Para ensamblar un transcriptoma a partir de los alineamientos, se utilizará **StringTie**, pero no se puede utilizar directamente el archivo en formato SAM, primero hay que transformarlo a un formato BAM ordenado respecto al genoma. Para convertir los archivos SAM en archivos BAM ordenados según su posición en el genoma de referencia, se utilizará **SamTools**.

```
mkdir resultados/bam
muestras=("SRR1634146" "SRR1634147" "SRR1634148" "SRR1634149" "SRR1634152" "SRR1634169"
"SRR1634266" "SRR1634268"
"SRR1634269" "SRR1634270" "SRR1634272" "SRR1634454")
for i in ${muestras[*]}
do
  samtools view -Su resultados/CheMed_v1_BIUU/$i\_bowtie2_very_sensitive_g2.sam | samtools sort
-o resultados/bam/$i\_bowtie2_very_sensitive_g2.sorted.bam
done
```

Una vez que se disponga del formato adecuado, se está en disposición de utilizar **StringTie** en cada una de las muestras. Utilizando el argumento -A se obtiene un archivo de texto por muestra con la información de cada transcrito ensamblado incluyendo su abundancia en distintas unidades.

```
mkdir resultados/Stringtie
muestras=("SRR1634146" "SRR1634147" "SRR1634148" "SRR1634149" "SRR1634152" "SRR1634169"
"SRR1634266" "SRR1634268"
"SRR1634269" "SRR1634270" "SRR1634272" "SRR1634454")
for i in ${muestras[*]}
do
  herramientas/stringtie/./stringtie resultados/bam/$i\_bowtie2_very_sensitive_g2.sorted.bam -A
resultados/Stringtie/$i\_abundance_very_sensitive_g2.txt --rf -o
resultados/Stringtie/$i\_bowtie2_very_sensitive_g2.gff -l $i
done
```

Se creará una `lista` que contenga doce `dataframes`, uno por cada informe de los transcritos creados a partir del ensamblaje de cada una de las muestras.

```
gff<-list()
for (i in 1:12) {
  gff[[i]] <- read.delim(paste("resultados/",
targets$Código[i],"_abundance_very_sensitive.txt",sep = "") ,
                         header=TRUE, comment.char="#", as.is = T)
}
names(gff)<-targets$Código
```

El número de transcritos en cada una de las muestras es el siguiente.

```
transcritos<-c()
for (i in 1:12) {
  transcritos[i]<- nrow(gff[[i]])
}
data.frame( targets$Código, transcritos)
```

Se calculará la longitud de cada transcrito de cada muestra y se añadirá una columna (`longitud`) a cada `dataframe` con dicha información.

```
for (i in 1:12) {
  for (j in 1:nrow(gff[[i]])) {
    gff[[i]]$longitud<- gff[[i]][[6]]-gff[[i]][[5]]+1
  }
}
```

Se representarán las longitudes de los transcritos en cada estado (`Active1`, `Active2`, `Torpor`) para cada individuo (`Tanager`, `Quetzal`, `Osprey`, `Towhee`) con un *Density plot*.

```
longitud <- c(gff[[1]]$longitud, gff[[2]]$longitud, gff[[3]]$longitud, gff[[4]]$longitud,
              gff[[5]]$longitud, gff[[6]]$longitud, gff[[7]]$longitud, gff[[8]]$longitud,
              gff[[9]]$longitud, gff[[10]]$longitud, gff[[11]]$longitud, gff[[12]]$longitud)
summary(longitud)

nombres<-c(rep("Tanager",nrow(gff[[1]])), rep("Quetzal", nrow(gff[[2]])), rep("Osprey",
nrow(gff[[3]])),
           rep("Towhee", nrow(gff[[4]])), rep("Osprey", nrow(gff[[5]])), rep("Towhee",
nrow(gff[[6]])),
           rep("Tanager", nrow(gff[[7]])), rep("Towhee", nrow(gff[[8]])), rep("Quetzal",
nrow(gff[[9]])),
           rep("Tanager", nrow(gff[[10]])), rep("Osprey", nrow(gff[[11]])), rep("Quetzal",
nrow(gff[[12]])))

fecha<- c(rep("octubre", nrow(gff[[1]])), rep("octubre", nrow(gff[[2]])),
               rep("octubre", nrow(gff[[3]])), rep("octubre", nrow(gff[[4]])),
               rep("diciembre", nrow(gff[[5]])), rep("diciembre", nrow(gff[[6]])),
               rep("diciembre", nrow(gff[[7]])), rep("febrero", nrow(gff[[8]])), rep("febrero",
nrow(gff[[9]])),
               rep("febrero", nrow(gff[[10]])), rep("febrero", nrow(gff[[11]])),
rep("diciembre", nrow(gff[[12]])))

muetras_longitud <- data.frame(longitud, nombres, fecha)

ggplot(muetras_longitud, aes(x = log2(longitud), colour = nombres))  +
geom_density(size = 0.1) + facet_wrap(~ fecha) +
theme(legend.position = "top") + xlab("log2(longitud)(pb)")+ xlim(c(7,15))
```

Veamos el resumen de los datos estadísticos de las longitudes de los transcritos ensamblados en cada muestra.

```
resumen_longitud <- c()
for (i in 1:12) {
  resumen_longitud <- rbind(resumen_longitud, unname(summary(gff[[i]]$longitud)))
}

resumen_longitud <- data.frame(resumen_longitud)
names(resumen_longitud) <- names(summary(gff[[1]]$longitud))
rownames(resumen_longitud) <- names(gff)
resumen_longitud
```

Se representarán las coberturas de los transcritos en cada estado (`Active1`, `Active2`, `Torpor`) para cada individuo (`Tanager`, `Quetzal`, `Osprey`, `Towhee`) con un *Density plot*.

```
Coverage<-c(gff[[1]]$Coverage, gff[[2]]$Coverage, gff[[3]]$Coverage, gff[[4]]$Coverage,
            gff[[5]]$Coverage, gff[[6]]$Coverage, gff[[7]]$Coverage, gff[[8]]$Coverage,
```

```
            gff[[9]]$Coverage, gff[[10]]$Coverage, gff[[11]]$Coverage, gff[[12]]$Coverage)
summary(Coverage)
muetras_Coverage<- data.frame(Coverage, nombres, fecha)

ggplot(muetras_Coverage, aes(x = log2(Coverage), colour = nombres))  +
geom_density(size = 0.1) + facet_wrap(~ fecha) +
theme(legend.position = "top") + xlab("log2(Coverage)") + xlim(c(0,3))
```

Veamos el resumen de los datos estadísticos de la cobertura en los transcritos ensamblados para cada muestra.

```
resumen_Coverage <- c()
for (i in 1:12) {
  resumen_Coverage <- rbind(resumen_Coverage, unname(summary(gff[[i]]$Coverage)))
}

resumen_Coverage <- data.frame(resumen_Coverage)
names(resumen_Coverage) <- names(summary(gff[[1]]$Coverage))
rownames(resumen_Coverage) <- names(gff)
resumen_Coverage
```

Se representarán los FPKM en cada estado (`Active1`, `Active2`, `Torpor`) para cada individuo (`Tanager`, `Quetzal`, `Osprey`, `Towhee`) con un *Density plot*.

```
FPKM<-c(gff[[1]]$FPKM, gff[[2]]$FPKM, gff[[3]]$FPKM, gff[[4]]$FPKM, gff[[5]]$FPKM,
        gff[[6]]$FPKM, gff[[7]]$FPKM, gff[[8]]$FPKM, gff[[9]]$FPKM,
        gff[[10]]$FPKM, gff[[11]]$FPKM, gff[[12]]$FPKM)

muetras_FPKM<- data.frame(FPKM,nombres,fecha)
summary(FPKM)
ggplot(muetras_FPKM, aes(x = log2(FPKM), colour = nombres))  +
geom_density(size = 0.1) + facet_wrap(~ fecha) +
theme(legend.position = "top") + xlab("log2(FPKM)") + xlim(c(0,2.5))
```

Veamos el resumen de los datos estadísticos de los FPKM en los transcritos ensamblados para cada muestra.

```
resumen_FPKM <- c()
for (i in 1:12) {
  resumen_FPKM <- rbind(resumen_FPKM, unname(summary(gff[[i]]$FPKM)))
}

resumen_FPKM <- data.frame(resumen_FPKM)
names(resumen_FPKM) <- names(summary(gff[[1]]$FPKM))
rownames(resumen_FPKM) <- names(gff)
resumen_FPKM
```

Se representarán los TPM en cada estado (`Active1`, `Active2`, `Torpor`) para cada individuo (`Tanager`, `Quetzal`, `Osprey`, `Towhee`) con un *Density plot*.

```
TPM<-c(gff[[1]]$TPM, gff[[2]]$TPM, gff[[3]]$TPM, gff[[4]]$TPM, gff[[5]]$TPM, gff[[6]]$TPM,
gff[[7]]$TPM,
       gff[[8]]$TPM, gff[[9]]$TPM, gff[[10]]$TPM, gff[[11]]$TPM, gff[[12]]$TPM)
muetras_TPM<- data.frame(TPM,nombres,fecha)
summary(TPM)
ggplot(muetras_TPM, aes(x = log(TPM), colour = nombres))  +
geom_density(size = 0.1) + facet_wrap(~ fecha) +
theme(legend.position = "top") + xlab("log2(TPM)") + xlim(c(0.5,3))
```

Veamos el resumen de los datos estadísticos de los TPM para cada muestra.

```
resumen_TPM <- c()
for (i in 1:12) {
  resumen_TPM <- rbind(resumen_TPM, unname(summary(gff[[i]]$TPM)))
}

resumen_TPM <- data.frame(resumen_TPM)
names(resumen_TPM) <- names(summary(gff[[1]]$TPM))
rownames(resumen_TPM) <- names(gff)
resumen_TPM
```

Los transcritos de cada una de las muestras se pueden unir de manera no redundante con el argumento *merge* de **StringTie**, pero antes se debe crear un archivo de texto (`assembly_GFF_list.txt`) que contenga el nombre de todos los archivos que fusionaremos.

```
cd resultados/Stringtie
ls SRR*_bowtie2_very_sensitive.gff > assembly_GFF_list_very_sensitive_g2.txt
```

Se fusionarán todos los transcriptomas en un solo archivo con todos los transcritos.

```
cd resultados/Stringtie
../../herramientas/stringtie/./stringtie --merge -l transcritoma -o merged_very_sensitive_g2.gff
assembly_GFF_list_very_sensitive_g2.txt
```

Se creará un `dataframe` con los datos obtenidos de la unión de todos los transcritos de cada muestra. A este `dataframe` se le añadirá la columna longitud, como se hizo anteriormente, y se creará un *Density plot* con las longitudes de todos los transcritos.

```
merged <- read.delim("resultados/merged_very_sensitive_g2.gff", header=FALSE,
                     comment.char="#", stringsAsFactors=FALSE)
colnames(merged)<-c("seqnames", "StringTie", "trans_exon", "start", "end", "", "strand", "", "")
for (j in 1:nrow(merged)) {
    merged$longitud<- merged[[5]]-merged[[4]]+1
}
nrow(merged)
summary(merged$longitud)
ggplot(merged, aes(x = log2(longitud)))  +
geom_density( size = 0.8)  +
xlab("log(longitud)(pb)")  +
xlim(c(7.3,16))
```

Comparación de los transcritos con anotaciones en humanos

Se convertirá el ensamblaje fusionado de todas las muestras en un archivo FASTA con las secuencias de los transcritos potenciales.

```
herramientas/bedtools2/bin/bedtools getfasta -fi
datos/GCA_004024725.1_CheMed_v1_BIUU_genomic.fna -bed
resultados/Stringtie/merged_very_sensitive_g2.gff -fo resultados/transcriptoma_g2.2.fa
```

Se buscarán los transcritos que tienen similitud con las proteínas caracterizadas en humanos.

```
herramientas/blastx -query resultados/transcriptoma_g2.2.fa -db
datos/GCF_000001405.39_GRCh38.p13_protein.faa -out resultados/blastx_very_sensitive_g2.A.outfmt6
-evalue 1e-4 -num_threads 2 -max_target_seqs 10 -outfmt 6
```

Se generará un archivo con la información anterior más el porcentaje de la proteína humana que suponen los transcritos.

```
herramientas/trinityrnaseq-v2.8.6/util/analyze_blastPlus_topHit_coverage.pl
resultados/blastx_very_sensitive_g2.A.outfmt6 resultados/transcriptoma_g2.2.fa
datos/GCF_000001405.39_GRCh38.p13_protein.faa
```

Se seleccionarán únicamente los transcritos que supongan un 70% o más de la proteína con la que tienen similitud.

```
nrow(merged)/2
hitlength <- read.delim("resultados/blastx_very_sensitive_g2.A.outfmt6.w_pct_hit_length",
                        header=T, stringsAsFactors=FALSE)
nrow(hitlength)
length(unique(hitlength$sseqid)) # transcritos con proteina homologa sin contar las redundacias
trans_70R <-hitlength[hitlength$pct_hit_len_aligned >=70,]
length(unique(trans_70R$sseqid))  # transcritos con 70% o más de la longitud de la proteina
homologa sin contar los redundantes
trans_70 <-hitlength$X.qseqid[hitlength$pct_hit_len_aligned >=70]
length(trans_70)
```

Se creará un objeto readDNAStringSet que contenga las secuencias del transcriptoma ensamblado con **Stringtie** y se comprobará su número.

```
transc <- readDNAStringSet("resultados/transcriptoma_g2.2.fa")
transcI <- transc[seq(1, length(transc), 2)]
length(transcI)
head(transcI)
```

Se creará un archivo FASTA con las secuencias de los transcritos que tengan un 70% o más de la longitud de la proteína con la que tienen homología.

```
writeXStringSet(DNAStringSet(transcI)[trans_70], 'resultados/transcriptoma_70.fa')
```

Cuantificación de lecturas

El pseudoalineamiento que realiza **Kallisto** requiere del procesamiento del archivo que contiene el transcriptoma para crear un índice.

```
herramientas/kallisto_linux-v0.46.0/kallisto/kallisto index -i resultados/transcriptoma_70.idx
resultados/transcriptoma_70.fa
```

Se correrá **Kallisto** para ver la abundancia de cada transcrito en cada muestra.

```
mkdir resultados/kallisto resultados/kallisto_datos
muestras=("SRR1634146" "SRR1634147" "SRR1634148" "SRR1634149" "SRR1634152" "SRR1634169"
"SRR1634266" "SRR1634268"
"SRR1634269" "SRR1634270" "SRR1634272" "SRR1634454")
for i in ${muestras[*]}
do
  herramientas/kallisto_linux-v0.46.0/kallisto/kallisto quant -t 2 -i
resultados/transcriptoma_70.idx -o resultados/kallisto/$i -b 100
resultados/Trimmomatic/$i\_f_p.fastq resultados/Trimmomatic/$i\_r_p.fastq 2>
resultados/kallisto_datos/$i\kallisto.log
done
```

Identificación de genes diferencialmente expresados

Se especificará dónde se encuentran los resultados obtenidos por **Kallisto**.

```
sample_id <- dir(file.path("resultados", "kallisto"))
```

Veamos una lista de rutas a los resultados de **Kallisto** ordenados por el nombre de las muestras.

```
kal_dirs <- file.path("resultados", "kallisto", sample_id)
kal_dirs
```

La matriz de diseño solo contendrá tres individuos, puesto que Towhee perdió mucho peso hecho que podría alterar los resultados.

```
# Matriz de diseño con todos los individuos
targets$kal_dirs<-kal_dirs
metadatos_total <- data.frame(sample = c("Ta1", "Qu1", "Os1", "Os2", "Ta2", "QuT", "TaT", "OsT",
"Qu2"), estado = targets_TN$Estado, nombre =targets_TN$Animal,
                        path = targets_TN$kal_dirs, nom_esta = paste( targets_TN$Animal,
targets_TN$Estado, sep = "_"), stringsAsFactors = FALSE)
metadatos_total

# Matriz de diseño sin Towhee
targets_TN <- targets[targets$Animal!="Towhee",]
metadatos <- data.frame(sample = targets_TN$Código, estado = targets_TN$Estado, nombre
=targets_TN$Animal,
                        path = targets_TN$kal_dirs, stringsAsFactors = FALSE)
metadatos
```

Se compararán los estados fisiológicos dos a dos, así que se creará una tabla de `metadatos` que contenga los estados de dos en dos.

Comparación `Active1` con `Active2`:

```
metadatos_A1_A2 <- metadatos[metadatos$estado==c("Active1") | metadatos$estado==c("Active2"),]
```

Comparación `Active1` con `Torpor`:

```
metadatos_A1_Tp <- metadatos[metadatos$estado==c("Active1") | metadatos$estado==c("Torpor"),]
```

Comparación `Active2` con `Torpor`:

```
metadatos_A2_Tp <- metadatos[metadatos$estado==c("Active2") | metadatos$estado==c("Torpor"),]
```

Comparación `Active1` y `Active2` con `Torpor`:

```
metadatos_AA_Tp <- metadatos
metadatos_AA_Tp$estado <- c(rep("Active",5), rep("Torpor",3), "Active")
```

Los objetos Sleuth `so_A1_A2`, `so_A1_Tp` y `so_A2_Tp` deberán ser creados por la función `sleuth_prep`.

```
# num_cores <- 2
so_A1_A2 <- sleuth_prep(metadatos_A1_A2, extra_bootstrap_summary = TRUE)
so_A1_Tp <- sleuth_prep(metadatos_A1_Tp, extra_bootstrap_summary = TRUE)
so_A2_Tp <- sleuth_prep(metadatos_A2_Tp, extra_bootstrap_summary = TRUE)
so_AA_Tp <- sleuth_prep(metadatos_AA_Tp, extra_bootstrap_summary = TRUE)
so <- sleuth_prep(metadatos_total, extra_bootstrap_summary = TRUE)
```

Se pretende examinar qué genes están diferencialmente expresados entre los diferentes estados dos a dos teniendo en cuenta las diferencias entre los individuos. Para ello se ajustarán dos modelos en cada pareja de estados. El

primero se conoce como “modelo reducido” e incluirá el parámetro que se pretende controlar, que corresponderá a la variación entre individuos (`nombre`). El modelo reducido se ajustará con `sleuth_fit`.

```
so1_A1_A2 <- sleuth_fit(so_A1_A2, ~nombre, 'reduced')
so1_A1_Tp <- sleuth_fit(so_A1_Tp, ~nombre, 'reduced')
so1_A2_Tp <- sleuth_fit(so_A2_Tp, ~nombre, 'reduced')
so1_AA_Tp <- sleuth_fit(so_AA_Tp, ~nombre, 'reduced')
so1 <- sleuth_fit(so, ~nombre, 'reduced')
```

El segundo modelo se conoce como “modelo completo” e incluirá todos los parámetros (`nombre` y `estado`). El modelo completo se ajustará con `sleuth_fit`.

```
so1_A1_A2 <- sleuth_fit(so1_A1_A2, ~estado + nombre, 'full')
so1_A1_Tp <- sleuth_fit(so1_A1_Tp, ~estado + nombre, 'full')
so1_A2_Tp <- sleuth_fit(so1_A2_Tp, ~estado + nombre, 'full')
so1_AA_Tp <- sleuth_fit(so1_AA_Tp, ~estado + nombre, 'full')
so1 <- sleuth_fit(so1, ~estado + nombre, 'full')
```

Se compararán los dos modelos con la prueba de razón de verosimilitud y con el test de Wald.

```
# LRT
so1_A1_A2 <- sleuth_lrt(so1_A1_A2, 'reduced', 'full')
so1_A1_Tp <- sleuth_lrt(so1_A1_Tp, 'reduced', 'full')
so1_A2_Tp <- sleuth_lrt(so1_A2_Tp, 'reduced', 'full')
so1_AA_Tp <- sleuth_lrt(so1_AA_Tp, 'reduced', 'full')

# Wald Test
so1_A1_A2_wt <- sleuth_wt(so1_A1_A2, which_beta =  "estadoActive2")
so1_A1_Tp_wt <- sleuth_wt(so1_A1_Tp, which_beta =  "estadoTorpor")
so1_A2_Tp_wt <- sleuth_wt(so1_A2_Tp, which_beta =  "estadoTorpor")
so1_AA_Tp_wt <- sleuth_wt(so1_AA_Tp, which_beta =  "estadoTorpor")
```

La función `sleuth_results` usará los p-valores de la comparación de los transcritos para determinar los niveles génicos y la expresión génica diferencial. Se creará un `dataframe` por cada par de estados con los datos estadísticos para cada transcrito que tenga una expresión génica diferencialmente significativa. Se considerará que un transcrito tiene una expresión génica diferencialmente significativa si `qval` es mayor o igual a 0.05.

Comparación `Active1` con `Active2`:

```
# LRT
sleuth_table_A1_A2 <- sleuth_results(so1_A1_A2, 'reduced:full', 'lrt', show_all = FALSE)
sleuth_significant_A1_A2 <- dplyr::filter(sleuth_table_A1_A2, qval <= 0.05)
head(sleuth_significant_A1_A2, 20)[1:6]

# Wald Test
sleuth_table_A1_A2_wt <- sleuth_results(so1_A1_A2_wt, 'estadoActive2', 'wt', show_all = FALSE)
sleuth_significant_A1_A2_wt <- dplyr::filter(sleuth_table_A1_A2_wt, qval <= 0.05)
```

Comparación `Active1` con `Torpor`:

```
# LRT
sleuth_table_A1_Tp <- sleuth_results(so1_A1_Tp, 'reduced:full', 'lrt', show_all = F)
sleuth_significant_A1_Tp <- dplyr::filter(sleuth_table_A1_Tp, qval <= 0.05)
head(sleuth_significant_A1_Tp, 20)[1:6]
# Wald Test
sleuth_table_A1_Tp_wt <- sleuth_results(so1_A1_Tp_wt, 'estadoTorpor', 'wt', show_all = FALSE)
sleuth_significant_A1_Tp_wt <- dplyr::filter(sleuth_table_A1_Tp_wt, qval <= 0.05)
```

Comparación `Active2` con `Torpor`:

```
#LRT
sleuth_table_A2_Tp <- sleuth_results(so1_A2_Tp, 'reduced:full', 'lrt', show_all = FALSE)
sleuth_significant_A2_Tp <- dplyr::filter(sleuth_table_A2_Tp, qval <= 0.05)
head(sleuth_significant_A2_Tp, 20)[1:6]

# Wald Test
sleuth_table_A2_Tp_wt <- sleuth_results(so1_A2_Tp_wt, 'estadoTorpor', 'wt', show_all = FALSE)
sleuth_significant_A2_Tp_wt <- dplyr::filter(sleuth_table_A2_Tp_wt, qval <= 0.05)
```

Comparación `Active1` y `Active2` con `Torpor`:

```
#LRT
sleuth_table_AA_Tp <- sleuth_results(so1_AA_Tp, 'reduced:full', 'lrt', show_all = FALSE)
sleuth_significant_AA_Tp <- dplyr::filter(sleuth_table_AA_Tp, qval <= 0.05)
head(sleuth_significant_A2_Tp, 20)[1:6]

# Wald Test
```

```
sleuth_table_AA_Tp_wt <- sleuth_results(so1_AA_Tp_wt, 'estadoTorpor', 'wt', show_all = FALSE)
sleuth_significant_AA_Tp_wt <- dplyr::filter(sleuth_table_AA_Tp_wt, qval <= 0.05)
```

Se introducirá en los `dataframe` anteriores una columna con el nombre de la proteína correspondiente a cada transcrito. Se obtendrá, también el número de proteínas obtenidas.

Comparación `Active1` con `Active2`:

```
# LRT
proteinas_A1_A2 <- c()
ifelse(nrow(sleuth_significant_A1_A2) == 0, proteinas_A1_A2 <- NA, for (i in
1:nrow(sleuth_significant_A1_A2)) {
  proteinas_A1_A2 <- c(proteinas_A1_A2,

hitlength$hit_descr[hitlength$X.qseqid==sleuth_significant_A1_A2$target_id[i]])
})
length(unique(proteinas_A1_A2))
ifelse(nrow(sleuth_significant_A1_A2) == 0, proteinas_A1_A2 <- NA,
sleuth_significant_A1_A2$proteina <- proteinas_A1_A2)

# Wald Test
proteinas_A1_A2_wt <- c()
for (i in 1:nrow(sleuth_significant_A1_A2_wt)) {
  proteinas_A1_A2_wt <- c(proteinas_A1_A2_wt,

hitlength$hit_descr[hitlength$X.qseqid==sleuth_significant_A1_A2_wt$target_id[i]])
}
length(unique(proteinas_A1_A2_wt))
sleuth_significant_A1_A2_wt$proteina <- proteinas_A1_A2_wt
```

Comparación `Active1` con `Torpor`:

```
# LRT
proteinas_A1_Tp <- c()
for (i in 1:nrow(sleuth_significant_A1_Tp)) {
  proteinas_A1_Tp <- c(proteinas_A1_Tp,

hitlength$hit_descr[hitlength$X.qseqid==sleuth_significant_A1_Tp$target_id[i]])
}
length(unique(proteinas_A1_Tp))
sleuth_significant_A1_Tp$proteina <- proteinas_A1_Tp

# Wald Test
proteinas_A1_Tp_wt <- c()
for (i in 1:nrow(sleuth_significant_A1_Tp_wt)) {
  proteinas_A1_Tp_wt <- c(proteinas_A1_Tp_wt,

hitlength$hit_descr[hitlength$X.qseqid==sleuth_significant_A1_Tp_wt$target_id[i]])
}
length(unique(proteinas_A1_Tp_wt))
sleuth_significant_A1_Tp_wt$proteina <- proteinas_A1_Tp_wt
```

Comparación `Active2` con `Torpor`:

```
# LRT
proteinas_A2_Tp <- c()
for (i in 1:nrow(sleuth_significant_A2_Tp)) {
  proteinas_A2_Tp <- c(proteinas_A2_Tp,

hitlength$hit_descr[hitlength$X.qseqid==sleuth_significant_A2_Tp$target_id[i]])
}
length(unique(proteinas_A2_Tp))
sleuth_significant_A2_Tp$proteina <- proteinas_A2_Tp

# Wald Test
proteinas_A2_Tp_wt <- c()
for (i in 1:nrow(sleuth_significant_A2_Tp_wt)) {
  proteinas_A2_Tp_wt <- c(proteinas_A2_Tp_wt,

hitlength$hit_descr[hitlength$X.qseqid==sleuth_significant_A2_Tp_wt$target_id[i]])
}
length(unique(proteinas_A2_Tp_wt))
sleuth_significant_A2_Tp_wt$proteina <- proteinas_A2_Tp_wt
```

Comparación `Active1` y `Active2` con `Torpor`:

```
# LRT
proteinas_AA_Tp <- c()
for (i in 1:nrow(sleuth_significant_AA_Tp)) {
  proteinas_AA_Tp <- c(proteinas_AA_Tp,

hitlength$hit_descr[hitlength$X.qseqid==sleuth_significant_AA_Tp$target_id[i]])
}
length(unique(proteinas_AA_Tp))
sleuth_significant_AA_Tp$proteina <- proteinas_AA_Tp

# Wald Test
proteinas_AA_Tp_wt <- c()
for (i in 1:nrow(sleuth_significant_AA_Tp_wt)) {
  proteinas_AA_Tp_wt <- c(proteinas_AA_Tp_wt,

hitlength$hit_descr[hitlength$X.qseqid==sleuth_significant_AA_Tp_wt$target_id[i]])
}
length(unique(proteinas_AA_Tp_wt))
sleuth_significant_AA_Tp_wt$proteina <- proteinas_AA_Tp_wt
```

La identificación de nuestros transcritos, en algunos casos, es la utilizada por *Reference Sequence* para peptidos y, en otros casos, es la utilizada por *Reference Sequence* para peptidos predichos. Con la intención de homogeneizar el modo de identificar los transcritos se extraerán con BioMart los identificadores utilizados por *Ensembl Genomes* para genes y peptidos, además del símbolo del *HUGO Gene Nomenclature Committee* y una descripción.

```
mart <- useMart(biomart = "ensembl", dataset = "hsapiens_gene_ensembl")
t2g <- biomaRt::getBM(attributes = c( "hgnc_symbol", "ensembl_gene_id", "ensembl_peptide_id",
"refseq_peptide", "refseq_peptide_predicted", "description"), mart = mart)
```

Se obtendrá un `dataframe` con las equivalencias entre el locus y el nombre de la proteína.

Se introducirán en los resultados obtenidos para cada comparación dos columnas una con el identificador *Reference Sequence* con el número de versión denominada `locus` y otra sin él, denominada `locus_nv`. También se introducirá una columna para el símbolo del *HUGO Gene Nomenclature Committee*, otra para la identificación del gen utilizada por *Ensembl Genomes*, otra para la identificación del peptido utilizada por *Ensembl Genomes* y otra para la descripción.

Comparación `Active1` con `Active2`:

```
# Se introduce en sleuth_significant_A1_A2 el nombre del locus

## LRT
# locus_A1_A2 <- c()
# for (i in 1:nrow(sleuth_significant_A1_A2)) {
#   locus_A1_A2 <- c(locus_A1_A2,
#
hitlength$sseqid[hitlength$X.qseqid==sleuth_significant_A1_A2$target_id[i]])
# }
# sleuth_significant_A1_A2$locus <- locus_A1_A2

## Wald Test
locus_A1_A2_wt <- c()
for (i in 1:nrow(sleuth_significant_A1_A2_wt)) {
  locus_A1_A2_wt <- c(locus_A1_A2_wt,

hitlength$sseqid[hitlength$X.qseqid==sleuth_significant_A1_A2_wt$target_id[i]])
}
sleuth_significant_A1_A2_wt$locus <- locus_A1_A2_wt

#Locus sin version

## LRT
# locus_A1_A2_c <- c()
# for (i in 1:length(locus_A1_A2)) {
#   locus_A1_A2_c[i] <- paste(strsplit(locus_A1_A2[i],
"")[[1]][1:(length(strsplit(locus_A1_A2[i], "")[[1]])-2)], collapse = "")
# }
# sleuth_significant_A1_A2$locus_nv <- locus_A1_A2_c

## Wald Test
locus_A1_A2_c_wt <- c()
for (i in 1:length(locus_A1_A2_wt)) {
  locus_A1_A2_c_wt[i] <- paste(strsplit(locus_A1_A2_wt[i],
"")[[1]][1:(length(strsplit(locus_A1_A2_wt[i], "")[[1]])-2)], collapse = "")
}
sleuth_significant_A1_A2_wt$locus_nv <- locus_A1_A2_c_wt
```

```r
# hgnc, ensembl_gen_id, ensembl_peptide_id, description

## LRT
# hgnc_A1_A2 <- c()
# ensembl_gen_id_A1_A2 <- list()
# ensembl_peptide_id_A1_A2 <- list()
# description_A1_A2 <- c()
# for (i in 1:length(locus_A1_A2_c)) {
#   if(length(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A1_A2_c[i]]) == 0) {
#     hgnc_A1_A2[i] <- NA
#     ensembl_gen_id_A1_A2[[i]] <- NA
#     ensembl_peptide_id_A1_A2[[i]] <- NA
#     description_A1_A2[i] <- NA
#   }
#   else {
#     hgnc_A1_A2[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A1_A2_c[i]])
#     ensembl_gen_id_A1_A2[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide ==
locus_A1_A2_c[i]])
#     ensembl_peptide_id_A1_A2[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
locus_A1_A2_c[i]])
#     description_A1_A2[i] <- unique(t2g$description[t2g$refseq_peptide == locus_A1_A2_c[i]])
#     }
# }
# for (i in 1:length(locus_A1_A2_c)) {
#   if(length(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A1_A2_c[i]]) == 0) {
#     hgnc_A1_A2[i] <- hgnc_A1_A2[i]
#     ensembl_gen_id_A1_A2[[i]] <- ensembl_gen_id_A1_A2[[i]]
#     ensembl_peptide_id_A1_A2[[i]] <- ensembl_peptide_id_A1_A2[[i]]
#     description_A1_A2[i] <- description_A1_A2[i]
#   }
#   else {
#     hgnc_A1_A2[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A1_A2_c[i]])
#     ensembl_gen_id_A1_A2[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide_predicted ==
locus_A1_A2_c[i]])
#     ensembl_peptide_id_A1_A2[[i]] <-
unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted == locus_A1_A2_c[i]])
#     description_A1_A2[i] <- unique(t2g$description[t2g$refseq_peptide_predicted ==
locus_A1_A2_c[i]])
#     }
# }
# sleuth_significant_A1_A2$hgnc <- hgnc_A1_A2
# for (i in 1:length(ensembl_gen_id_A1_A2)) {
# sleuth_significant_A1_A2$ensembl_gen_id[i] <- paste(ensembl_gen_id_A1_A2[[i]], collapse =  "
")
# }
# for (i in 1:length(ensembl_peptide_id_A1_A2)) {
# sleuth_significant_A1_A2$ensembl_peptide_id[i] <- paste(ensembl_peptide_id_A1_A2[[i]],
collapse =  " ")
# }
#
# sleuth_significant_A1_A2$description <- description_A1_A2
#
# buscar_A1_A2 <- sleuth_significant_A1_A2[is.na(hgnc_A1_A2),][13:19]   # Nombre de proteinas
que no estan en hgnc
# write.csv(buscar_A1_A2, "resultados/buscar_A1_A2")

## Wald Test
hgnc_A1_A2_wt <- c()
ensembl_gen_id_A1_A2_wt <- list()
ensembl_peptide_id_A1_A2_wt <- list()
description_A1_A2_wt <- c()
for (i in 1:length(locus_A1_A2_c_wt)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A1_A2_c_wt[i]]) == 0) {
    hgnc_A1_A2_wt[i] <- NA
    ensembl_gen_id_A1_A2_wt[[i]] <- NA
    ensembl_peptide_id_A1_A2_wt[[i]] <- NA
    description_A1_A2_wt[i] <- NA
  }
  else {
    hgnc_A1_A2_wt[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A1_A2_c_wt[i]])
    ensembl_gen_id_A1_A2_wt[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide ==
locus_A1_A2_c_wt[i]])
    ensembl_peptide_id_A1_A2_wt[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
locus_A1_A2_c_wt[i]])
```

```
    description_A1_A2_wt[i] <- unique(t2g$description[t2g$refseq_peptide ==
locus_A1_A2_c_wt[i]])
    }
}
for (i in 1:length(locus_A1_A2_c_wt)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A1_A2_c_wt[i]]) == 0) {
    hgnc_A1_A2_wt[[i]] <- hgnc_A1_A2_wt[[i]]
    ensembl_gen_id_A1_A2_wt[i] <- ensembl_gen_id_A1_A2_wt[i]
    ensembl_peptide_id_A1_A2_wt[[i]] <- ensembl_peptide_id_A1_A2_wt[[i]]
    description_A1_A2_wt[i] <- description_A1_A2_wt[i]
  }
  else {
    hgnc_A1_A2_wt[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide_predicted ==
locus_A1_A2_c_wt[i]])
    ensembl_gen_id_A1_A2_wt[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide_predicted ==
locus_A1_A2_c_wt[i]])
    ensembl_peptide_id_A1_A2_wt[[i]] <-
unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted == locus_A1_A2_c_wt[i]])
    description_A1_A2_wt[i] <- unique(t2g$description[t2g$refseq_peptide_predicted ==
locus_A1_A2_c_wt[i]])
    }
}

sleuth_significant_A1_A2_wt$hgnc <- hgnc_A1_A2_wt
for (i in 1:length(ensembl_gen_id_A1_A2_wt)) {
sleuth_significant_A1_A2_wt$ensembl_gen_id[i] <- paste(ensembl_gen_id_A1_A2_wt[[i]], collapse =
" ")
}
for (i in 1:length(ensembl_peptide_id_A1_A2_wt)) {
sleuth_significant_A1_A2_wt$ensembl_peptide_id[i] <- paste(ensembl_peptide_id_A1_A2_wt[[i]],
collapse =  " ")
}

sleuth_significant_A1_A2_wt$description <- description_A1_A2_wt

buscar_A1_A2_wt <- sleuth_significant_A1_A2_wt[is.na(hgnc_A1_A2_wt),][12:18]   # Nombre de
proteinas que no estan en hgnc
write.csv(buscar_A1_A2_wt, "resultados/buscar_A1_A2_wt")

write.csv(sleuth_significant_A1_A2_wt, "resultados/sleuth_significant_A1_A2_wt.txt")
```

Comparación `Active1` con `Torpor`:

```
# Se introduce en sleuth_significant_A1_Tp el nombre del locus

## LRT
locus_A1_Tp <- c()
for (i in 1:nrow(sleuth_significant_A1_Tp)) {
  locus_A1_Tp <- c(locus_A1_Tp,

hitlength$sseqid[hitlength$X.qseqid==sleuth_significant_A1_Tp$target_id[i]])
}
sleuth_significant_A1_Tp$locus <- locus_A1_Tp

## Wald Test
locus_A1_Tp_wt <- c()
for (i in 1:nrow(sleuth_significant_A1_Tp_wt)) {
  locus_A1_Tp_wt <- c(locus_A1_Tp_wt,

hitlength$sseqid[hitlength$X.qseqid==sleuth_significant_A1_Tp_wt$target_id[i]])
}
sleuth_significant_A1_Tp_wt$locus <- locus_A1_Tp_wt


#Locus sin versión

## LRT
locus_A1_Tp_c <- c()
for (i in 1:length(locus_A1_Tp)) {
  locus_A1_Tp_c[i] <- paste(strsplit(locus_A1_Tp[i], "")[[1]][1:(length(strsplit(locus_A1_Tp[i],
"")[[1]])-2)], collapse = "")
}
sleuth_significant_A1_Tp$locus_nv <- locus_A1_Tp_c

## Wald Test
```

```
locus_A1_Tp_c_wt <- c()
for (i in 1:length(locus_A1_Tp_wt)) {
  locus_A1_Tp_c_wt[i] <- paste(strsplit(locus_A1_Tp_wt[i],
"")[[1]][1:(length(strsplit(locus_A1_Tp_wt[i], "")[[1]])-2)], collapse = "")
}
sleuth_significant_A1_Tp_wt$locus_nv <- locus_A1_Tp_c_wt

# hgnc, ensembl_gen_id, ensembl_peptide_id, description

## LRT
hgnc_A1_Tp <- c()
ensembl_gen_id_A1_Tp <- list()
ensembl_peptide_id_A1_Tp <- list()
description_A1_Tp <- c()
for (i in 1:length(locus_A1_Tp_c)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A1_Tp_c[i]]) == 0) {
    hgnc_A1_Tp[i] <- NA
    ensembl_gen_id_A1_Tp[[i]] <- NA
    ensembl_peptide_id_A1_Tp[[i]] <- NA
    description_A1_Tp[i] <- NA
  }
  else {
    hgnc_A1_Tp[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A1_Tp_c[i]])
    ensembl_gen_id_A1_Tp[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide ==
locus_A1_Tp_c[i]])
    ensembl_peptide_id_A1_Tp[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
locus_A1_Tp_c[i]])
    description_A1_Tp[i] <- unique(t2g$description[t2g$refseq_peptide == locus_A1_Tp_c[i]])
    }
}
for (i in 1:length(locus_A1_Tp_c)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A1_Tp_c[i]]) == 0) {
    hgnc_A1_Tp[i] <- hgnc_A1_Tp[i]
    ensembl_gen_id_A1_Tp[[i]] <- ensembl_gen_id_A1_Tp[[i]]
    ensembl_peptide_id_A1_Tp[[i]] <- ensembl_peptide_id_A1_Tp[[i]]
    description_A1_Tp[i] <- description_A1_Tp[i]
  }
  else {
    hgnc_A1_Tp[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A1_Tp_c[i]])
    ensembl_gen_id_A1_Tp[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide_predicted ==
locus_A1_Tp_c[[i]]])
    ensembl_peptide_id_A1_Tp[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted
== locus_A1_Tp_c[[i]]])
    description_A1_Tp[i] <- unique(t2g$description[t2g$refseq_peptide_predicted ==
locus_A1_Tp_c[i]])
    }
}

sleuth_significant_A1_Tp$hgnc <- hgnc_A1_Tp
for (i in 1:length(ensembl_gen_id_A1_Tp)) {
sleuth_significant_A1_Tp$ensembl_gen_id[i] <- paste(ensembl_gen_id_A1_Tp[[i]], collapse =  " ")
}
for (i in 1:length(ensembl_peptide_id_A1_Tp)) {
sleuth_significant_A1_Tp$ensembl_peptide_id[i] <- paste(ensembl_peptide_id_A1_Tp[[i]], collapse
=  " ")
}

sleuth_significant_A1_Tp$description <- description_A1_Tp


buscar_A1_Tp <- sleuth_significant_A1_Tp[is.na(hgnc_A1_Tp),][13:19]   # Nombre de proteinas que
no estan en hgnc
write.csv(buscar_A1_Tp, "resultados/buscar_A1_Tp")

## Wald Test
hgnc_A1_Tp_wt <- c()
ensembl_gen_id_A1_Tp_wt <- list()
ensembl_peptide_id_A1_Tp_wt <- list()
description_A1_Tp_wt <- c()
for (i in 1:length(locus_A1_Tp_c_wt)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A1_Tp_c_wt[i]]) == 0) {
    hgnc_A1_Tp_wt[i] <- NA
    ensembl_gen_id_A1_Tp_wt[[i]] <- NA
    ensembl_peptide_id_A1_Tp_wt[[i]] <- NA
```

```
    description_A1_Tp_wt[i] <- NA
  }
  else {
    hgnc_A1_Tp_wt[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A1_Tp_c_wt[i]])
    ensembl_gen_id_A1_Tp_wt[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide ==
locus_A1_Tp_c_wt[i]])
    ensembl_peptide_id_A1_Tp_wt[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
locus_A1_Tp_c_wt[i]])
    description_A1_Tp_wt[i] <- unique(t2g$description[t2g$refseq_peptide ==
locus_A1_Tp_c_wt[i]])
    }
}
for (i in 1:length(locus_A1_Tp_c_wt)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A1_Tp_c_wt[i]]) == 0) {
    hgnc_A1_Tp_wt[i] <- hgnc_A1_Tp_wt[i]
    ensembl_gen_id_A1_Tp_wt[[i]] <- ensembl_gen_id_A1_Tp_wt[[i]]
    ensembl_peptide_id_A1_Tp_wt[[i]] <- ensembl_peptide_id_A1_Tp_wt[[i]]
    description_A1_Tp_wt[i] <- description_A1_Tp_wt[i]
  }
  else {
    hgnc_A1_Tp_wt[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide_predicted ==
locus_A1_Tp_c_wt[i]])
    ensembl_gen_id_A1_Tp_wt[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide_predicted ==
locus_A1_Tp_c_wt[i]])
    ensembl_peptide_id_A1_Tp_wt[[i]] <-
unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted == locus_A1_Tp_c_wt[i]])
    description_A1_Tp_wt[i] <- unique(t2g$description[t2g$refseq_peptide_predicted ==
locus_A1_Tp_c_wt[i]])
    }
}
sleuth_significant_A1_Tp_wt$hgnc <- hgnc_A1_Tp_wt
for (i in 1:length(ensembl_gen_id_A1_Tp_wt)) {
sleuth_significant_A1_Tp_wt$ensembl_gen_id[i] <- paste(ensembl_gen_id_A1_Tp_wt[[i]], collapse =
" ")
}
for (i in 1:length(ensembl_peptide_id_A1_Tp_wt)) {
sleuth_significant_A1_Tp_wt$ensembl_peptide_id[i] <- paste(ensembl_peptide_id_A1_Tp_wt[[i]],
collapse =  " ")
}
sleuth_significant_A1_Tp_wt$description <- description_A1_Tp_wt

buscar_A1_Tp_wt <- sleuth_significant_A1_Tp_wt[is.na(hgnc_A1_Tp_wt),][12:18]   # Nombre de
proteinas que no estan en hgnc
write.csv(buscar_A1_Tp_wt, "resultados/buscar_A1_Tp_wt")

write.csv(sleuth_significant_A1_Tp_wt, "resultados/sleuth_significant_A1_Tp_wt.odt")
```

Comparación `Active2` con `Torpor`:

```
# Se introduce en sleuth_significant_A2_Tp el nombre del locus

## LRT
locus_A2_Tp <- c()
for (i in 1:nrow(sleuth_significant_A2_Tp)) {
  locus_A2_Tp <- c(locus_A2_Tp,

hitlength$sseqid[hitlength$X.qseqid==sleuth_significant_A2_Tp$target_id[i]])
}
sleuth_significant_A2_Tp$locus <- locus_A2_Tp

## Wald Test
locus_A2_Tp_wt <- c()
for (i in 1:nrow(sleuth_significant_A2_Tp_wt)) {
  locus_A2_Tp_wt <- c(locus_A2_Tp_wt,

hitlength$sseqid[hitlength$X.qseqid==sleuth_significant_A2_Tp_wt$target_id[i]])
}
sleuth_significant_A2_Tp_wt$locus <- locus_A2_Tp_wt

#Locus sin versioón

## LRT
locus_A2_Tp_c <- c()
for (i in 1:length(locus_A2_Tp)) {
  locus_A2_Tp_c[i] <- paste(strsplit(locus_A2_Tp[i], "")[[1]][1:(length(strsplit(locus_A2_Tp[i],
```

```
"")[[1]])-2)], collapse = "")
}
sleuth_significant_A2_Tp$locus_nv <- locus_A2_Tp_c

## Wald Test
locus_A2_Tp_c_wt <- c()
for (i in 1:length(locus_A2_Tp_wt)) {
  locus_A2_Tp_c_wt[i] <- paste(strsplit(locus_A2_Tp_wt[i],
"")[[1]][1:(length(strsplit(locus_A2_Tp_wt[i], "")[[1]])-2)], collapse = "")
}
sleuth_significant_A2_Tp_wt$locus_nv <- locus_A2_Tp_c_wt

# hgnc, ensembl_gen_id, ensembl_peptide_id, description

## LRT
hgnc_A2_Tp <- c()
ensembl_gen_id_A2_Tp <- list()
ensembl_peptide_id_A2_Tp <- list()
description_A2_Tp <- c()
for (i in 1:length(locus_A2_Tp_c)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A2_Tp_c[i]]) == 0) {
    hgnc_A2_Tp[i] <- NA
    ensembl_gen_id_A2_Tp[[i]] <- NA
    ensembl_peptide_id_A2_Tp[[i]] <- NA
    description_A2_Tp[i] <- NA
  }
  else {
    hgnc_A2_Tp[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A2_Tp_c[i]])
    ensembl_gen_id_A2_Tp[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide ==
locus_A2_Tp_c[i]])
    ensembl_peptide_id_A2_Tp[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
locus_A2_Tp_c[i]])
    description_A2_Tp[i] <- unique(t2g$description[t2g$refseq_peptide == locus_A2_Tp_c[i]])
    }
}
for (i in 1:length(locus_A2_Tp_c)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A2_Tp_c[i]]) == 0) {
    hgnc_A2_Tp[i] <- hgnc_A2_Tp[i]
    ensembl_gen_id_A2_Tp[[i]] <- ensembl_gen_id_A2_Tp[[i]]
    ensembl_peptide_id_A2_Tp[[i]] <- ensembl_peptide_id_A2_Tp[[i]]
    description_A2_Tp[i] <- description_A2_Tp[i]
  }
  else {
    hgnc_A2_Tp[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A2_Tp_c[i]])
    ensembl_gen_id_A2_Tp[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide_predicted ==
locus_A2_Tp_c[[i]]])
    ensembl_peptide_id_A2_Tp[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted
== locus_A2_Tp_c[[i]]])
    description_A2_Tp[i] <- unique(t2g$description[t2g$refseq_peptide_predicted ==
locus_A2_Tp_c[i]])
    }
}

sleuth_significant_A2_Tp$hgnc <- hgnc_A2_Tp
for (i in 1:length(ensembl_gen_id_A2_Tp)) {
sleuth_significant_A2_Tp$ensembl_gen_id[i] <- paste(ensembl_gen_id_A2_Tp[[i]], collapse =  " ")
}
for (i in 1:length(ensembl_peptide_id_A2_Tp)) {
sleuth_significant_A2_Tp$ensembl_peptide_id[i] <- paste(ensembl_peptide_id_A2_Tp[[i]], collapse
=  " ")
}

sleuth_significant_A2_Tp$description <- description_A2_Tp


buscar_A2_Tp <- sleuth_significant_A2_Tp[is.na(hgnc_A2_Tp),][13:19]   # Nombre de proteinas que
no estan en hgnc
write.csv(buscar_A2_Tp, "resultados/buscar_A2_Tp")

## Wald Test
hgnc_A2_Tp_wt <- c()
ensembl_gen_id_A2_Tp_wt <- list()
ensembl_peptide_id_A2_Tp_wt <- list()
description_A2_Tp_wt <- c()
```

```
for (i in 1:length(locus_A2_Tp_c_wt)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A2_Tp_c_wt[i]]) == 0) {
    hgnc_A2_Tp_wt[i] <- NA
    ensembl_gen_id_A2_Tp_wt[[i]] <- NA
    ensembl_peptide_id_A2_Tp_wt[[i]] <- NA
    description_A2_Tp_wt[i] <- NA
  }
  else {
    hgnc_A2_Tp_wt[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide == locus_A2_Tp_c_wt[i]])
    ensembl_gen_id_A2_Tp_wt[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide ==
locus_A2_Tp_c_wt[i]])
    ensembl_peptide_id_A2_Tp_wt[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
locus_A2_Tp_c_wt[i]])
    description_A2_Tp_wt[i] <- unique(t2g$description[t2g$refseq_peptide ==
locus_A2_Tp_c_wt[i]])
    }
}
for (i in 1:length(locus_A2_Tp_c_wt)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_A2_Tp_c_wt[i]]) == 0) {
    hgnc_A2_Tp_wt[i] <- hgnc_A2_Tp_wt[i]
    ensembl_gen_id_A2_Tp_wt[[i]] <- ensembl_gen_id_A2_Tp_wt[[i]]
    ensembl_peptide_id_A2_Tp_wt[[i]] <- ensembl_peptide_id_A2_Tp_wt[[i]]
    description_A2_Tp_wt[i] <- description_A2_Tp_wt[i]
  }
  else {
    hgnc_A2_Tp_wt[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide_predicted ==
locus_A2_Tp_c_wt[i]])
    ensembl_gen_id_A2_Tp_wt[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide_predicted ==
locus_A2_Tp_c_wt[i]])
    ensembl_peptide_id_A2_Tp_wt[[i]] <-
unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted == locus_A2_Tp_c_wt[i]])
    description_A2_Tp_wt[i] <- unique(t2g$description[t2g$refseq_peptide_predicted ==
locus_A2_Tp_c_wt[i]])
    }
}
sleuth_significant_A2_Tp_wt$hgnc <- hgnc_A2_Tp_wt
for (i in 1:length(ensembl_gen_id_A2_Tp_wt)) {
sleuth_significant_A2_Tp_wt$ensembl_gen_id[i] <- paste(ensembl_gen_id_A2_Tp_wt[[i]], collapse =
" ")
}
for (i in 1:length(ensembl_peptide_id_A2_Tp_wt)) {
sleuth_significant_A2_Tp_wt$ensembl_peptide_id[i] <- paste(ensembl_peptide_id_A2_Tp_wt[[i]],
collapse =  " ")
}
sleuth_significant_A2_Tp_wt$description <- description_A2_Tp_wt

buscar_A2_Tp_wt <- sleuth_significant_A2_Tp_wt[is.na(hgnc_A2_Tp_wt),][12:18]   # Nombre de
proteinas que no estan en hgnc
write.csv(buscar_A2_Tp_wt, "resultados/buscar_AA_Tp_wt")

write.csv(sleuth_significant_A2_Tp_wt, "resultados/sleuth_significant_A2_Tp_wt.odt")
```

Comparación `Active1` y `Active2` con `Torpor`:

```
# Se introduce en sleuth_significant_AA_Tp el nombre del locus

## LRT
locus_AA_Tp <- c()
for (i in 1:nrow(sleuth_significant_AA_Tp)) {
  locus_AA_Tp <- c(locus_AA_Tp,

hitlength$sseqid[hitlength$X.qseqid==sleuth_significant_AA_Tp$target_id[i]])
}
sleuth_significant_AA_Tp$locus <- locus_AA_Tp

## Wald Test
locus_AA_Tp_wt <- c()
for (i in 1:nrow(sleuth_significant_AA_Tp_wt)) {
  locus_AA_Tp_wt <- c(locus_AA_Tp_wt,

hitlength$sseqid[hitlength$X.qseqid==sleuth_significant_AA_Tp_wt$target_id[i]])
}
sleuth_significant_AA_Tp_wt$locus <- locus_AA_Tp_wt

#Locus sin versión
```

```r
## LRT
locus_AA_Tp_c <- c()
for (i in 1:length(locus_AA_Tp)) {
  locus_AA_Tp_c[i] <- paste(strsplit(locus_AA_Tp[i], "")[[1]][1:(length(strsplit(locus_AA_Tp[i],
"")[[1]])-2)], collapse = "")
}
sleuth_significant_AA_Tp$locus_nv <- locus_AA_Tp_c

## Wald Test
locus_AA_Tp_c_wt <- c()
for (i in 1:length(locus_AA_Tp_wt)) {
  locus_AA_Tp_c_wt[i] <- paste(strsplit(locus_AA_Tp_wt[i],
"")[[1]][1:(length(strsplit(locus_AA_Tp_wt[i], "")[[1]])-2)], collapse = "")
}
sleuth_significant_AA_Tp_wt$locus_nv <- locus_AA_Tp_c_wt

# Nombres hgnc, ensembl_gen_id, ensembl_peptide_id, description

## LRT
hgnc_AA_Tp <- c()
ensembl_gen_id_AA_Tp <- list()
ensembl_peptide_id_AA_Tp <- list()
description_AA_Tp <- c()
for (i in 1:length(locus_AA_Tp_c)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide == locus_AA_Tp_c[i]]) == 0) {
    hgnc_AA_Tp[i] <- NA
    ensembl_gen_id_AA_Tp[[i]] <- NA
    ensembl_peptide_id_AA_Tp[[i]] <- NA
    description_AA_Tp[i] <- NA
  }
  else {
    hgnc_AA_Tp[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide == locus_AA_Tp_c[i]])
    ensembl_gen_id_AA_Tp[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide ==
locus_AA_Tp_c[i]])
    ensembl_peptide_id_AA_Tp[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
locus_AA_Tp_c[i]])
    description_AA_Tp[i] <- unique(t2g$description[t2g$refseq_peptide == locus_AA_Tp_c[i]])
    }
}
for (i in 1:length(locus_AA_Tp_c)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_AA_Tp_c[i]]) == 0) {
    hgnc_AA_Tp[i] <- hgnc_AA_Tp[i]
    ensembl_gen_id_AA_Tp[[i]] <- ensembl_gen_id_AA_Tp[[i]]
    ensembl_peptide_id_AA_Tp[[i]] <- ensembl_peptide_id_AA_Tp[[i]]
    description_AA_Tp[i] <- description_AA_Tp[i]
  }
  else {
    hgnc_AA_Tp[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_AA_Tp_c[i]])
    ensembl_gen_id_AA_Tp[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide_predicted ==
locus_AA_Tp_c[[i]]])
    ensembl_peptide_id_AA_Tp[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted
== locus_AA_Tp_c[[i]]])
    description_AA_Tp[i] <- unique(t2g$description[t2g$refseq_peptide_predicted ==
locus_AA_Tp_c[i]])
    }
}

sleuth_significant_AA_Tp$hgnc <- hgnc_AA_Tp
for (i in 1:length(ensembl_gen_id_AA_Tp)) {
sleuth_significant_AA_Tp$ensembl_gen_id[i] <- paste(ensembl_gen_id_AA_Tp[[i]], collapse =  " ")
}
for (i in 1:length(ensembl_peptide_id_AA_Tp)) {
sleuth_significant_AA_Tp$ensembl_peptide_id[i] <- paste(ensembl_peptide_id_AA_Tp[[i]], collapse
=  " ")
}

sleuth_significant_AA_Tp$description <- description_AA_Tp


buscar_AA_Tp <- sleuth_significant_AA_Tp[is.na(hgnc_AA_Tp),][13:19]   # Nombre de proteinas que
no estan en hgnc
write.csv(buscar_AA_Tp_, "resultados/buscar_AA_Tp_")
```

```
## Wald Test
hgnc_AA_Tp_wt <- c()
ensembl_gen_id_AA_Tp_wt <- list()
ensembl_peptide_id_AA_Tp_wt <- list()
description_AA_Tp_wt <- c()
for (i in 1:length(locus_AA_Tp_c_wt)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide == locus_AA_Tp_c_wt[i]]) == 0) {
    hgnc_AA_Tp_wt[i] <- NA
    ensembl_gen_id_AA_Tp_wt[[i]] <- NA
    ensembl_peptide_id_AA_Tp_wt[[i]] <- NA
    description_AA_Tp_wt[i] <- NA
  }
  else {
    hgnc_AA_Tp_wt[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide == locus_AA_Tp_c_wt[i]])
    ensembl_gen_id_AA_Tp_wt[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide ==
locus_AA_Tp_c_wt[i]])
    ensembl_peptide_id_AA_Tp_wt[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
locus_AA_Tp_c_wt[i]])
    description_AA_Tp_wt[i] <- unique(t2g$description[t2g$refseq_peptide ==
locus_AA_Tp_c_wt[i]])
    }
}
for (i in 1:length(locus_AA_Tp_c_wt)) {
  if(length(t2g$hgnc_symbol[t2g$refseq_peptide_predicted == locus_AA_Tp_c_wt[i]]) == 0) {
    hgnc_AA_Tp_wt[i] <- hgnc_AA_Tp_wt[i]
    ensembl_gen_id_AA_Tp_wt[[i]] <- ensembl_gen_id_AA_Tp_wt[[i]]
    ensembl_peptide_id_AA_Tp_wt[[i]] <- ensembl_peptide_id_AA_Tp_wt[[i]]
    description_AA_Tp_wt[i] <- description_AA_Tp_wt[i]
  }
  else {
    hgnc_AA_Tp_wt[i] <- unique(t2g$hgnc_symbol[t2g$refseq_peptide_predicted ==
locus_AA_Tp_c_wt[i]])
    ensembl_gen_id_AA_Tp_wt[[i]] <- unique(t2g$ensembl_gene_id[t2g$refseq_peptide_predicted ==
locus_AA_Tp_c_wt[i]])
    ensembl_peptide_id_AA_Tp_wt[[i]] <-
unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted == locus_AA_Tp_c_wt[i]])
    description_AA_Tp_wt[i] <- unique(t2g$description[t2g$refseq_peptide_predicted ==
locus_AA_Tp_c_wt[i]])
    }
}
sleuth_significant_AA_Tp_wt$hgnc <- hgnc_AA_Tp_wt
for (i in 1:length(ensembl_gen_id_AA_Tp_wt)) {
sleuth_significant_AA_Tp_wt$ensembl_gen_id[i] <- paste(ensembl_gen_id_AA_Tp_wt[[i]], collapse =
" ")
}
for (i in 1:length(ensembl_peptide_id_AA_Tp_wt)) {
sleuth_significant_AA_Tp_wt$ensembl_peptide_id[i] <- paste(ensembl_peptide_id_AA_Tp_wt[[i]],
collapse =  " ")
}
sleuth_significant_AA_Tp_wt$description <- description_AA_Tp_wt

buscar_AA_Tp_wt <- sleuth_significant_AA_Tp_wt[is.na(hgnc_AA_Tp_wt),][12:18]   # Nombre de
proteinas que no estan en hgnc
write.csv(buscar_AA_Tp_wt, "resultados/buscar_AA_Tp_wt")
write.csv(sleuth_significant_AA_Tp_wt, "resultados/sleuth_significant_AA_Tp_wt.odt")
```

Se crea un vector con la descripción de los genes cuyo símbolo y códigos *Ensembl* no se han podido obtener con Biomart.

```
## LRT
unique(c(unique(proteinas_A2_Tp[is.na(hgnc_A2_Tp)]), unique(proteinas_A1_Tp[is.na(hgnc_A1_Tp)]),
unique(proteinas_A1_A2[is.na(hgnc_A1_A2)]), unique(proteinas_AA_Tp[is.na(hgnc_AA_Tp)])))

## Wals Test
unique(c(unique(proteinas_A2_Tp_wt[is.na(hgnc_A2_Tp_wt)]),
unique(proteinas_A1_Tp_wt[is.na(hgnc_A1_Tp_wt)]),
unique(proteinas_A1_A2_wt[proteinas_A1_A2_wt])))
```

Se creara un vector con el código de los genes coincidentes en diferentes comparaciones.

```
# Genes diferencialmente expresados en A1_Tp y en A2_Tp

## LRT
coincidencias_A1Tp_A2Tp <- c()
for (i in 1:length(locus_A1_Tp)) {
```

```r
  if(table(locus_A1_Tp[i]==locus_A2_Tp)[[1]] != length(locus_A2_Tp)) {
    coincidencias_A1Tp_A2Tp <- c(coincidencias_A1Tp_A2Tp, locus_A1_Tp[i])
  }
}
# length(unique(coincidencias_A1Tp_A2Tp))

## Wald Test
coincidencias_A1Tp_A2Tp_wt <- c()
for (i in 1:length(locus_A1_Tp_wt)) {
  if(table(locus_A1_Tp_wt[i]==locus_A2_Tp_wt)[[1]] != length(locus_A2_Tp_wt)) {
    coincidencias_A1Tp_A2Tp_wt <- c(coincidencias_A1Tp_A2Tp_wt, locus_A1_Tp_wt[i])
  }
}
length(unique(coincidencias_A1Tp_A2Tp_wt))

# Genes diferencialmente expresados en A1_Tp y en A1_A2

## LRT
# coincidenciasA1Tp_A1A2 <- c()
# for (i in 1:length(locus_A1_Tp)) {
#   if(table(locus_A1_Tp[i]==locus_A1_A2)[[1]] != length(locus_A1_A2)) {
#     coincidenciasA1Tp_A1A2 <- c(coincidenciasA1Tp_A1A2, locus_A1_Tp[i])
#   }
# }
# length(unique(coincidenciasA1Tp_A1A2))
## Wald Test
coincidenciasA1Tp_A1A2_wt <- c()
for (i in 1:length(locus_A1_Tp_wt)) {
  if(table(locus_A1_Tp_wt[i]==locus_A1_A2_wt)[[1]] != length(locus_A1_A2_wt)) {
    coincidenciasA1Tp_A1A2_wt <- c(coincidenciasA1Tp_A1A2_wt, locus_A1_Tp_wt[i])
  }
}
length(unique(coincidenciasA1Tp_A1A2_wt))

# Genes diferencialmente expresados en A2_Tp y en A1_A2
## LRT
# coincidenciasA2Tp_A1A2 <- c()
# for (i in 1:length(locus_A2_Tp)) {
#   if(table(locus_A2_Tp[i]==locus_A1_A2)[[1]] != length(locus_A1_A2)) {
#     coincidenciasA2Tp_A1A2 <- c(coincidenciasA2Tp_A1A2, locus_A2_Tp[i])
#   }
# }
# length(unique(coincidenciasA2Tp_A1A2))

## Wald Test
coincidenciasA2Tp_A1A2_wt <- c()
for (i in 1:length(locus_A2_Tp_wt)) {
  if(table(locus_A2_Tp_wt[i]==locus_A1_A2_wt)[[1]] != length(locus_A1_A2_wt)) {
    coincidenciasA2Tp_A1A2_wt <- c(coincidenciasA2Tp_A1A2_wt, locus_A2_Tp_wt[i])
  }
}
length(unique(coincidenciasA2Tp_A1A2_wt))

# Genes diferencialmente expresados en A2_Tp y en AA_Tp
# LRT
coincidenciasA2Tp_AATp <- c()
for (i in 1:length(locus_A2_Tp)) {
  if(table(locus_A2_Tp[i]==locus_AA_Tp)[[1]] != length(locus_AA_Tp)) {
    coincidenciasA2Tp_AATp <- c(coincidenciasA2Tp_AATp, locus_A2_Tp[i])
  }
}
# length(unique(coincidenciasA2Tp_AATp))

## Wald Test
coincidenciasA2Tp_AATp_wt <- c()
for (i in 1:length(locus_A2_Tp_wt)) {
  if(table(locus_A2_Tp_wt[i]==locus_AA_Tp_wt)[[1]] != length(locus_AA_Tp_wt)) {
    coincidenciasA2Tp_AATp_wt <- c(coincidenciasA2Tp_AATp_wt, locus_A2_Tp_wt[i])
  }
}
length(unique(coincidenciasA2Tp_AATp_wt))

# Genes diferencialmente expresados en A1_Tp y en A2_Tp AA_Tp
```

```
## LRT
coincidencias_A1Tp_AATp <- c()
for (i in 1:length(locus_A1_Tp)) {
  if(table(locus_A1_Tp[i]==locus_AA_Tp)[[1]] != length(locus_AA_Tp)) {
    coincidencias_A1Tp_AATp <- c(coincidencias_A1Tp_AATp, locus_A1_Tp[i])
  }
}
# length(unique(coincidencias_A1Tp_AATp))

## Wald Test
coincidencias_A1Tp_AATp_wt <- c()
for (i in 1:length(locus_A1_Tp_wt)) {
  if(table(locus_A1_Tp_wt[i]==locus_AA_Tp_wt)[[1]] != length(locus_AA_Tp_wt)) {
    coincidencias_A1Tp_AATp_wt <- c(coincidencias_A1Tp_AATp_wt, locus_A1_Tp_wt[i])
  }
}
length(unique(coincidencias_A1Tp_AATp_wt))

# Genes diferencialmente expresados en AA_Tp y en A1_A2
## LRT
# coincidenciasAATp_A1A2 <- c()
# for (i in 1:length(locus_AA_Tp)) {
#   if(table(locus_AA_Tp[i]==locus_A1_A2)[[1]] != length(locus_A1_A2)) {
#     coincidenciasAATp_A1A2 <- c(coincidenciasAATp_A1A2, locus_AA_Tp[i])
#   }
# }
# length(unique(coincidenciasAATp_A1A2))

## Wald Test
coincidenciasAATp_A1A2_wt <- c()
for (i in 1:length(locus_AA_Tp_wt)) {
  if(table(locus_AA_Tp_wt[i]==locus_A1_A2_wt)[[1]] != length(locus_A1_A2_wt)) {
    coincidenciasAATp_A1A2_wt <- c(coincidenciasAATp_A1A2_wt, locus_AA_Tp_wt[i])
  }
}
length(unique(coincidenciasAATp_A1A2_wt))
```

Se creará un archivo por cada comparación con el identificador *Ensembl* de los genes diferencialmente expresados, otro con los genes sobre-expresados y otro con los genes infra-expresados en las comparaciones de estados. Se considera sobre-expresado o infra-expresado en el estado que aparece al final en el nombre del archivo; por ejemplo, en el archivo `sobrexpresados_A1_A2.txt` están los genes sobre-expresados en el estado Active2 cuando comparamos Active1 y Active2.

```
write(unique(unlist(strsplit(sleuth_significant_A1_Tp_wt$ensembl_gen_id, " "))),
"resultados/sleuth_significant_A1_Tp_wt.txt")
write(unique(unlist(strsplit(sleuth_significant_A1_A2_wt$ensembl_gen_id, " "))),
"resultados/sleuth_significant_A1_A2_wt.txt")
write(unique(unlist(strsplit(sleuth_significant_A2_Tp_wt$ensembl_gen_id, " "))),
"resultados/sleuth_significant_A2_Tp_wt.txt")
write(unique(unlist(strsplit(sleuth_significant_AA_Tp_wt$ensembl_gen_id, " "))),
"resultados/sleuth_significant_AA_Tp_wt.txt")


# A1_A2
nrow(sobrexpresados_A1_A2 <- sleuth_significant_A1_A2_wt[sleuth_significant_A1_A2_wt$b>0,])
write(unique(unlist(strsplit(sobrexpresados_A1_A2$ensembl_gen_id, " "))),
"resultados/sobrexpresados_A1_A2.txt")
nrow(infraexpresados_A1_A2 <- sleuth_significant_A1_A2_wt[sleuth_significant_A1_A2_wt$b<0,])
write(unique(unlist(strsplit(infraexpresados_A1_A2$ensembl_gen_id, " "))),
"resultados/infraexpresados_A1_A2.txt")
# A1_Tp
nrow(sobrexpresados_A1_Tp <- sleuth_significant_A1_Tp_wt[sleuth_significant_A1_Tp_wt$b>0,])
write(unique(unlist(strsplit(sobrexpresados_A1_Tp$ensembl_gen_id, " "))),
"resultados/sobrexpresados_A1_Tp.txt")
nrow(infraexpresados_A1_Tp <- sleuth_significant_A1_Tp_wt[sleuth_significant_A1_Tp_wt$b<0,])
write(unique(unlist(strsplit(infraexpresados_A1_Tp$ensembl_gen_id, " "))),
"resultados/infraexpresados_A1_Tp.txt")
# A2_Tp
nrow(sobrexpresados_A2_Tp <- sleuth_significant_A2_Tp_wt[sleuth_significant_A2_Tp_wt$b>0,])
write(unique(unlist(strsplit(sobrexpresados_A2_Tp$ensembl_gen_id, " "))),
"resultados/sobrexpresados_A2_Tp.txt")
nrow(infraexpresados_A2_Tp <- sleuth_significant_A2_Tp_wt[sleuth_significant_A2_Tp_wt$b<0,])
write(unique(unlist(strsplit(infraexpresados_A2_Tp$ensembl_gen_id, " "))),
"resultados/infraexpresados_A2_Tp.txt")
# AA_Tp
```

```
nrow(sobrexpresados_AA_Tp <- sleuth_significant_AA_Tp_wt[sleuth_significant_AA_Tp_wt$b>0,])
write(unique(unlist(strsplit(sobrexpresados_AA_Tp$ensembl_gen_id, " "))),
"resultados/sobrexpresados_AA_Tp.txt")
nrow(infraexpresados_AA_Tp <- sleuth_significant_AA_Tp_wt[sleuth_significant_AA_Tp_wt$b<0,])
write(unique(unlist(strsplit(infraexpresados_AA_Tp$ensembl_gen_id, " "))),
"resultados/infraexpresados_AA_Tp.txt")
```

Se podrán visualizar los resultados en `R shiny`.

Comparación `Active2` con `Active1`:

```
sleuth_live(so1_A1_A2)
```

Comparación `Active1` con `Torpor`:

```
sleuth_live(so1_A1_Tp)
```

Comparación `Active2` con `Torpor`:

```
sleuth_live(so1_A2_Tp)
```

Se representará un gráfico de componentes principales para cada par de estados.

```
plot_pca(so1, pc_x = 1, pc_y = 2, color_by = 'nom_esta', text_labels = T)
plot_pc_variance(so1)
```

Se representará un gráfico donde se verán las distribuciones de cuentas agrupadas en estado fisiológico e individuo para cada comparación de estados.

```
plot_group_density(so1_A1_A2, use_filtered = TRUE, units = "est_counts",
  trans = "log", grouping = setdiff(colnames(so1_A1_A2$sample_to_covariates),
  "sample"), offset = 1)

plot_group_density(so1_A1_Tp, use_filtered = TRUE, units = "est_counts",
  trans = "log", grouping = setdiff(colnames(so1_A1_A2$sample_to_covariates),
  "sample"), offset = 1)

plot_group_density(so1_A2_Tp, use_filtered = TRUE, units = "est_counts",
  trans = "log", grouping = setdiff(colnames(so1_A1_A2$sample_to_covariates),
  "sample"), offset = 1)
```

Se descargará el archivo con la información de los transcritos obtenidos en el articulo original que incluye una columna con el porcentaje de la proteína humana que suponen cada transcrito. Una vez descargado se descomprime.

```
cd datos
wget 'https://datadryad.org/stash/downloads/file_stream/692'
tar -vzxf 692
```

Se crea un `dataframe` con los datos descargados.

```
blast_original <- read.delim("/media/diosrael/Elements
SE/datos/blastx.outfmt6.w_pct_hit_length.sorted", stringsAsFactors=FALSE)
```

Se seleccionan los transcritos del estudio original con una longitud >70% de la longitud de la proteína homologa y se crea un archivo con la información para posteriormente ser analizada en linea.

```
blast_original_70 <- blast_original[blast_original$pct_hit_len_aligned>70,]
prot_original <-sort(unique(blast_original_70$sseqid))
write(prot_original, "resultados/genes_originales.txt")
```

Se introduce en el `dataframe` que contiene los datos de nuestro transcriptoma una columna con la identificación sin el numero que codifica la versión.

```
trans_70R

sseqid_nv <- c()
for (i in 1:nrow(trans_70R)) {
  sseqid_nv[i] <- paste(strsplit(trans_70R$sseqid[i],
"")[[1]][1:(length(strsplit(trans_70R$sseqid[i][1], "")[[1]])-2)], collapse = "")
}
trans_70R$sseqid_nv <- sseqid_nv
```

Se crea un archivo que contenga las identificaciones para proteínas de *Ensembl* de los transcritos del transcriptoma obtenido en nuestro ensamblado para compararlos en linea con los obtenidos en el articulo original.

```
ensembl_peptide_id_70 <- list()
for (i in 1:nrow(trans_70R)) {
  if(length(t2g$refseq_peptide[t2g$refseq_peptide == trans_70R$sseqid_nv[i]]) == 0) {
```

```
    ensembl_peptide_id_70[[i]] <- NA
  }
  else {
    ensembl_peptide_id_70[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide ==
trans_70R$sseqid_nv[i]])
    }
}

for (i in 1:nrow(trans_70R)) {
  if(length(t2g$refseq_peptide_predicted[t2g$refseq_peptide_predicted ==
trans_70R$sseqid_nv[i]]) == 0) {
    ensembl_peptide_id_70[[i]] <- ensembl_peptide_id_70[[i]]
  }
  else {
    ensembl_peptide_id_70[[i]] <- unique(t2g$ensembl_peptide_id[t2g$refseq_peptide_predicted ==
trans_70R$sseqid_nv[i]])
    }
}
genes_70 <- sort((unique(unlist(ensembl_peptide_id_70)))[-6])
write(genes_70, "resultados/genes_present.txt")
```