Trabajo fin de grado

Análisis de transcriptomas a resolución de célula única aplicados a la inferencia de trayectorias de diferenciación celular en tejidos complejos

Autor: *Santiago Royo Sierra*
Directores: *Pierpaolo Bruscolini y Joaquín Sanz Remón*

28 de junio de 2021

Índice

1. Introducción

En la última década, en genómica se ha producido una gran revolución técnica alrededor del desarrollo de las tecnologías de *Next-Generation-Sequencing* (NGS). El concepto de NGS sirve para englobar todas las tecnologías destinadas a llevar a cabo la secuenciación masiva a gran escala de cualquier ácido nucleico desarrolladas principalmente en los últimos 10-15 años. El proceso que realizan estas tecnologías se basa en la extracción de ácidos nucleicos, fragmentación de los mismos en cadenas típicamente pequeñas de oligo-nucleótidos[1], transcripción inversa, (en el caso del RNA-seq) y su posterior secuenciación. Esto es muy útil porque permite observar patrones de variación entre individuos, mutaciones somáticas (cáncer), etc, y lo que nos interesa en este caso, cuantificar fluctuaciones en la cantidad de ARN en células en respuesta a estímulos externos, o entre células de distinto tipo, es decir, medir la expresión génica. Así pues, el gran desarrollo de estas tecnologías ha provocado un abaratamiento y facilitación de la producción de datos genómicos de manera masiva [1] como se puede observar en la figura 1:

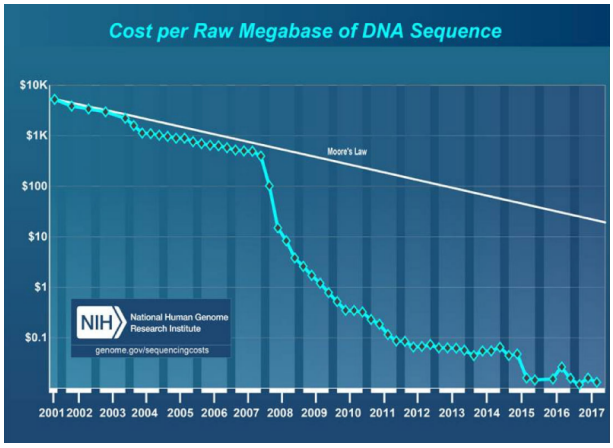

Figura 1: Coste por megabase de secuencia de ADN con el paso del tiempo. Fuente: National Institutes of Health-National Human Genome Research Institute [2].

Lo que llama la atención de esta figura 1 es que a partir de 2007-2008 aparece un descenso extremadamente acusado de los precios, el cual, dado que el eje Y está en escala logarítmica, se corresponde a una tasa de abaratamiento más que exponencial, la cual se reflejaría como una línea recta (como referencia la Ley de Moore) por lo que el coste se ha reducido a un ritmo enorme: el coste asociado a secuenciar un millón de nucleótidos ha pasado en 10 años de costar unos mil dólares a costar menos de uno.

Este abaratamiento ha traído consigo la generación de una gran cantidad y la complejidad de los datasets generados, lo cual, por un lado requiere de herramientas de análisis más sofisticadas, y por otro aumentan el rango de observabilidad de procesos biológicos cada vez más complejos y específicos. Es aquí donde la aplicación de modelos estadísticos resulta imprescindible para el análisis e interpretación de la dinámica de procesos biológicos, que se manifiesta, por ejemplo, a través de cambios en la expresión génica y otros fenotipos moleculares.

En este contexto de verdadera revolución tecnológica, una de las aplicaciones más recien-

[1] Un oligo-nucleótido es una secuencia corta de ADN o ARN

tes, y al mismo tiempo, revolucionarias, las constituyen las llamadas técnicas de secuenciación a resolución de célula única (*single-cell -omics*), que aplicada a la secuenciación de ARN (single-cell RNAseq, o scRNA-seq) apareció por primera vez en 2009 [3]. Esta nueva técnica de *scRNA-seq* se diferencia de las metodologías clásicas (comúnmente englobadas bajo el término "bulk RNA-seq") en que, por vez primera, a partir de muestras conteniendo cientos, o incluso miles de células, se puede identificar de qué célula individual procede cada fragmento secuenciado. Los métodos de secuenciación single-cell resultan especialmente ventajosos para estudiar tejidos heterogéneos, formados por células de varios tipos. En este contexto, si se pretende usar bulk RNA-seq, existen dos opciones: o se aislan los tipos celulares, lo cual modifica los fenotipos celulares al sacarlas del tejido y por tanto alejarlas de su estado "natural"; o se secuencian todas células juntas de manera que se tiene el nivel de expresión medio de cada gen en una gran población de células, pero se pierde toda la información relacionada con la heterogeneidad del tejido. Esto se soluciona con single-cell, ya que solo es necesario aislar las células casi finalizado el proceso por lo que la imagen que se capta es lo más parecido a las células en su estado "natural" y además, dado que se estudia la expresión génica de cada célula, es muy útil para el estudio de la heterogeneidad de respuestas celulares, la estocasticidad de la expresión génica y la inferencia de las redes de regulación génica en las células. Debido a esto, se está convirtiendo, también gracias a su abaratamiento, en una tecnología de referencia para estudiar patrones de variación entre células en tejidos complejos.

Si bien hoy en día existen múltiples plataformas experimentales para realizar experimentos en genómica single-cell, Dropseq es una de las tecnologías más populares. Esta metodología experimental consta de varios pasos: 1) se colocan las células del tejido a estudiar (con el grado de heterogeneidad que se requiera) en suspensión, y dicha suspensión se hace pasar a través de un canal de un dispositivo microfluídico, con la intención de que las células lo atraviesen, idealmente, de una en una. 2) Al flujo generado, se le añaden partículas "beads", cada una de las cuales porta copias de un oligo-nucleótido que es único en cada bead. Al encontrarse, las células y los "beads" forman gotículas por pares, dentro de las cuales se produce la lisis celular, transcripción inversa y etiquetado de los fragmentos resultantes con los oligo-nucleótidos de cada "bead", de tal modo que luego al secuenciarse todos los fragmentos a la vez, se puede "demultiplexar" los que proceden de cada célula. Esto, en transcriptómica, significa que somos capaces de estimar los niveles de expresión para cada célula por separado. El enfoque experimental propuesto por los creadores de DropSeq ha sido refinado posteriormente en otras plataformas experimentales, como inDrop o 10X Genomics, siendo ésta última la tecnología utilizada para generar los datos que analizaremos en este TFG. A continuación se muestra la figura donde aparece el proceso que acabamos de describir:

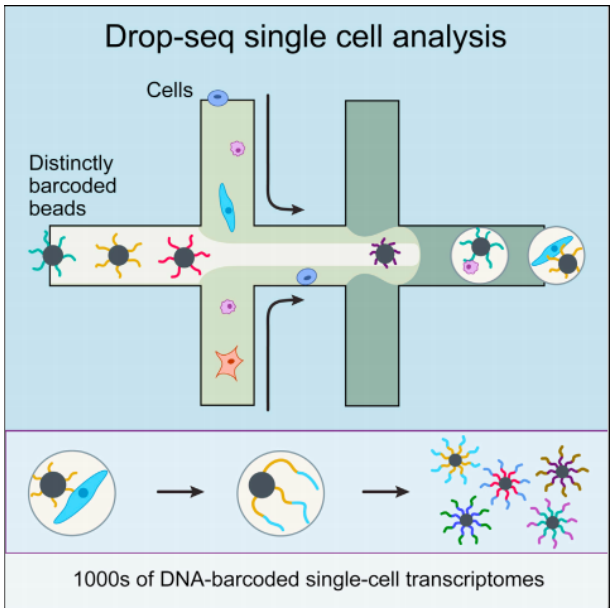

Figura 2: Estrategia Drop-seq para la generación de datos single-cell. Fuente: [4].

Una vez descritas muy someramente las técnicas experimentales empleadas para la generación de datos en genómica single-cell, se debe destacar una diferencia fundamental con los datos obtenidos mediante secuenciación bulk: en bulk, normalmente, los metadatos (i.e. los principales atributos de cada muestra) los conocemos antes de realizar el experimento, pero en single-cell, el metadato más importante de cada célula (es decir, el tipo celular si está bien definido, o el grado de diferenciación si hablamos de tejidos sometidos a cambios paulatinos entre fenotipos (por ejemplo células hematopoyéticas)) es, a priori, desconocido. Ello ha generado una gran necesidad de desarrollar métodos computacionales para inferir la identidad de las células. Nos encontramos ante dos enfoques: uno basado en proponer clusters de células (herramienta Seurat para R [5]) con transcriptomas similares que asignar a tipos celulares diferentes, y otro, basado en generar trayectorias, y asignar seudo-tiempos que localicen la posición de cada célula en las mismas (modelos de Monocle [6] y SCOUP [7]). El primer enfoque tiene la limitación de que depende de un parámetro arbitrario de resolución, que hace necesario contar con conocimiento externo previo para proponer clusters. El segundo ordena las células a lo largo de trayectorias, pero no ayuda, en muchos casos, a determinar de qué tipo son las células, de acuerdo a definiciones clásicas de los tipos celulares involucrados en el tejido.

En este trabajo, nos hemos dedicado a desarrollar un método para tratar de aportar información sobre la relación entre ambos enfoques. Para ello, lo que proponemos es, partiendo de una partición de clusters a una resolución arbitraria, inferir seudo-tiempos asociados a cada cluster, que después se puedan ordenar en una trayectoria global. De este modo, podemos obtener seudo-tiempos de desarrollo para cada cluster producido por una herramienta como Seurat, en lugar de a células individuales, proponiendo así, un enfoque intermedio entre los dos métodos más populares. Todo ello lo hemos hecho modelizando el proceso de desarrollo celular usando procesos de Ornstein-Uhlenbeck (OU), utilizando un método Bayesiano de inferencia de los parámetros del proceso. Esto ha sido realizado sobre un tejido de células madre hematopoyéticas, que es un tejido que se adecúa a la perfección al tipo de problema que nos interesa. Cabe decir que se trata de un objetivo ambicioso en el que se ha priorizado adquirir conocimiento en los

métodos y disciplinas involucradas.

Así, este trabajo presenta un doble objetivo:

- Por un lado, un objetivo fundamental ha sido la familiarización con los datos scRNA-seq, principalmente con las herramientas fundamentales para su pre-tratamiento, reducción de dimensionalidad y análisis.
- Por otro lado, hemos desarrollado e implementado un modelo que puede ser concebido como una herramienta para asignar coordenadas, en un árbol genérico de trayectorias de diferenciación celular, a clusters arbitrariamente pequeños de células. Para ello, se ha profundizado en conceptos avanzados de estadística (inferencia Bayesiana, priors conjugados), y física estadística (procesos de Ornstein-Uhlenbeck).

En lo sucesivo se explica:

1. Detalles de la generación de datos.
2. Como Seurat obtiene los clusters y subclusters.
3. Células utilizadas como datos en el trabajo.
4. El proceso Ornstein-Uhlenbeck, particularmente, en este sistema.

1.1. Generación de datos en scRNA-seq

De entre las diversas tecnologías disponibles para generar datos de scRNA-seq, las plataformas más populares, incluyendo Drop-seq, inDrop y 10XGenomics, (que es la metodología experimental que se usó para generar los datos que analizaremos en este trabajo) comparten una estrategia metodológica común para realizar perfiles transcriptómicos de miles de células individuales de manera rápida y eficiente, separándolas en gotas acuosas de tamaño nanométrico, asociando un código de barras diferente a los ARN de cada célula y secuenciándolos todos juntos [4]. En la siguiente figura 3, aparecen los pasos que se deben seguir desde la extracción de células del tejido hasta la generación de los datos que serán empleados en el posterior tratamiento.

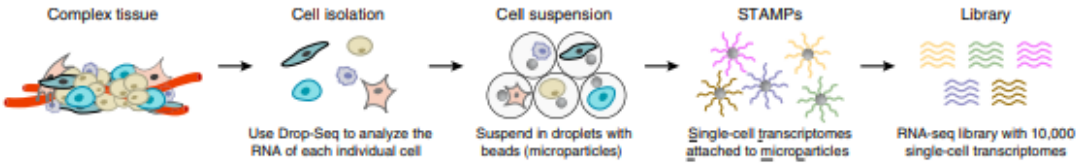

Figura 3: Generación de datos de manera secuenciada. Fuente: [4].

Como vemos en la figura 3, lo primero que se debe hacer una vez capturado un grupo de células de un tejido, es aislar cada una de ellas para analizar el ARN individualmente. A continuación, se suspenden en unas gotas en cuyo interior se encuentran también unas micropartículas ("beads") que contienen cebadores (primers) con códigos de barras. Estos códigos son los que permiten identificar cada célula y su origen en el estudio posterior. Cada célula es lisada para liberar su ARNm, los cuales se hibridan con los cebadores mencionados. Los ARNm se transcriben de

forma inversa a ADNc formándose los transcriptomas de célula única unidos a micropartículas (STAMPs-Single cell Transcriptomes Attached to Microparticles). Estos STAMPs con código de barras permiten inferir la célula origen de cada transcripción. Tras la formación de éstos, se realiza una amplificación por PCR (Polymerase Chain Reaction) para producir una biblioteca lo suficientemente grande, debido a la baja eficiencia de la transcripción inversa.

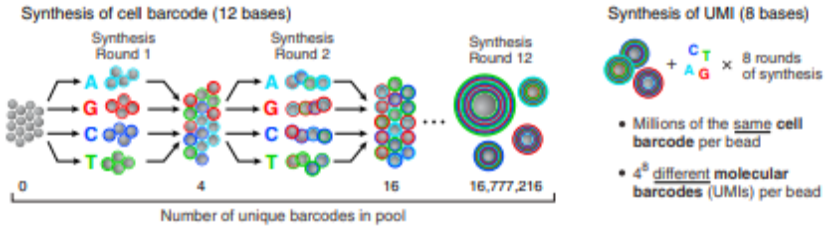

Figura 4: Generación del código de barras de manera secuenciada. Fuente: [4].

El proceso para generar el código de barras de la célula está descrito en la figura 4. El grupo de micropartículas es separado aleatoriamente en 4 subgrupos de igual tamaño y a cada uno se añade una de las 4 bases nitrogenadas del ADN (adenina (A), citosina (C), guanina (G), timina (T)) para posteriormente volver a juntar las micropartículas. Este ciclo se repite 12 veces, de manera que el resultado es un conjunto de micropartículas y cada una de las cuales posee una de las 4^{12} posibles secuencias en todo su complemento de cebadores. Así, el código de barras sintetizado en cualquier micropartícula refleja la trayectoria única de ésta a través de los ciclos. Después de esto, se realiza la síntesis de los identificadores moleculares únicos (UMI- Unique Molecular Identifier). Los UMIs son un tipo de código de barras molecular que se utiliza para corregir errores y aumentar la precisión durante la secuenciación. Esto se consigue gracias a que a cada cebador se le asocia una secuencia de código de barras que va a permitir identificarlo, de manera que tras la amplificación por PCR, estos UMIs van a permitir descartar aquellos duplicados que se deben a clones PCR y poder conservar los duplicados biológicos reales. En este proceso de fabricación de UMIs, todas las micropartículas juntas se someten a 8 rondas de síntesis degenerada con las 4 bases de ADN disponibles durante cada ciclo, de forma que cada cebador individual recibe una de las 4^8 posibles secuencias (UMIs).

Finalmente, la lectura de los códigos de barras celulares, de los UMIs y de las secuencias de ADNc para cada STAMP permite separar los fragmentos secuenciados según la célula individual de la que cada uno procede. De este modo, cuando se pretende cuantificar la expresión de cada gen, en cada célula, lo que se hace es mapear (o alinear), cada fragmento a una posición del genoma de referencia de la especie bajo estudio. De este modo, se puede cuantificar el número de fragmentos que, para cada célula, se asignan a posiciones genómicas asociadas a cada gen, lo que significa, de facto, obtener una estimación de los niveles de expresión de cada gen, en cada célula de cada muestra analizada, de manera independiente.

1.2. Enfoque para la identificación de células

Previamente a realizar el análisis que se lleva a cabo mediante el enfoque que aparece a continuación, es necesario realizar el tratamiento de datos, pero éste viene descrito en el apartado de Métodos, de manera que nos centraremos únicamente en este análisis mencionado.

1.2.1. Clustering

Seurat aplica un enfoque de clustering basado en grafos, donde la métrica de distancia que impulsa el análisis de agrupación se basa en las componentes principales (PCs) que han sido calculadas previamente en el análisis de componentes principales (PCA-Principal Component Analysis). Este enfoque basado en grafos toma como referencias trabajos previos ya aplicados a datos de *scRNA-seq* [8] y a datos de CyTOF (Cytometry by Time of Flight) [2][9]. Lo que hacen estos métodos es incorporar las células en un grafo de K-vecinos más cercanos (KNN- K-Nearest Neighbor), con enlaces entre las células con patrones de expresión similares, y luego dividen este grafo en comunidades altamente conexas.

En Seurat, primero se construye un grafo KNN que se basa en la distancia euclidiana en el espacio de componentes principales y se calculan los pesos de los enlaces entre células basándose en el solapamiento entre sus vecindades locales, comparando del primer al k-ésimo vecino más próximo de cada célula implicada en el enlace (similitud de Jaccard). La función en R que permite realizar este paso es *FindNeighbors* y toma como input el número de componentes principales a considerar.

Una vez definido el grafo KNN, para agrupar las células se aplican técnicas de optimización de la modularidad, como el algoritmo de Louvain. Así, se consigue un agrupamiento iterativo, con el objetivo de optimizar la función de modularidad estándar. La función en R que implementa esto es *FindClusters* y contiene un parámetro de resolución que establece la "granularidad" del clustering descendente, de manera que valores mayores para este parámetro conducen a un mayor número de clusters. Este parámetro es el parámetro arbitrario del que hablamos anteriormente. A este respecto, es importante apuntar que, aunque numerosos métodos han sido propuestos en ciencia de redes en los últimos años para producir particiones en comunidades "óptimas", que no dependan de un parámetro arbitrario de resolución, estos métodos no han sido aplicados mayoritariamente en el campo, en parte porque, a menudo, poder adaptar la resolución de las particiones, para asociar los clusters conseguidos a tipos celulares conocidos a priori con especificidad y precisión controlables durante el análisis resulta crucial.

1.3. El tejido: Células madre hematopoyéticas

En este trabajo hemos utilizado células del tejido hematopoyético. Estas células utilizadas reciben el nombre de células LKS, que son células madre del tejido hematopoyético, en nuestro caso, de los ratones. El dataset utilizado ha sido introducido como input a partir de [10], con la diferencia de que en nuestro caso, con la intención de simplificar los datos y el proceso posterior, las células a estudiar son células en "reposo", es decir, extraídas de sujetos experimentales no sometidos a ninguno de los tratamientos estudiados en [10]. Las células LKS, caracterizadas por expresar, simultáneamente los genes c-kit y Sca-1; y carecer de marcadores de linaje (lin(-)) se dividen en dos niveles jerárquicos: células hematopoyéticas propiamente dichas, y progenitores multipotentes (MPPs), derivados de las anteriores. Puesto que lo que se pretende con la utilización de este dataset es testear, a modo de prueba de concepto, la viabilidad de nuestro método,

[2]La citometría por tiempo de vuelo, o CyTOF, es una aplicación de la citrometría de masas que se utiliza para cuantificar las dianas marcadas en la superficie y el interior de las células individuales.

tras realizar el pretatamiento y normalización del data-set completo de celulas LKS en "reposo" estudiadas en [10], nos centraremos en estudiar, únicamente un tipo celular: el de las células madre hematopoyéticas de corto-plazo (HSC_ST, acrónimo de Hematopoietic Stem Cells, Short Term), y, más precisamente la dinámica de diferenciación de las mismas, desde el tipo celular "progenitor": las células madres hematopoyéticas de largo plazo (HSC_LT: Hematopoietic Stem Cells Long-Term).

1.4. Proceso de Ornstein-Uhlenbeck

Para inferir el tipo celular a partir de los datos de expresión génica, se ha desarrollado un modelo [11] inspirado en la herramienta SCOUP [7]. SCOUP relaciona los diferentes tipos celulares con un tipo celular "raíz", asumiendo que la evolución desde la raíz al tipo celular bien definido sea un proceso estocástico de Ornstein-Uhlenbeck (OU), que representa una variable moviéndose hacia un atractor con movimiento browniano. Se trata de un proceso de Gauss-Markov, por lo que cumple tanto las condiciones de proceso gaussiano como las de Markov, además de ser temporalmente homogéneo.

Un proceso OU X_t satisface la siguiente ecuación diferencial estocástica:

$$dX_t = -\alpha(X_t - \theta)dt + \sigma dW_t \tag{1}$$

donde α denota la fuerza de relajación hacia el atractor, θ el valor del atractor, σ la fuerza del ruido y W_t el "ruido blanco".

Este proceso podemos visualizarlo de manera más clara en la siguiente figura:

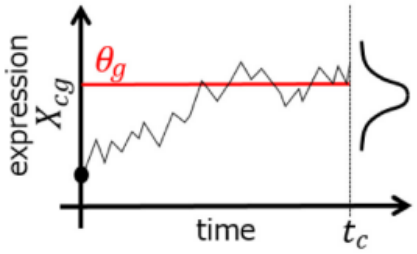

Figura 5: Diagrama conceptual del proceso OU. El proceso Ornstein-Uhlenbeck representa una variable $X_{c,g}$, es decir, la expresión de un gen g en una célula c, moviéndose hacia el atractor θ_g con movimiento browniano. Se observa que el valor a tiempo t satisface la distribución normal. Fuente: [7].

Si bien el planteamiento de SCOUP es interesante, también evidencia unas limitaciones, relacionadas con el hecho que impone un origen común: es decir, el arbol evolutivo es esencialmente una estrella, y no permite ramificaciones a partir de especies intermedias. En nuestro enfoque [11], tratamos de describir la diferenciación celular como una secuencia de procesos de OU, cada uno con origen en una célula, por ejemplo de tipo j, que suponemos está bastante cerca de su atractor θ_j en el momento de diferenciarse, que podemos asumir para ella la distribución final. La diferenciación provoca un cambio instantáneo de atractor, que pasa a ser el de tipo k;

sin embargo, el estado de la célula no cambia en el momento de la diferenciación, y la célula va evolucionando hacia el nuevo atractor en un tiempo que puede ser más largo de la propia vida celular (con que será su progenie la que llegará al nuevo atractor). Microscópicamente, podemos imaginar que el cambio de atractor corresponde a la instauración de un nuevo patrón en los marcadores de la cromatina, que sin embargo tarda a tener sus efectos sobre la dinámica de evolución celular.

2. Métodos

En esta sección se describe el tratamiento de datos previo que se ha llevado a cabo antes de introducir los datos al modelo y una descripción de éste junto con el método de implementación llevado a cabo del mismo.

2.1. Tratamiento de datos

Como parte esencial de cualquier modelización matemática de datos genómicos, es importante realizar un control de calidad exhaustivo, seguido de un pre-tratamiento de los datos riguroso, para identificar los patrones de variación debidos a causas técnicas y mitigarlos, eliminando genes y muestras afectados más intensamente por dichos problemas, o a través de otros procedimientos. Este proceso es fundamental para asegurar la calidad de los datos y por lo tanto de los resultados y conclusiones obtenidas de los mismos [12].

En lo que sigue, describiremos el proceso del tratamiento al que hemos sometido los datos crudos procedentes del estudio [10]. En la siguiente figura 6 se puede ver de manera secuencial los pasos seguidos durante este tratamiento:

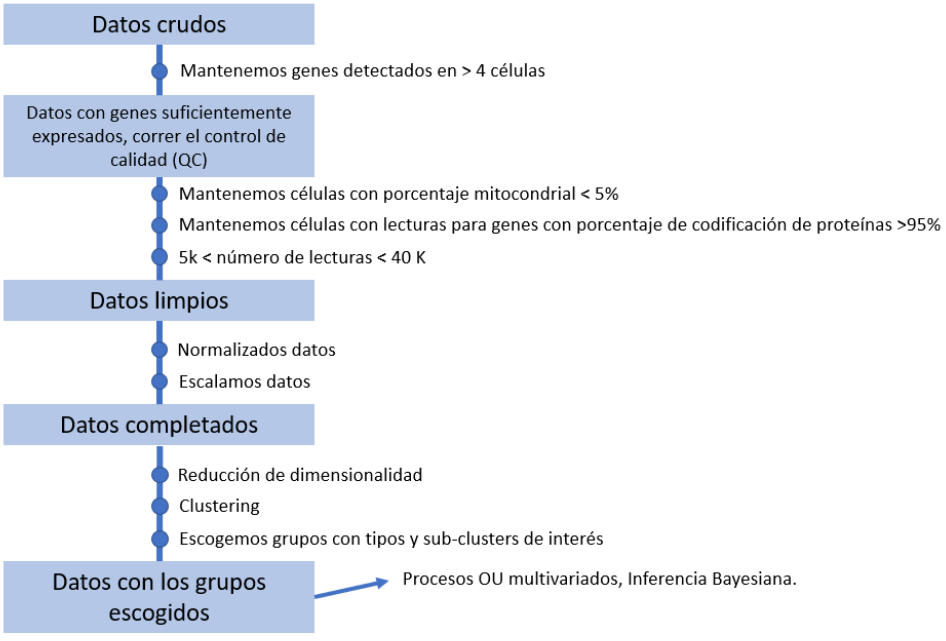

Figura 6: Esquema del tratamiento de datos.

Más específicamente, el proceso comienza con la matriz del número de fragmentos de RNA secuenciados correspondientes a cada gen y célula, que proviene del mapado de los propios fragmentos secuenciados en relación al genoma de referencia del ratón, durante el cual, primero se seleccionan los elementos (genes y células) que se conservarán para los análisis (tras un procedimiento de control de clalidad estandar, descrito en los anexos), para posteriormente proceder a transformarlos como se describe a continuación, antes de usarlos como inputs de nuestro modelo estadístico.

2.1.1. Normalización y escalado

El proceso de medida en un experimento de secuenciación consiste, en esencia, en un experimento de sampling de fragmentos de RNA de un pool (población) común (en este caso definido a nivel de célula única) donde todos los genes están representados en una proporción dada. Teniendo en cuenta lo anterior, la necesidad de normalizar los datos antes de cualquier integración de los mismos a través de varias muestras (células), es evidente. Los fragmentos observados en cada gen y célula no pueden compararse entre ellos si no, y esto es, esencialmente, por dos razones. En primer lugar, la profundidad de sampleado (profundidad de secuenciación) no es uniforme para células distintas; y por otro, aunque un gen se exprese al mismo nivel en dos muestras, la probabilidad de observarlo en un experimento de sampling aleatorio dependerá de la presencia, en mayor o menor cantidad, del resto de genes, esto es, de la composición de la librería.

Para mitigar dichos problemas y poder analizar datos conjuntos de muchas muestras, existen una multitud de métodos de normalización [13]. En nuestro caso, hemos utilizado el método implementado por J. Marioni y su equipo en el paquete de R *scRAN*, que es específico para el tratamiento de células individuales (scRNAseq) y cuyo método viene descrito en [14]. Se trata de una estrategia de de-convolución para normalizar la escala de células individuales cuando los niveles de expresión de éstas son pequeños y por tanto nuestros datos son altamente "sparse". El proceso que se sigue se basa en seleccionar iterativamente grupos de células y sumar los perfiles de expresión de las mismas. Estos perfiles de expresión agrupados se normalizan con respecto a una pseudocélula de referencia media, que ha sido construida promediando los recuentos de todas las células. Así, se define un coeficiente de normalización para el grupo de células como el valor esperado del cociente del valor esperado de las sumas de los recuentos del grupo y el de todos los genes. Con esto, se define un coeficiente de normalización para el grupo de células, las cuales no son tan "sparse" como si tratásemos células individuales, lo que da robustez al proceso. Esto lo vemos en la siguiente expresión obtenida del artículo mencionado previamente que describe este método [14]:

$$E(R_{i,k}) = \frac{E(V_{i,k})}{E(U_i)} \tag{2}$$

donde i son los genes, k hace referencia a que pertenece al grupo de células escogido S_k, $V_{i,k}$ es la suma de las expresiones de los genes a lo largo de todas las células que pertenecen al grupo S_k, U_i es la media de las expresiones de los genes a lo largo de todas las células del dataset y el valor esperado de $R_{i,k}$ será pues el factor de tamaño para el grupo de células S_k.

El coeficiente de escala para el grupo seleccionado es después concebido como una suma de contribuciones de las células constituyentes, de modo que dichos coeficientes a nivel de grupo pueden escribirse como una ecuación lineal en función de las contribuciones de cada célula. Re-

pitiendo este proceso para distintos grupos de células obtenemos un sistema lineal cuya solución nos permite obtener los factores de tamaño de las células individuales, resolviendo un problema de de-convolución. Esto lo podemos visualizar en la siguiente figura:

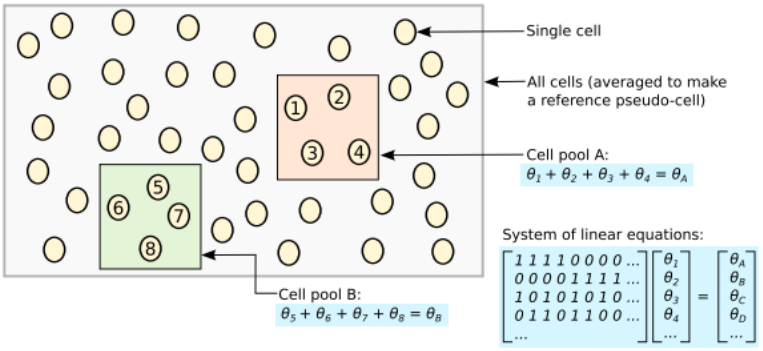

Figura 7: Esquema del método de deconvolución. Todas las células del dataset son promediadas para hacer la pseudocélula de referencia. Los valores de expresión de las células del grupo A son sumados y normalizados con respecto a la de referencia para obtener el factor de tamaño del grupo θ_A. Haciendo lo mismo para distintos grupos llegamos a un sistema lineal que permite calcular el factor de tamaño para cada célula. Fuente: [14].

La gran ventaja de éste método es que la suma de células por grupo permite, reduciendo la frecuencia de ceros en los datos, realizar estimaciones más precisas para los coeficientes de normalización de las células tras la de-convolución.

Una vez los datos han sido normalizados, el siguiente proceso consiste en eliminar de los mismos el efecto sistemático que las principales variables técnicas del dataset (interpretadas más arriba como métricas de calidad), ejercen sobre los valores, ya normalizados, de expresión. Para ello se realiza un ajuste lineal de los datos normalizados y se obtienen los residuos de dicho ajuste. Las variables a corregir que se han usado aquí, ya mencionadas arriba, son el porcentaje de RNA mitocondrial, de genes codificantes de proteinas y el número de fragmentos únicos (UMIs), todos ellos estandarizados para presentar media cero y desviación estandar unidad:

$$\hat{X}_i = \frac{X_i - \bar{X}}{sd(X)} \tag{3}$$

De modo que lo que se hace es estimar los siguientes modelos lineales, para el gen i, y la célula j

$$E_{i,j} = \beta_{o,i} + \sum_k \beta_{i,k} \cdot \hat{X}_{j,k} + \epsilon_{i,j} \tag{4}$$

Tras estimar los modelos lineales expresados en la anterior ecuación, se resta el efecto de las variables técnicas $\hat{X}$, para obtener una matriz de expresión escalada provisional:

$$E'_{i,j} = E_{i,j} - \sum_k \beta_{i,k} \cdot \hat{X}_{j,k} = \beta_{o,i} + \epsilon_{i,j} \tag{5}$$

caracterizada por el intercept de cada modelo, y la variación residual, a su alrededor, que no puede ser explicada como un efecto técnico asociado a las variables independientes anteriormente mencionadas.

En este momento, la variable $E'_{i,j}$ todavía presenta un problema técnico muy importante de heteroscedasticidad, es decir: la varianza residual de cada gen (varianza del vector de residuales ϵ) presenta un perfil de variación sistemático, que tiene un origen técnico, con respecto a la expresión media de cada gen (valor de los intercepts β_o), tal como se ve en la siguiente figura:

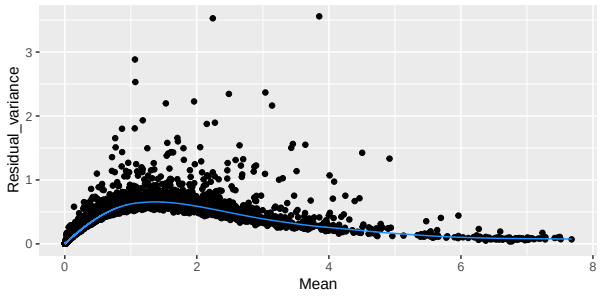

Figura 8: Representación de la varianza residual frente a la expresión media de cada gen.

Para corregir este problema, (lo que se denomina comúnmente estabilizar la varianza), se realiza un ajuste de la tendencia, a través de métodos de regresión local (por ejemplo, loess, splines, o simplemente rolling averages: serie azul en la anterior figura), y, para cada gen, se descompone la varianza como la suma de una componente técnica (siempre positiva: valor de la curva predicha para el gen, dada su expresión media) más un término de varianza biológica que resulta de la diferencia entre la varianza residual total y la varianza técnica. Utilizando dicha descomposición de la varianza es facil ver que, re-escalando los residuales de la ecuación 5 como sigue: $\hat{\epsilon}_{i,j} = \epsilon_{i,j}/\sigma_{tech}(\mu_o(i))$, y reconstruyendo el vector de expresiones así: $E''_{i,j} = \beta_{o,i} + \hat{\epsilon}_{i,j}$, se puede corregir, de facto, la relación sistemática de heteroscedasticidad, en su mayor parte, en los datos transformados. Esto ofrece una manera óptima de estimar, y poder comparar la varianza residual entre distintos genes, aspecto crucial para poder hacer una reducción de dimensionalidad que capture los patrones biológicos de variación, estando, a la vez, lo menos afectada posible por aspectos técnicos.

2.1.2. Reducción de dimensionalidad

El último paso del tratamiento de los datos es conocido como reducción de dimensionalidad. Este paso es muy relevante ya que se suele trabajar con grandes datasets (computacionalmente vienen representados como matrices de GxC) que contienen miles de genes. Cada uno de estos genes representa una dimensión de los datos, por lo que tendremos miles de dimensiones. Esto produce un gran coste computacional además de una difícil interpretación de los datos. Con la intención de simplificar esto, se realiza una reducción de dimensionalidad, donde destaca de entre los diferentes métodos el Análisis de Componentes Principales (PCA). Este algoritmo lineal de reducción consiste en definir jerárquicamente unas direcciones privilegiadas que deben ser las soluciones simultáneas a estos dos problemas de optimización:

- Las direcciones inferidas captan tanta variación como sea posible entre las variaciones observadas.
- Las direcciones inferidas minimizan los cuadrados de las distancias entre cada punto en $\mathbb{R}^g$ y las direcciones principales.

De esta manera, este análisis construye una transformación lineal que escoge un nuevo sistema de coordenadas donde cada eje será cada una de estas direcciones privilegiadas que recibirán el nombre de componentes principales. Estas componentes principales serán pues elegidas jerárquicamente de manera que la primera componente principal será la dirección que mayor varianza describa, la segunda será la dirección que describa la segunda mayor varianza y además sea ortogonal a la primera y así sucesivamente.

Así, cada componente principal será una combinación lineal de las variables originales y serán además independientes o no correlacionadas entre sí. Dicho de otra manera, el PCA puede considerarse como una rotación de los ejes del sistema de coordenadas de las variables originales a nuevos ejes ortogonales, de manera que estos ejes coincidan con la dirección de máxima varianza de los datos.

La siguiente ecuación desarrollada en [15] nos permite visualizar matemáticamente la descripción cualitativa realizada para las componentes principales:

$$Z_m = \sum_{n=1}^{n} \theta_{nm} G_n \tag{6}$$

donde Z_m es la componente principal, n es el número de variables, en este caso genes, G será el gen y θ_{nm} recibe el nombre de loading y da una idea del peso que tiene cada variable en cada componente. En nuestro caso, estos loadings hacen referencia a la rotación que se debe realizar sobre cada gen para obtener la componente principal.

Una decisión muy importante es la elección del número de componentes que vamos a calcular. Esto precisa de un equilibrio entre mantener la máxima información de los datos y reducir el coste computacional. Para ello, a menudo resulta útil mostrar gráficamente la desviación estándar representada por cada componente principal. Esto lo vemos en la figura siguiente:

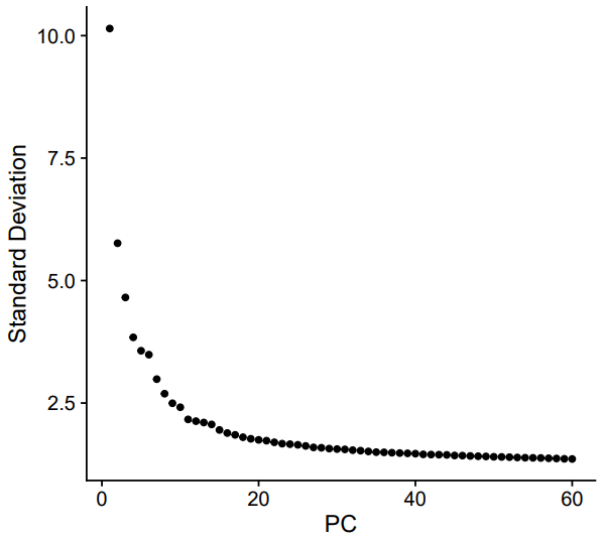

Figura 9: Representación gráfica de la desviación estándar frente a las componentes principales.

Como vemos, esta gráfica nos da una idea de cuantas componentes principales necesitamos para describir gran parte de la varianza. Vemos que a partir de 20, el decaimiento de la varianza

explicada es muy paulatino, y que, en comparación, las primeras componentes explican una mayor proporción de varianza. Dado que el número de componentes principales a utilizar definirá la dimensionalidad del problema (o sea, el número de parámetros), decidimos considerar sólo las 20 primeras, como un compromiso entre contener las componentes explicando las mayores cantidades de varianza y, al mismo tiempo, reducir el número de parámetros de lo que será nuestro modelo.

2.1.3. Clustering

El clustering ha sido realizado de la misma manera que lo realiza Seurat, como ha sido descrito en el apartado 1.2.1. En nuestro caso, de las anotaciones de cada tipo celular dentro de la población LKS, publicadas en [10], comprobamos que, pueden ser reproducidas usando una resolución $res = 0{,}2$. En concreto, a dicha resolución, como se ve abajo, encontramos un cluster para cada tipo celular, excepto para las células hematopoyéticas de corto plazo (ST_HSC), que quedan divididas en dos sub-clusters, tal como se ve en la siguiente figura. Ello nos permite encontrar en este tipo celular un ejemplo mínimo en el que poder explorar el método que se describirá a contuación, donde las células de un tipo quedan divididas en sub-clusters de expresión más homogénea. Se trata de dos sub-clusters, de 193 y 57 células, respectivamente: de modo que las matrices de datos a considerar presentan, finalmente, 193x20 y 57x20 (tantas filas como células observadas y tantas columnas como componentes principales).

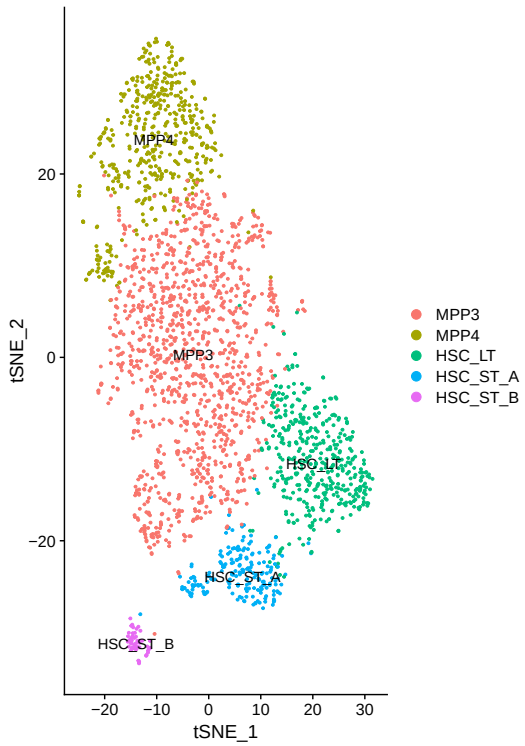

Figura 10: Representación del clustering mediante el método t-SNE(t-distributed Stochastic Neighbor Embedding). Éste es un método de reducción de dimensionalidad no lineal ampliamente usado en genómica, que a diferencia del PCA, no transforma las dimensiones originales en otras nuevas, sino que convierte realmente toda la información en todas las dimensiones en un plot bidimensional.

2.2. Modelo

2.2.1. Introducción

Nuestro enfoque se va a basar en interpretar el proceso de diferenciación de cada célula como un conjunto de transiciones consecutivas, sobre una topología de árbol que asumiremos conocida a priori. Así, cada tipo celular se interpretará como un nodo en dicho árbol, y las células transitarán a través del árbol modificando sus expresiones génicas de acuerdo a procesos de Ornstein-Uhlenbeck multi-variados. Para reducir la dimensionalidad del problema, en lugar de trabajar con matrices de expresión génicas, trabajaremos con matrices de componentes principales, considerando solo las G=20 primeras. Por tanto, nuestra matriz de datos es NxG, donde G es el número de componentes principales (20), y N el número de células.

Más específicamente, para todos los tipos celulares excepto para el tipo raíz, asumimos, que en cada instante t, cada célula, cuyo estado representamos con $X_c(t)$ en la base de componentes principales está en un punto distinto de un proceso de decaimiento de tipo Ornstein-Uhlenbeck, hacia un determinado atractor, que viene definido, en última instancia, por el tipo celular al que pertenece la célula en cuestión, el cual está matemáticamente caracterizado por los siguientes parámetros:

- Vector de valores de las componentes principales que caracterizan el atractor del tipo celular j: θ_j (dimensiones $1xG$)
- Matriz de covarianzas (en la base de componentes principales) alrededor del tipo j: ω_j (dimensiones GxG)

Suponemos también que una célula, cuando está suficientemente cerca del atractor de su tipo, puede empezar un proceso de diferenciación, que consiste en un cambio instantáneo del atractor hacia el del tipo "hijo", y la siguiente dinámica de relajación hacia el nuevo atractor. Si el decaimiento de una célula dada (es decir, el proceso de diferenciación celular), comenzó en un momento τ_c, en el que dicha célula tenía un vector de estado igual a $X_{c,0}$, el modelo asume que el decaimiento hacia el atractor del tipo celular "hijo" (tipo celular j al que pertenece la célula bajo análisis c), sigue una distribución de probabilidad de tipo Ornstein-Uhlenbeck, así que al tiempo de observación experimental t_c, cuando ha pasado un tiempo $\Delta t_c = t_c - \tau_c$ desde la diferenciación, la probabilidad de observar dicha célula el estado X_c será:

$$\begin{aligned} p(X_c|\Delta t_c, \alpha_j, \theta_j, w_j, X_{c,0}) = p_{OU}(X_c|\Delta t_c, \alpha_j, \theta_j, w_j, X_{c,0}) = \mathcal{N}(\mu_j(\Delta t_c), \Sigma_j(\Delta t_c)) = \\ = \frac{e^{-1/2(X_c-\mu_j(\Delta t_c))^T(\Sigma_j(\Delta t_c))^{-1}(X_c-\mu_j(\Delta t_c))}}{(2\pi)^{G/2}det(\Sigma_j(\Delta t_c))^{1/2}} \end{aligned} \tag{7}$$

que, como se ve, no es más que una normal multi-variada con vector de medias $\mu_j(\Delta t_c)$, y matriz de covarianzas $\Sigma_j(\Delta t_c)$ estimados, según sucede en un proceso OU a partir de una interpolación en un decaimiento exponencial multi-variado entre los valores asociados al estado inicial (atractor padre) y final (atractor hijo). En ese caso, la relación entre el estado de salida y el observado para una célula caracterizada por un seudo-tiempo igual a Δt_c es:

$$\begin{aligned} \mu_j(\Delta t_c) &= \theta_j - e^{-\alpha_j \Delta t_c}(\theta_j - X_{c,0}) \\ \Sigma_j(\Delta t_c) &= w_j - e^{-\alpha_j \Delta t_c} w_j \left(e^{-\alpha_j \Delta t_c}\right)^T \end{aligned} \tag{8}$$

El problema de este enfoque es que el estado inicial de cada célula al iniciarse la diferenciación es desconocido. Así que asumimos que el estado inicial de cualquier célula perteneciente al tipo j obedece a otra normal multi-variada, esta vez, en función de los parámetros que describen la distribución de datos para las células en el atractor asociado al tipo celular padre: $P(X_{c,o}) = \mathcal{N}(\theta_{\partial j}, w_{\partial j})$. De modo que, para obtener la probabilidad de observar, a seudo-tiempo Δt_c, la célula c, en estado X_c, integramos a lo largo de todos los posibles puntos de partida, así:

$$\begin{aligned} p(X_c|\Delta t_c, \Xi_j) &= \int p_{OU}(X_c|\Delta t_c, \alpha_j, \theta_j, w_j, X^{0,c}) p(X_{c,0}) dX_{c,0} \\ &= p_{MOU}(X_c|\Delta t_c, \alpha_j, \theta_j, w_j, \theta_{\partial j}, w_{\partial j}) \end{aligned} \tag{9}$$

con la distribución Ornstein-Uhlenbeck modificada definida como:

$$p_{MOU}(X_c|\Delta t_c, \alpha_j, \theta_j, w_j, \theta_{\partial j}, w_{\partial j}) = \mathcal{N}(\mu_j(\Delta t_c), \Sigma_j(\Delta t_c)) \tag{10}$$

donde ahora podemos escribir los valores esperados para los valores de las PCs (componentes principales) que caracterizan el atractor y la matriz de covarianza asociada a cada célula, dado el valor de su seudo-tiempo, en función de las matrices de transición α_j, y los valores asociados, en cada caso, a los atractores del tipo celular padre, y propio, de cada célula, como sigue:

$$\begin{aligned} \mu_j(\Delta t_c) &= \theta_j - e^{(-\alpha_j \Delta t_c)}(\theta_j - \theta_{\partial j}) \\ \Sigma_j(\Delta t_c) &= w_j - \left[e^{(-\alpha_j \Delta t_c)}\right](w_j - w_{\partial j})\left[e^{(-\alpha_j \Delta t_c)}\right]^T \end{aligned} \tag{11}$$

El resultado anterior para una célula c asume que sepamos a qué tipo celular pertenece. Una manera rigurosa de tratar la probabilidad global sería, pues, repartir las células entre clusters correspondientes al tipo celular, de forma arbitraria, y sumar sobre todas las particiones posibles. Sin embargo, para mayor simplicidad, escogemos solo la partición más probable, obtenida por clustering de las células en K diferentes tipos celulares.

Esto supone asumir que conocemos, a priori, qué células pertenecen a cada tipo celular, algo que, en nuestro caso es cierto, pues tenemos un buen conocimiento previo del tejido, al menos hasta este nivel.

A partir de las anteriores asunciones, estamos en condiciones de buscar una expresión para la likelihood global del sistema. Si asumimos, como se ha dicho, que la partición en tipos celulares es conocida a priori, podemos escribir que la probabilidad de que una célula cualquiera pertenezca al tipo j es simplemente $p(j) = N(j)/N$, donde $N(j)$ y N son los números de células en el tipo j, y en total. Por otra parte, dado que lo que nos interesa es proponer un método para estratificar los clusters asociados a cada tipo celular en sub-clusters más pequeños, los cuales queremos ordenar en el eje de seudo-tiempos, a partir de ahora asumiremos que todas las células en un mismo sub-cluster $\{j, a\}$, tienen exactamente el mismo seudo-tiempo:

$$\Delta t_{j,c} = \Delta t_{j,c'} \forall \{c, c'\} | c \in \pi_{j,a} \wedge c' \in \pi_{j,a} \tag{12}$$

Gracias a las anteriores asunciones, podemos escribir la likelihood del siguiente modo:

$$p(X|\Xi) = \prod_{j=1}^{n_j} \left[(N_j/N)^{N_j} \prod_{a=1}^{N_j^{cl}} \left(\prod_{c \in \pi_{j,a}} p_{MOU}(X_c|\Delta t_{j,a}, \alpha_j, \theta_j, w_j, \theta_{\partial j}, w_{\partial_j}) \right) \right] \tag{13}$$

A partir de esta expresión se puede observar que los productos a lo largo de todas las células pertenecientes a un mismo sub-cluster $\{j,a\}$ pueden agruparse como sigue:

$$P(X_{j,a}|\mu_j(\Delta t_{j,a}),\Sigma_j(\Delta t_{j,a})) = \prod_{c\in\pi_{j,a}} p_{MOU}(X_c|\Delta t_{j,a},\alpha_j,\theta_j,\omega_j,\theta_{\partial j},\omega_{\partial j}) = \frac{e^{-\frac{n_{j,a}}{2}Tr[C^{j,a}(\Sigma_j(\Delta t_{j,a}))^{-1}]}}{(2\pi)^{\frac{G}{2}}\det(\Sigma_j(\Delta t_{j,a}))^{\frac{1}{2}}} \tag{14}$$

donde hemos introducido la siguiente matriz de covarianzas:

$$\overline{C}^{j,a} = \frac{1}{n_{j,a}}\sum_{c\in\pi_{j,a}}(X_c-\mu_j(\Delta t_{j,a}))(X_c-\mu_j(\Delta t_{j,a}))^T \tag{15}$$

lo que implica que la distribución descrita en la ecuación 14 no es sino una normal multi-variada $\mathcal{N}(\mu_j(\Delta t_{j,a}),\Sigma_j(\Delta t_{j,a}))$ cuyo parámetro de medias es $\mu_j(\Delta t_{j,a})$ y el parámetro matriz de covarianza, es igual a $\Sigma_j(\Delta t_{j,a})$, evaluada en el vector promedio de componentes principales asociadas al sub-cluster $\{j,a\}$:

$$\overline{X}_{j,a} = \frac{1}{n_{j,a}}\sum_{c\in\pi_{j,a}} X_c \tag{16}$$

Todo ello nos permite reescribir la ecuación de la likelihood, de acuerdo a nuestro modelo, simplemente, como un producto de normales multi-variadas evaluadas en los vectores promedio de cada sub-cluster:

$$p(X|\Xi) \propto \prod_{j=1}^{n}\prod_{a=1}^{n_j}\mathcal{N}(\overline{X}_{j,a}|\mu_j(\Delta t_{j,a}),\Sigma_j(\Delta t_{j,a})) \tag{17}$$

donde, como ya se ha dicho n es el número de tipos celulares en el dataset, y n_j el número de sub-clusters en el tipo j. Finalmente, es crucial recordar que los parámetros μ_j y Σ_j asociados a los distintos tipos y sub-clusters presentes en los datos no son independientes, sino que están acoplados mediante el proceso OU, tal como se describe más arriba en las ecuaciones 11, las cuales, una vez recordamos que los seudo-tiempos son considerados comunes para todas las células de un mismo sub-cluster, quedan como sigue, para el sub-cluster $\{j,a\}$:

$$\begin{aligned}\mu_j(\Delta t_{j,a}) &= \theta_j - e^{(-\alpha_j\Delta t_{j,a})}(\theta_j-\theta_{\partial j})\\ \Sigma_j(\Delta t_{j,a}) &= w_j - \left[e^{(-\alpha_j\Delta t_{j,a})}\right](w_j-w_{\partial j})\left[e^{(-\alpha_j\Delta t_{j,a})}\right]^T\end{aligned} \tag{18}$$

En lo sucesivo, por simplicidad, escribiremos $\mu_{j,a}=\mu_j(\Delta t_{j,a})$ y $\Sigma_{j,a}=\Sigma_j(\Delta t_{j,a})$.

2.2.2. Estimación de los parámetros: teorema de Bayes sobre cada sub-cluster

En la sección anterior hemos enunciado las características del modelo matemático que usaremos para describir el estado de nuestro sistema, y, más concretamente, la probabilidad de que nuestro modelo, particularizado alrededor de unos valores específicos de los parámetros del modelo, nos genere un set dado de observaciones: en otras palabras, la likelihood de nuestro modelo.

A partir de aquí, necesitamos inferir los valores de dichos parámetros capaces de maximizar la probabilidad a posteriori, definida de acuerdo al teorema de Bayes:

$$p(\Xi|X) = \frac{p(X|\Xi)p(\Xi)}{\int d\Xi p(X|\Xi)p(\Xi)} \tag{19}$$

Si bien hasta este momento, nos hemos estado refiriendo a Ξ, como el set de parámetros independientes sobre el que ejecutar nuestra inferencia, en este momento es muy útil interpretar como parámetros provisionalmente independientes el set de promedios $\mu_{j,a}$, y covarianzas $\Sigma_{j,a}$ para enunciar la forma que el teorema de Bayes tiene, para nuestro problema, inicialmente de manera independiente, en cada sub-cluster $\{j, a\}$:

$$p(\mu_{j,a}, \Sigma_{j,a}|X_{j,a}) = \frac{P(X_{j,a}|\mu_{j,a}, \Sigma_{j,a})p(\mu_{j,a}, \Sigma_{j,a})}{\int d\mu'_{j,a} d\Sigma'_{j,a} \mathcal{N}(X_{j,a}|\mu'_{j,a}, \Sigma'_{j,a})p(\mu'_{j,a}, \Sigma'_{j,a})} \tag{20}$$

donde $P(X_{j,a}|\mu_{j,a}, \Sigma_{j,a}) = \mathcal{N}(\overline{X}_{j,a}|\mu_{j,a}, \Sigma_{j,a})$ es la likelihood de acuerdo con las ecuaciones 14-17. Partiendo de esta base, podemos ver que otra manera de describir nuestro objetivo es encontrar el set óptimo de parámetros μ y Σ que maximicen las probabilidades a posteriori de la ecuación anterior, teniendo en cuenta, como se verá más tarde, que no pueden maximizarse independientemente, puesto que dichos parámetros, como hemos visto antes, están ligados a través de la dinámica OU.

Pero antes de ello, es preciso definir el segundo ingrediente que, junto a las likelihoods, nos va a permitir escribir la anterior expresión de manera explícita, es decir el prior $p(\mu_{j,a}, \Sigma_{j,a})$. Para este prior, hemos optado por asumir que la $p(\mu_{j,a}, \Sigma_{j,a})$ de la ecuación 20 esté regida por una distribución normal-inverse-Wishart ($\mathcal{NIW}$), la cual puede entenderse como la distribución de probabilidad asociada a una variable aleatoria multi-variada cuyo vector de promedios μ obedece a una distribución normal multi-variada de media η y varianza Σ/γ; mientras que, a su vez, sus covarianzas Σ se distribuyen de acuerdo a una inverse-Wishart $\mathcal{W}^{-1}$ cuyos parámetros son la matriz de iguales dimensiones ψ y los grados de libertad ν, es decir si

$$\begin{aligned} p(\mu|\eta, \Sigma, \gamma) &= \mathcal{N}(\eta, \Sigma/\gamma) \\ p(\Sigma|\psi, \nu) &= \mathcal{W}^{-1}(\psi, \nu) \end{aligned} \tag{21}$$

entonces:

$$p(\mu, \Sigma|\eta, \psi, \gamma, \nu) = \mathcal{N}(\mu|\eta, \Sigma/\gamma)W^{-1}(\Sigma|\psi, \nu) = \mathcal{NIW}(\mu, \Sigma|\eta, \psi, \gamma, \nu) \tag{22}$$

cuya expresión explícita nos va a definir en última instancia nuestros priors sobre los parámetros μ y Σ en función de los hiper-parámetros del prior: $\{\eta, \psi, \gamma, \nu\}$:

$$\begin{aligned} p(\mu_{j,a}, \Sigma_{j,a}) &= \mathcal{NIW}(\eta_j, \gamma_j, \psi_j, \nu_j) \\ &= \frac{\gamma_j^{\frac{G}{2}} \det(\psi_j)^{\frac{\nu_j}{2}}}{(2\pi)^{\frac{G}{2}} 2^{\frac{\nu_j G}{2}} \Gamma_G(\frac{\nu_j}{2}) \det(\Sigma_{j,a})^{\frac{\nu_j+G+2}{2}}} e^{-\frac{\gamma_j}{2}(\mu_{j,a}-\eta_j)^T \Sigma_{j,a}^{-1}(\mu_{j,a}-\eta_j) - \frac{1}{2}Tr(\psi_j \Sigma_{j,a}^{-1})} \end{aligned} \tag{23}$$

La elección de este tipo de prior presenta múltiples ventajas.

En primer lugar, los parámetros η y ψ son relativamente sencillos de interpretar:

- η es el parámetro de localización de la distribución, esto es, el valor esperado de μ a priori. Por ello, si recordamos que estamos considerando sub-clusters que proceden de un tipo común, el prior razonable para este parámetro será el vector de componentes principales del atractor del padre, a partir del cual, las expresiones de los sub-clusters hijos divergerán, no sabemos a priori cuán rápidamente ni en qué dirección. Por ello: $\eta_j = \theta_{\partial j}$

- ψ, a su vez, nos da el valor esperado de Σ, a priori, a través de la relación: $\Sigma_{j,mean} = \psi_j/(\nu_j - G - 1)$, de modo que su valor a priori, razonando del mismo modo, se puede establecer para cada uno de los sub-clusters del tipo j, usando la información del padre: $\psi_j = \omega_{\partial j}(\nu_j - G - 1)$, lo que implica que el valor esperado a priori para Σ no es sino la covarianza del atractor del padre.

La segunda y más importante ventaja es que la distribución normal inverse Wishart es el prior conjugado de la distribución que define nuestras likelihoods, esto es, la normal multi-variada. Ello implica que el posterior definido en 20 (que define los multiplicandos del producto que queremos maximizar) se puede escribir, también, como una distribución $\mathcal{NIW}$, como sigue:

$$\begin{aligned} p(\mu_{j,a}, \Sigma_{j,a}|X_{j,a}) &= \mathcal{N}(\overline{X}_{j,a}|\mu_{j,a}, \Sigma_{j,a})\mathcal{NIW}(\mu_{j,a}, \Sigma_{j,a}|\eta_{j,a}, \gamma_{j,a}, \psi_{j,a}, \nu_{j,a}) \\ &= \mathcal{NIW}(\mu_{j,a}, \Sigma_{j,a}|\eta'_{j,a}, \gamma'_{j,a}, \psi'_{j,a}, \nu'_{j,a}) \end{aligned} \tag{24}$$

donde los parámetros de la distribución de posteriors, se relacionan con los de la distribución a priori, como sigue:

$$\begin{aligned} \eta'_{j,a} &= \frac{N_{j,a}\overline{X}_{j,a} + \gamma_j\eta_j}{N_{j,a} + \gamma_j} \\ \gamma'_{j,a} &= \gamma_j + N_{j,a} \\ \psi'_{j,a} &= \psi_j + N_{j,a}\overline{C}_{j,a} + \frac{N_{j,a}\gamma_j}{N_{j,a} + \gamma_j}(\overline{X}_{j,a} - \eta_j)(\overline{X}_{j,a} - \eta_j)^T \\ \nu'_{j,a} &= \nu_j + N_{j,a} \end{aligned} \tag{25}$$

donde

$$\begin{aligned} \overline{X}_{j,a} &= \frac{1}{N_{j,a}} \sum_{c \in \pi_{j,a}} X_c \\ \overline{C}_{j,a} &= \frac{1}{N_{j,a}} \sum_{c \in \pi_{j,a}} (X_c - \overline{X}_{j,a})(X_c - \overline{X}_{j,a})^T \end{aligned} \tag{26}$$

En cuanto a los dos parámetros escalares de la $\mathcal{NIW}$, γ y ν, son parámetros que controlan la dispersión de las distribuciones normal multi-variada (a la que obedece μ: primera eq. en 21), e inverse-Wishart (a la que obedece Σ: segunda eq. en 21), de modo que, a mayores valores de estos parámetros, los priors de medias y covarianzas son más estrechos (priors más informativos), y, a consecuencia, tal como se ve en 25, en el cálculo de las distribuciones a posteriori de los parámetros, tienen más peso las asunciones a priori (η, ψ), que las observaciones (X,C). Puesto que ambos parámetros presentan la misma tendencia (es decir, son inversamente proporcionales al grado de información que tienen los priors), es razonable ligarlos, por ejemplo, a través de la relación $\nu = G + \gamma - 1$, usada en [16], que, siempre que $\gamma > 0$ (propiedad imprescindible para γ en

esta distribución) garantiza que también se cumpla la otra condición indispensable concerniente a ν, ($\nu > G - 1$). En cuanto al valor específico que utilizar para γ, y por tanto para ν, en cada tipo y cluster, asumiremos, como primera aproximación, que γ es el mismo para cada sub-cluster del mismo tipo celular (de modo que hablamos de γ_j en realidad), y su valor lo estimaremos en un proceso de optimización global de la probabilidad a posteriori, tal como se explica en la siguiente sección:

2.2.3. Estimación de los parámetros: acoplamiento entre sub-clusters

Una vez que hemos establecido que las probabilidades a posteriori asociadas a observar un cierto par $(\mu_{j,a}, \Sigma_{j,a}|X_{j,a})$ en un cierto sub-cluster pueden obtenerse a través de distribuciones $\mathcal{NIW}$, la probabilidad a posteriori global, de acuerdo al teorema de Bayes (eq. 19) puede escrbirse como un producto de $\mathcal{NIW}s$:

$$p(\Xi|X) \propto \prod_{j=1}^{n}\prod_{a=1}^{n_j} \mathcal{NIW}(\mu_{j,a}, \Sigma_{j,a}|\eta'_{j,a}, \gamma'_{j,a}, \psi'_{j,a}, \nu'_{j,a}) = \prod_{j=1}^{n} \mathcal{P}_j \tag{27}$$

En este momento, es preciso recordar que, el conjunto de $\mu_{j,a}, \Sigma_{j,a}$ que se van a evaluar en cada uno de esos factores no son independientes, y por tanto, el ejercicio no es tan sencillo como estimar los valores de μ y Σ que, independientemente, maximizan dichos posteriors. Eso, obviamente, equivaldría a estudiar cada cluster por separado, ignorando el acoplamiento existente entre ellos, es decir, la dinámica de diferenciación.

En su lugar, lo que necesitamos hacer es recordar la ecuación 18:

$$\begin{aligned} \mu_{j,a} &= \theta_j - e^{(-\alpha_j \Delta t_{j,a})}(\theta_j - \theta_{\partial j}) \\ \Sigma_{j,a} &= w_j - \left[e^{(-\alpha_j \Delta t_{j,a})}\right](w_j - w_{\partial j})\left[e^{(-\alpha_j \Delta t_{j,a})}\right]^T \end{aligned} \tag{28}$$

y sustituir, desde 28, los valores de $\mu_{j,a}$, y $\Sigma_{j,a}$, en función de las matrices de transición α_j, los seudo-tiempos de cada sub-cluster $\Delta t_{j,a}$ y los parámetros del atractor del tipo j, (hijo) θ_j y ω_j, todos ellos desconocidos, en cada uno de los multiplicandos $\mathcal{P}_j$, de la eq. 27.

Al hacer esto, tenemos, para cada tipo celular, una expresión a maximizar, en función de una serie de objetos conocidos:

- Parámetros del atractor del tipo padre: $\theta_{\partial j}$ y $\omega_{\partial j}$
- Priors para los sub-clusters del tipo hijo: ψ_j y η_j, establecidos en función de los anteriores.
- Observaciones para los sub-clusters del tipo hijo: $\overline{X}_{j,a}$ y $\overline{C}_{j,a}$

Así como una serie de elementos a optimizar que son:

- Parámetros del atractor del tipo hijo j: θ_j y ω_j
- Matriz de transiciones OU: α_j
- Sets de seudo-tiempos para cada sub-cluster: $\Delta t_{j,a}$
- Parámetro escalar γ_j de la distribución $\mathcal{NIW}$, controlando la anchura de los priors, o sea, el grado de información asociado a los mismos, de manera global.

A este punto, es esencial darse cuenta de que el producto $\alpha_j \Delta t_{j,a}$ establece una sobreparametrización, si se pretende optimizar los dos objetos al completo, que equivale a la libertad de elegir unidades de seudo-tiempo arbitrarias. Para solucionar esto, fijaremos $\alpha_j(1,1) = 1$ en cada tipo j, para establecer una unidad de la escala temporal elegida. Por simplicidad, además, asumiremos que α_j es una matriz diagonal para todos los tipos celulares, lo cual es a priori sensato dado que trabajamos con componentes principales. Estas observaciones implican que α_j tiene en realidad $G-1$ grados de libertad, y que los seudo-tiempos estimados tendrán sólo valor relativo, y no comparable entre diferentes tipos celulares.

2.2.4. Estimación de los parámetros: primer tipo celular

En las anteriores secciones, hemos descrito el método a seguir para estimar decaimientos, seudo-tiempos, y propiedades del atractor asociadas a un tipo célular j, conociendo los valores del atractor padre, ∂j. Eso deja una obvia pregunta pendiente, a saber, cómo tratar el tipo celular raíz.

Nuestro enfoque consistirá pues en asumir que, para el tipo raíz, dado que no podemos establecer el mismo método que para el resto de tipos celulares, pues este no tiene "padre" y está formado por un único sub-cluster:

- θ_0 será la expresión media de las células de todo el cluster: $\theta_0 = \overline{X}_0$.
- ω_0 será la matriz de covarianza media de todo el cluster: $\omega_0 = \overline{C}_0$

3. Resultados

En esta sección se describen los resultados obtenidos con una implementación sencilla del modelo que aparece en la sección anterior, a modo de concepto y ejemplo para ver como funciona. Para ello, se ha utilizado el tipo celular HSC_ST, el cual está formado por tan solo dos subclusters, cuyo padre es HSC_LT. Así pues, la optimización de los parámetros, se ha llevado a cabo mediante la función de R *optim*, que nos permite optimizar varios parámetros al mismo tiempo de una misma función, a partir de unas condiciones iniciales. Se trata pues, de un ejercicio complejo que requiere de un coste computacional no despreciable dada la alta dimensionalidad del problema, cuya cardinalidad exacta es de $G(G+2)/5 + n_j$, que, en el ejemplo considerado asciende a 252 grados de libertad a optimizar (ratio datos/parámetros igual a 19.8).

Así, un primer objetivo fundamental es el de implementar el método y demostrar que un optimizador numérico más o menos estándar es capaz de navegar eficazmente en un espacio de parámetros de tan alta dimensionalidad. A continuación, se muestra la evolución de la función objetivo (igual a menos el logaritmo de la expresión $\mathcal{P}_j$, definida en la eq. 27, que debe, por tanto, ser minimizada), junto a la evolución de los parámetros a optimizar, ambos en cada iteración de *optim*, donde dichos parámetros son: $\gamma, \alpha, \Delta t, \omega, \theta$.

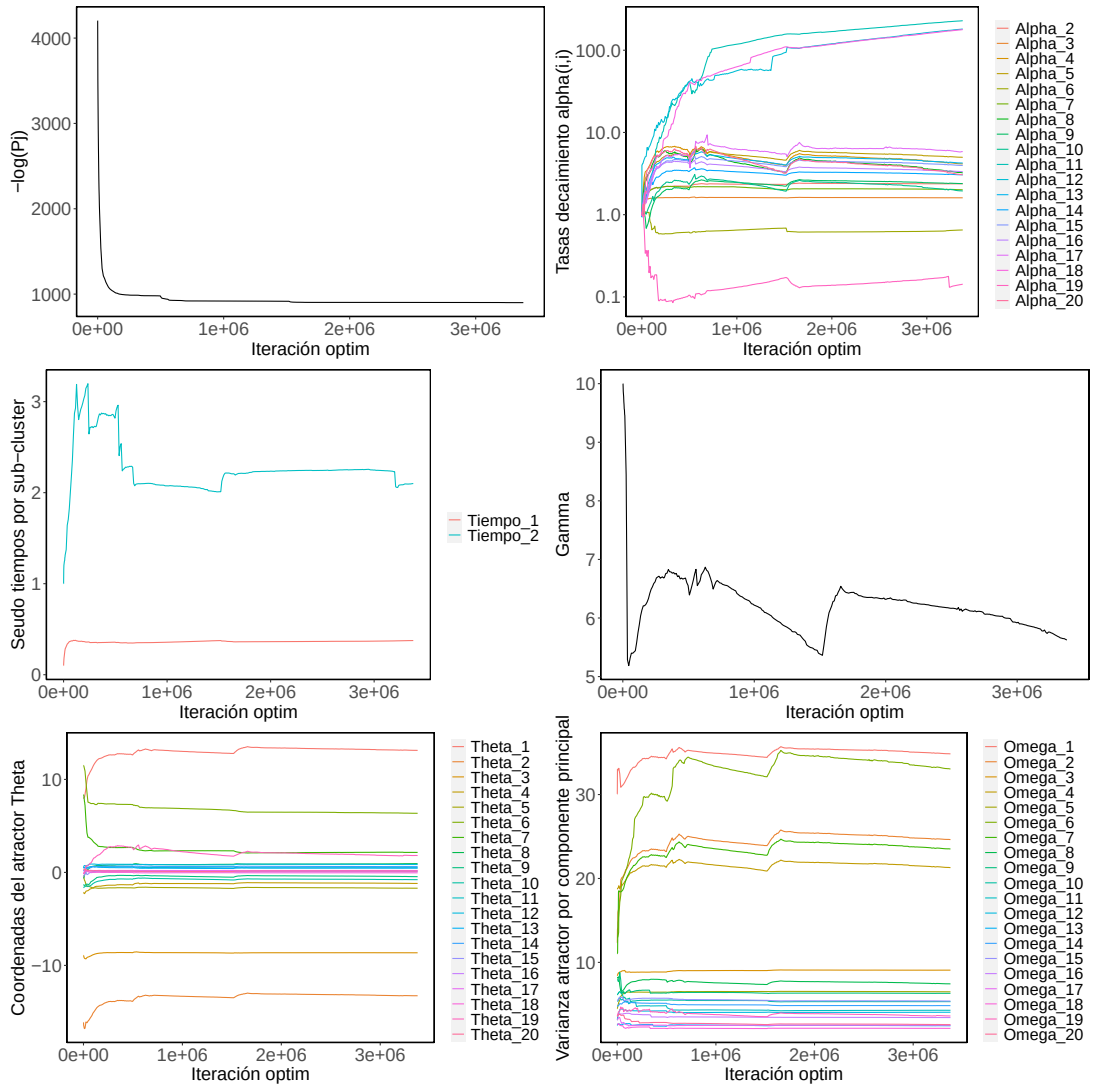

Figura 11: Evolución de los diferentes parámetros a optimizar frente al número de iteraciones realizadas por *optim*.

Tal como se ve en las gráficas anteriores, la función objetivo se estabiliza muy rápidamente en valores bajos, mientras que los parámetros mantienen la dinámica por más tiempo, llegan a converger, al menos cualitativamente, de manera aproximada, en la mayoría de los casos, presentando fluctuaciones (seudo-tiempo del sub-cluster 2 y γ) asociadas a saltos en la exploración del espacio de parámetros en el algoritmo, o tendencias residuales que no se lograron estabilizar por completo durante el proceso de optimización (α, notar escala logarítmica). El tiempo necesario para ejecutar el código fueron algo más de 12 horas en un ordenador de sobremesa.

Si en la anterior figura mostramos como el espacio de probabilidades del modelo puede ser explorado de un modo que, al menos superficialmente, parece satisfactorio, todavía tenemos que evaluar la calidad del ajuste propuesto por la solución óptima encontrada. Para ello, en la figura siguiente, hemos representado, utilizando la función *covEllipses*, del paquete de R *heplots*, las elipses asociadas a los términos de varianza y covarianza en el sub-espacio de las dos primeras componentes principales, que están asociadas tanto a los datos observados (objetos $\overline{X}$, y $\overline{C}$, en líneas discontínuas), como a las estimaciones proporcionadas por el modelo (objetos μ y Σ,

elipses sólidas azul y roja). Por último, hemos representado también las elipses asociadas a las estimaciones de los atractores padre, e hijo (objetos θ y ω, en elipses sólidas grises), que enmarcan la trayectoria de diferenciación:

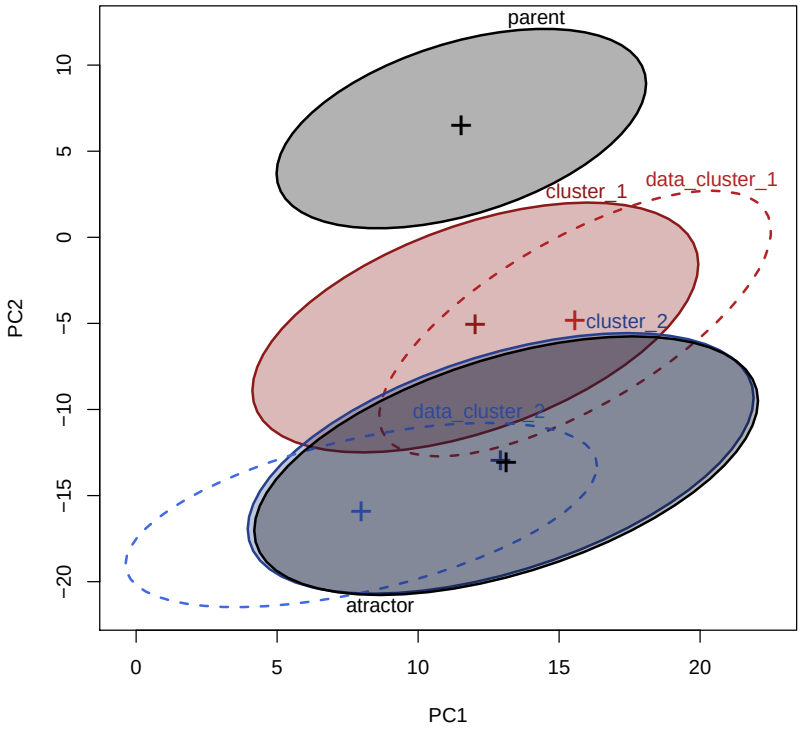

Figura 12: En esta figura representamos los datos de la segunda componente principal, frente a la primera, tanto para el sub-cluster 1, como para el sub-cluster 2, el tipo celular padre y el atractor estimado. Las elipses con líneas discontinuas representan los datos reales de los sub-clusters (rojo para el sub-cluster 1 y azul para el sub-cluster 2), y de entre las elipses con líneas continuas tenemos: la elipse gris superior que representa los datos del padre, la elipse gris inferior que representa los datos estimados para el atractor por el modelo y las elipses roja (para el sub-cluster 1) y azul (para el sub-cluster 2) son los datos estimados por el modelos para los sub-clusters. Las cruces representan las posiciones de las medias de los datos.

Como vemos en la figura 12, los datos estimados para ambos sub-clusters presentan un comportamiento cualitativamente coherente con la posición y covarianza de las nubes de puntos de los datos reales, sugiriendo una trayectoria de diferenciación (sucesión de los centros de las elipses sólidas roja y azul, hacia el atractor hijo) que, de algún modo, es una interpolación entre los datos observados para cada caso. Pese a que la observación anterior es cierta, resulta evidente que la trayectoria inferida, así como la evolución de las covarianzas a lo largo de ella, es muy poco flexible comparadas con los datos observados. Esto se debe, muy probablemente, a que, con la intención de reducir la dimensionlidad del problema, se ha impuesto una matriz α diagonal, lo que impide la flexibilidad de la evolución de las elipses en el espacio de componentes principales y por lo tanto nos hace observar una trayectoria rígida, de la que los datos se desvían, a un lado y al otro, significativamente.

4. Conclusiones

En la realización de este TFG he podido familiarizarme, al mismo tiempo, con aspectos experimentales en un campo como la genómica computacional, y al mismo tiempo, derivar, e implementar computacionalmente, un modelo basado en teoría de sistemas Brownianos, que hemos parametrizado usando inferencia Bayesiana. Aunque, por razones obvias, el nivel de profundidad que hemos cubierto en cada aspecto es limitado, la combinación de campos estudiados y técnicas empleadas alrededor de un problema común ha resultado satisfactoria. Como resultado del mismo, hemos propuesto una metodología para estimar seudo-tiempos asociados a sub-clusters de células de un tipo dado cuyos patrones de expresión sean relativamente homogéneos. Por supuesto, la implementación aquí propuesta solo puede entenderse como prueba de concepto del enfoque propuesto, y existen un número de modificaciones, y extensiones que podríamos, o deberíamos acometer si pretendemos llevar esta idea más allá.

En primer lugar, un requisito fundamental sería comprobar la robustez del algoritmo de fitting frente a diferentes condiciones iniciales de exploración del espacio de parámetros, que no hemos podido asegurar en el contexto de este TFG. Sería asimismo, muy conveniente testear la robustez de nuestros resultados ante cambios en parámetros del pipeline de pre-tratamiento de los datos, tales como el número de PCs consideradas, o el uso de umbrales diferentes en el control de calidad, etc. Además, sería a buen seguro interesante explorar otros métodos de optimización como *Simulated Annealing* o, eventualmente, convertir el problema a un problema de mínimos cuadrados entre μ y Σ frente a las modas de las NIW, en lugar de plantear el problema como una maximización de las $\mathcal{P}_j$. Otro paso sería solventar la limitación de utilizar α no diagonales, ya que como se ha visto en la figura 12, esto proporciona trayectorias de padre a hijo muy rígidas.

Una vez estos ejercicios técnicos estuviesen completos, sería muy interesante aplicar esta técnica a todo el dataset de [10], por ejemplo, y explorar la significancia biológica de los resultados.

Referencias

[1] Michael L. Metzker. Sequencing technologies the next generation. *Nature Reviews Genetics*, 11(1):31–46, 2010.

[2] Michael Chaldwell. We think genomics is a huge deal. *https://www.driehaus.com/perspectives/we-think-genomics-is-a-huge-deal*, 2019.

[3] Tang F. et al. mRNA-Seq whole-transcriptome analysis of a single cell. *Nature Methods*, 6(5):377–382, 2009.

[4] Macosko E.Z. et al. Highly parallel genome-wide expression profiling of individual cells using nanoliter droplets. *Cell*, 161(5):1202–1214, 2015.

[5] Rahul Satija, Jeffrey A. Farrell, David Gennert, Alexander F. Schier, and Aviv Regev. Spatial reconstruction of single-cell gene expression data. *Nature Biotechnology*, 33(5):495–502, 2015.

[6] Cole Trapnell, Davide Cacchiarelli, Jonna Grimsby, Prapti Pokharel, Shuqiang Li, Michael Morse, Niall J. Lennon, Kenneth J. Livak, Tarjei S. Mikkelsen, and John L. Rinn. The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells. *Nature Biotechnology*, 32(4):381–386, 2014.

[7] Hirotaka Matsumoto and Hisanori Kiryu. SCOUP: Probabilistic model based on the Ornstein-Uhlenbeck process to analyze single-cell expression data during differentiation. *BMC Bioinformatics*, 17(1):1–16, 2016.

[8] Chen Xu and Zhengchang Su. Identification of cell types from single-cell transcriptomes using a novel clustering method. *Bioinformatics*, 31(12):1974–1980, 2015.

[9] Jacob H Levine, Erin F Simonds, Sean C Bendall, Kara L Davis, El-ad D Amir, Michelle Tadmor, Oren Litvin, Harris Fienberg, Astraea Jager, Eli Zunder, Amanda L Gedman, Ina Radtke, James R Downing, Dana Pe, and P Garry. Data-Driven Phenotypic Dissection of AML Reveals Progenitor-like Cells that Correlate with Prognosis. 162(1):184–197, 2016.

[10] Nargis Khan, Jeffrey Downey, Joaquin Sanz, Eva Kaufmann, Birte Blankenhaus, Alain Pacis, Erwan Pernet, Eisha Ahmed, Silvia Cardoso, Anastasia Nijnik, Bruce Mazer, Christopher Sassetti, Marcel A. Behr, Miguel P. Soares, Luis B. Barreiro, and Maziar Divangahi. M. tuberculosis Reprograms Hematopoietic Stem Cells to Limit Myelopoiesis and Impair Trained Immunity. *Cell*, 183(3):752–770.e22, 2020.

[11] Pierpaolo Bruscolini. Cell differentiation as tree of Ornstein-Uhlenbeck processes. pages 1–7, 2021.

[12] Aisha A. AlJanahi, Mark Danielsen, and Cynthia E. Dunbar. An Introduction to the Analysis of Single-Cell RNA-Sequencing Data. *Molecular Therapy - Methods and Clinical Development*, 10(September):189–196, 2018.

[13] Catalina A. Vallejos, Davide Risso, Antonio Scialdone, Sandrine Dudoit, and John C. Marioni. Normalizing single-cell RNA sequencing data: Challenges and opportunities. *Nature Methods*, 14(6):565–571, 2017.

[14] Aaron T.L. Lun, Karsten Bach, and John C. Marioni. Pooling across cells to normalize single-cell RNA sequencing data with many zero counts. *Genome Biology*, 17(1):1–14, 2016.

[15] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *Springer Texts in Statistics An Introduction to Statistical Learning - with Applications in R*. 2013.

[16] Carlo Baldassi, Marco Zamparo, Christoph Feinauer, Andrea Procaccini, Riccardo Zecchina, Martin Weigt, and Andrea Pagnani. Fast and accurate multivariate Gaussian modeling of protein families: Predicting residue contacts and protein-interaction partners. *PLoS ONE*, 9(3):1–12, 2014.

[17] Tomislav Ilicic, Jong Kyoung Kim, Aleksandra A. Kolodziejczyk, Frederik Otzen Bagger, Davis James McCarthy, John C. Marioni, and Sarah A. Teichmann. Classification of low quality cells from single-cell RNA-seq data. *Genome Biology*, 17(1):1–15, 2016.

5. Anexos

5.1. Control de calidad de los datos

Como se observa en la figura 6, se parte de la matriz de fragmentos por gen y célula (es decir, los datos en crudo, una vez analizados a nivel de fragmentos individuales, y mapados sobre el genoma de referencia del ratón), introducidos como inputs. El primer paso que se realiza es la eliminación de los genes que no han podido ser detectados en un número mínimo de células (5), con el objetivo de eliminar todos aquellos que no van a ser suficientemente informativos. Tras filtrar genes cuya presencia no es informativa, en el siguiente paso procederemos al filtrado por células, de acuerdo a ciertas métricas de calidad de las mismas.

Los parámetros de control de calidad permiten separar las células de alta calidad (High Quality) de las células de baja calidad (Low Quality), que serán desechadas. Para ello, se pueden usar diversas métricas y enfoques. Nosotros nos fijaremos en tres métricas comúnmente usadas para establecer la calidad de las librerías celulares: el porcentaje de expresión asociada a genes mitocondriales, el porcentaje de RNA asociado a genes que codifican proteínas y el número de identificadores moleculares únicos (UMI-Unique Molecular Identifier) de cada una de ellas.

Así pues, descartaremos aquellas células que presenten un porcentaje de expresión mitocondrial por encima del 5 %. Esto se debe a que un porcentaje alto es un indicador de que las células están estresadas [17] o muertas y su ARNm citoplasmático ha podido filtrarse a través de una membrana rota y por tanto sólo se conserva el ARNm localizado en la mitocondria. También se descartan aquellas células cuyo porcentaje de genes que se manifiestan en ella con la función de codificación de proteínas es inferior al 95 % y las que presentan un número de UMIs por debajo de 5000, ya que esto se debe a células de baja calidad con pocos genes detectables, o por encima de 40000 ya que esto sería un indicativo de tener doublets, es decir, haber capturado dos células en lugar de una en el droplet y por lo tanto estaríamos rompiendo la condición de célula única. Con esto se consigue un criterio combinado, idealmente sin sesgo, para enriquecer las células a analizar en librerías de calidad adecuada y homogénea.

En la figura 13 se puede observar una representación gráfica del control de calidad aplicado sobre el dataset.

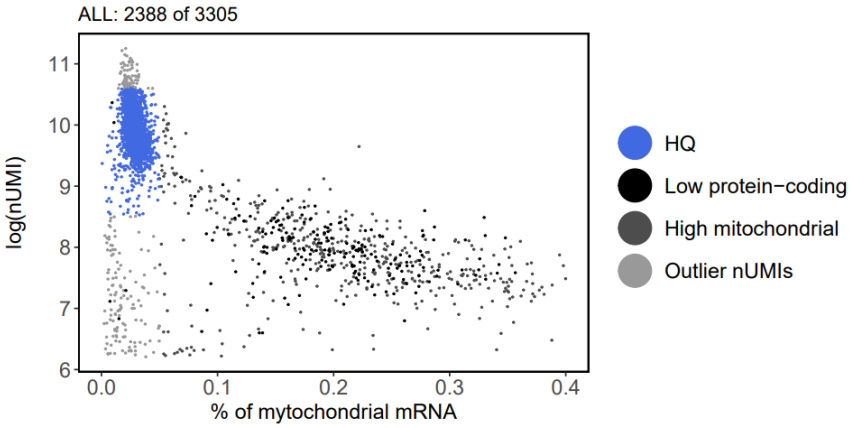

Figura 13: Representación gráfica del logaritmo del número de UMIs (es decir, número de fragmentos únicos secuenciados) frente al porcentaje de expresión mitocondrial de las células LKS. Coloreadas de azul son las células de alta calidad, de negro las células con un portentaje de codificación de proteínas inferior al umbral, de gris oscuro aquellas que presentan un porcentaje de expresión mitocondrial superior al umbral y de gris claro los outliers del número de UMIs. Así, 2388 de 3305 son células LKS HQ.

En la siguiente figura 14 aparecen únicamente las células de alta calidad tras realizar estos dos últimos pasos del control de calidad y se debe recalcar que será únicamente con éstas con las que se prosigue en el estudio. Estas representaciones nos sirven para confirmar que este proceso ha sido resuelto satisfactoriamente ya que todas células que en primera instancia habíamos concluido como células HQ han vuelto a mostrarse como tal en este segundo paso.

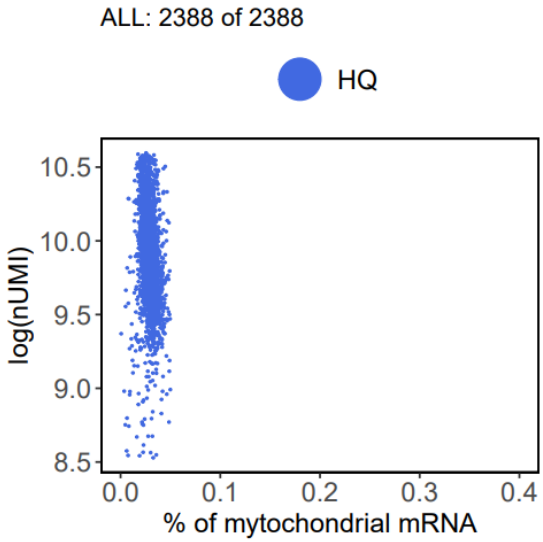

Figura 14: Representación gráfica del logaritmo del número de UMIs frente al porcentaje de expresión mitocondrial de las células LKS. Aparecen exclusivamente las células que conservamos por ser de alta calidad.

5.2. Código

A continuación se muestra el código realizado en la implementación.

Las librerías que se han utilizado han sido:

```
library(Seurat)
library(dplyr)
library(Matrix)
library(SingleCellExperiment)
library(GenomeInfoDb)
library(scran)
library(ggplot2)
library(cowplot)
library(scater)
library(limma)
library(LaplacesDemon)
library(HypergeoMat)
```

En primer lugar hemos cargado los datos y definido distintas funciones que calculan los parámetros a utilizar en el modelo según las ecuaciones descritas en la sección 2. Mencionamos aquí que data será una lista del tipo celular j donde cada elemento i de esa lista es la matriz del $sub-cluster_i$ de datos con la que trabajamos, es decir, la matriz embeddings del PCA.

```
load("pre_treatment_and_clustering_pruebasubclusters.RData")

ul=function(x){x[1:5,1:5]}
dcols=function(x){data.frame(colnames(x))}
options(width=1000)

get_X_bar=function(data){
x_bar=colMeans(data)
return(x_bar)
}
get_C_bar=function(data){

x_bar=get_X_bar(data)
nfilas=nrow(data)
cbarra<-vector("list", length=nrow(data))
fila=data
for(j in 1:nfilas){

fila=data[j,]-x_bar

cbarra[[j]]=(fila) %*% t(fila)

if(j>1)
cbarra[[j]]=cbarra[[j]]+cbarra[[j-1]]
}
```

```
cbarra=cbarra[[j]]
cbarra=cbarra/(nfilas)

return(cbarra)
}
get_psi_prime=function(psi,eta,gamma,N, C_bar, X_bar){
psi_prime=psi+N*C_bar+(((N*gamma)/(N+gamma))*((X_bar-eta)%*% t(X_bar-eta)))
return(psi_prime)
}
get_eta_prime=function(eta,gamma,N,X_bar){
eta_prime=((N*X_bar)+(gamma*eta))/(N+gamma)
return(eta_prime)
}
get_gamma_prime=function(gamma,N){
return(gamma+N)
}
get_nu_prime=function(nu,N){
return(nu+N)
}
get_mu=function(theta_parent,theta,alpha,tiempo){
mu=theta-expm(-alpha*tiempo)%*%(theta-theta_parent)
return(as.vector(mu))
}
get_sigma=function(omega_parent,omega,alpha,tiempo){
sigma=omega-expm(-alpha*tiempo)%*%(omega-omega_parent)%*%t(expm(-alpha*tiempo))
delta=max(abs(t(sigma)-sigma))
if(delta>0)
{
#digits=max(1,floor(-log10(delta))-2)
sigma=round(sigma,5)
}
return(as.matrix(sigma))
}
```

Cabe destacar, que los parámetros a optimizar han sido introducidos en "parámetros" pues es necesario que para que la función *optim* funcione correctamente, los parámetros a optimizar sean el primer argumento de la función a optimizar en forma de vector.

```
wrap_parameters=function(data){

data_binded=data[[1]]
for(i in 2:length(data))
data_binded=rbind(data_binded,data[[i]])
```

```
gamma_start=10
#gamma_start=mean(sapply(1:length(data),function(x){nrow(data[[x]])}))


theta_start=get_X_bar(data[[2]])
omega_start=get_C_bar(data[[2]])
omega_dofs_start=omega_start[upper.tri(omega_start,diag=TRUE)]

alphas_dofs_start=rep(1,19)
#tiempos_start=rep(1,length(data))
tiempos_start=c(0.1,1)
parameters=c(alphas_dofs_start,tiempos_start,gamma_start,theta_start,omega_dofs_start)
return(parameters)
}
rewrap_parameters=function(lista){


gamma_start=lista$gamma
theta_start=lista$theta
omega_start=lista$omega
omega_dofs_start=omega_start[upper.tri(omega_start,diag=TRUE)]

alphas_dofs_start=diag(lista$alphas)[2:G]
tiempos_start=lista$tiempos


parameters=c(alphas_dofs_start,tiempos_start,gamma_start,theta_start,omega_dofs_start)
return(parameters)
}
unwrap_parameters=function(parameters){

alphas=diag(G)
for(i in 2:G){
alphas[i,i]=parameters[i-1]
}
n_j=length(parameters)-G*(G+1)/2-2*G
tiempos=parameters[G:(G+n_j-1)]
gamma=parameters[G+n_j]
theta=parameters[(G+n_j+1):(G+n_j+G)]
omega_dofs_vect=parameters[(G+n_j+G+1):length(parameters)]
omega=diag(G)
omega[upper.tri(omega,diag=TRUE)]=omega_dofs_vect
t_omega=t(omega)
omega=omega+t_omega
```

```
omega[upper.tri(omega,diag=TRUE)]=omega_dofs_vect

output=list(alphas=alphas,tiempos=tiempos,gamma=gamma,theta=theta,omega=omega)
return(output)
}
get_parameters=function(data){
data_binded=data[[1]]
for(i in 2:length(data))
data_binded=rbind(data_binded,data[[i]])

alphas=diag(G)
tiempos=rep(1,length(data))
gamma=mean(sapply(1:length(data),function(x){nrow(data[[x]])}))
theta=get_X_bar(data_binded)
omega=get_C_bar(data_binded)
output=list(alphas=alphas,tiempos=tiempos,gamma=gamma,theta=theta,omega=omega)
return(output)
}
```

La siguiente función, es la que verdaderamente tiene peso pues utiliza las anteriores para calcular $-log(\mathcal{P}_j)$.

```
get_Pj=function(parametros,data,theta_parent,omega_parent){
glob<<-glob+1
if(glob%%100==0)print(glob)

pars_unwrapped=unwrap_parameters(parametros)
theta=pars_unwrapped$theta
omega=pars_unwrapped$omega
alpha=pars_unwrapped$alphas
tiempos=pars_unwrapped$tiempos
gamma=pars_unwrapped$gamma

#print(pars_unwrapped)
#print("esos eran los paramteros; pulsa key")
#y = keypress()

eta=theta_parent
nu=gamma+G-1
psi=omega_parent*(nu-G-1)
n_j=length(data)
```

```
sum_log_P_subclusters=0
for(a in 1:n_j)
{
X_bar=get_X_bar(data[[a]])
C_bar=get_C_bar(data[[a]])
N=nrow(data[[a]])

psi_prime=get_psi_prime(psi,eta,gamma,N, C_bar, X_bar)
eta_prime=get_eta_prime(eta,gamma,N,X_bar)
gamma_prime=gamma+N
nu_prime=nu+N

mu=get_mu(theta_parent,theta,alpha,tiempos[a])
sigma=get_sigma(omega_parent,omega,alpha,tiempos[a])
#print(mu)
#print(sigma)
#print(paste("esos eran mu y sigma del cluster ",a,"; pulsa key"))
#y = keypress()

if(FALSE){if(!isSymmetric(sigma)){
print("Sigma asimetrica, pulsa key:")
y = keypress()
sigma=round(sigma,digits=5)}}

append <- tryCatch(
{
dnorminvwishart(mu=mu, mu0=eta_prime,
lambda=gamma_prime, Sigma=sigma, S=psi_prime, nu=nu_prime, log=TRUE)
},
error = function(e){
NA
}
)

sum_log_P_subclusters=sum_log_P_subclusters+append
}
return(-sum_log_P_subclusters)
}
```

En esta última parte del código, obtenemos los valores para θ, ω del tipo celular padre y a partir de ahí obtenemos todos los parámetros del tipo celular hijo mediante la función *optim* y guardamos los resultados en una tabla para obtener los plots que se muestran en la sección de resultados.

```
###
### Cell type 0 (root): HSC_LT.
###

theta_HSC_LT=get_X_bar(HSC_LT)
omega_HSC_LT=get_C_bar(HSC_LT)

###
### Cell type 1: HSC_ST
###

data=HSC_ST
theta_parent=theta_HSC_LT
omega_parent=omega_HSC_LT
parameters=wrap_parameters(data)

#par_list=unwrap_parameters(result$par)
#par_listb=get_parameters(data)
#par_after_failed_first_fit=rewrap_parameters(parametros_HSC_ST)
param_mat=t(c(0,0,parameters,1))

glob=0
result<-optim(fn = get_Pj,control=list(maxit=1),
par=parameters,data=data,theta_parent=theta_parent,omega_parent=omega_parent)
param_mat=rbind(param_mat,t(c(glob,result$value,result$par,result$convergence)))
param_mat[1,2]=param_mat[2,2]

### Bucle para coger valores en la primera fase, mas dinamica
for(i in 1:10)
{
print(result)
print(unwrap_parameters(result$par))
result<-optim(fn = get_Pj,control=list(maxit=1500),
par=result$par,data=data,theta_parent=theta_parent,omega_parent=omega_parent)
param_mat=rbind(param_mat,t(c(glob,result$value,result$par,result$convergence)))
}
### Bucle para coger valores más a largo plazo
for(i in 1:200)
{
print(result)
print(unwrap_parameters(result$par))
result<-optim(fn = get_Pj,control=list(maxit=10000), par=result$par,data=data,theta
_parent=theta_parent,omega_parent=omega_parent)
param_mat=rbind(param_mat,t(c(glob,result$value,result$par,result$convergence)))
```

```
print(is.positive.definite(unwrap_parameters(result$par)$omega))
if(result$convergence==0)break
}
save(result,file="res_HSC_ST_fv.Rdata")
write.table(param_mat,"data_final_HSC_ST_fv.txt")
```