UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Desarrollo de un *pipeline* para análisis filogenético multi-locus

Trabajo de graduación presentado por Rafael Antonio León Pineda para optar al grado académico de Licenciado en Ingeniería en Bioinformática

Guatemala,

2024

UNIVERSIDAD DEL VALLE DE GUATEMALA
Facultad de Ingeniería

Desarrollo de un *pipeline* para análisis filogenético multi-locus

Trabajo de graduación presentado por Rafael Antonio León Pineda para optar al grado académico de Licenciado en Ingeniería en Bioinformática

Guatemala,

2024

Vo.Bo.:

(f) _____
Lic. Alejandro Vásquez

Tribunal Examinador:

(f) _____
Lic. Alejandro Vásquez

(f) _____
MSc. Douglas Barrios

(f) _____
MSc. Augusto Franco

Fecha de aprobación: Guatemala, _____ de _____ de 2024.

Prefacio

A las personas que me acompañaron en este lento proceso de aceptación y entendimiento.

A mis papás por su esfuerzo monumental para que sus hijos alcanzaran un grado académico universitario.

A Gaby por el amor, la complicidad y la paciencia en este proceso.

A todas y todos, gracias.

Índice

Lista de figuras

Lista de cuadros

Resumen

Este trabajo de graduación presenta el desarrollo de un *pipeline* bioinformático para el análisis de secuencias multi-locus (MLSA) con el fin de apoyar estudios filogenéticos. A diferencia de los enfoques de un solo gen, el MLSA integra datos de múltiples loci genéticos, incrementando la resolución y precisión en la inferencia de relaciones evolutivas, especialmente entre especies estrechamente relacionadas. El *pipeline* incorpora herramientas como *MUSCLE* y *Clustal Omega*para alineación de secuencias, AMAS y SeqKit para concatenación, y métodos de Máxima Verosimilitud, Vecino más cercano e Inferencia Bayesiana para la inferencia filogenética. Los resultados muestran un 80 % de coincidencia en un dendrograma generado por Máxima Verosimilitud y un 100 % de coincidencia con el dendrograma realizado con *MRBayes*, resaltando la confiabilidad del *pipeline* en diferentes métodos inferenciales.

Desarrollado en Python y Snakemake, el *pipeline* está diseñado para ser accesible y eficiente, abordando la necesidad de soluciones bioinformáticas fáciles de usar, especialmente para investigadores con recursos computacionales limitados. La validación con un artículo publicado de análisis MLSA, provee evidencia de que el proyecto facilita análisis filogenéticos robustos. Esta herramienta es especialmente valiosa en regiones con alta biodiversidad y recursos limitados, como Guatemala, ofreciendo una alternativa rentable y promoviendo capacidades locales en biología evolutiva y taxonomía.

Abstract

This thesis presents the development of a bioinformatics *pipeline* for multi-locus sequence analysis (MLSA) to support phylogenetic studies. Unlike single-gene approaches, MLSA integrates data from multiple genetic loci, increasing the resolution and accuracy of evolutionary relationship inference, especially among closely related species. The *pipeline* incorporates tools such as *MUSCLE* and *Clustal Omega* for sequence alignment, AMAS and SeqKit for concatenation, and Maximum Likelihood, Neighbor-Joining, and Bayesian Inference methods for phylogenetic inference. Results demonstrate an 80 % match in a dendrogram generated by Maximum Likelihood and a 100 % match with the dendrogram produced by *MRBayes*, underscoring the *pipeline*'s reliability across different inferential methods.

Developed in Python and Snakemake, the *pipeline* is designed to be accessible and efficient, addressing the need for user-friendly bioinformatics solutions, especially for researchers with limited computational resources. Validation against a published MLSA analysis article provides evidence that this project facilitates robust phylogenetic analysis. This tool is especially valuable in biodiversity-rich, resource-limited regions such as Guatemala, offering a cost-effective alternative and promoting local capabilities in evolutionary biology and taxonomy.

CAPÍTULO 1

Introducción

Este trabajo de graduación se enfocó en el desarrollo de un *pipeline* bioinformático para el análisis filogenético utilizando múltiples loci genéticos (MLSA), una técnica que permite inferir relaciones evolutivas con mayor precisión al combinar información de diferentes genes o regiones genómicas. La estructura del documento está diseñada para que el lector recorra tanto los aspectos teóricos como los técnicos y prácticos que fundamentan y aplican este *pipeline*, comprendiendo en cada sección los desafíos y decisiones que orientaron el desarrollo de la herramienta.

En la justificación se podrán encontrar cómo esta herramienta busca no solo mejorar el análisis filogenético, sino también ofrecer una solución adaptable a distintos entornos de investigación, con énfasis en su accesibilidad y eficiencia. La justificación establece las motivaciones que respaldan el diseño de un sistema que integra diferentes métodos de alineación y concatenación de secuencias, así como opciones de inferencia filogenética, ofreciendo así una mayor versatilidad para el usuario. Se presenta un contexto amplio sobre la importancia del análisis filogenético en el campo de la bioinformática y los avances tecnológicos que han permitido que estos estudios sean más detallados y precisos. También se destaca cómo la capacidad de personalización y flexibilidad de las herramientas bioinformáticas es fundamental en el trabajo de investigación.

El objetivo de esta investigación fue crear un *pipeline* filogenético multi-locus con la capacidad de seleccionar entre varios algoritmos de alineación y concatenación, así como entre diversos métodos de inferencia filogenética. Al igual que desarrollar una herramienta de visualización de los resultados del *pipeline* y comparar los resultados de la herramienta contra un estudio de análisis multi-locus bien documentado y con secuencias disponibles en internet.

En el marco teórico, se desarrollan los conceptos fundamentales para entender el análisis filogenético. Esta sección explora temas como la evolución y la filogenética molecular, junto con una explicación de los árboles filogenéticos y su interpretación. Además, se describen los criterios para la selección de loci y los métodos de alineación y concatenación, los cuales son esenciales para realizar un análisis multi-locus preciso y confiable.

En el marco metodológico, se detalla cada fase de implementación del *pipeline*, desde la carga y preparación de archivos hasta los procesos de alineación, concatenación y la inferencia filogenética final. Aquí, se explica la lógica detrás de la selección de herramientas como MUSCLE, Clustal Omega, AMAS y SeqKit, así como las opciones que el usuario puede ajustar en cada etapa para optimizar el análisis según sus necesidades específicas. El marco metodológico ofrece, así, una guía técnica exhaustiva sobre el funcionamiento de cada componente de la herramienta.

En la sección de resultados y discusión, se presentan los hallazgos derivados de la implementación y validación del *pipeline*, mostrando cómo se desempeña la herramienta en distintos escenarios de análisis. Esta sección incluye una comparación de los resultados obtenidos con diferentes métodos de inferencia filogenética, subrayando la importancia de la flexibilidad al permitir al usuario seleccionar entre herramientas y algoritmos. También se analizan los resultados en términos de precisión y eficiencia, destacando los beneficios de contar con un *pipeline* ajustable y adaptable a las condiciones de cada análisis.

Este trabajo de graduación ofrece al lector una guía completa sobre el diseño y la implementación de un *pipeline* bioinformático para el análisis filogenético basado en múltiples loci. Cada capítulo aborda un aspecto esencial del proyecto, desde la fundamentación teórica y los objetivos planteados hasta la metodología empleada y los resultados obtenidos. Este recorrido permite comprender los retos y decisiones involucradas en el desarrollo de una herramienta robusta y flexible, capaz de adaptarse a diferentes necesidades de investigación. Con esta estructura, el documento busca no solo describir el proceso de construcción del *pipeline*, sino también proporcionar una base sólida para quienes deseen profundizar en el análisis filogenético multi-locus y sus aplicaciones en el campo de la bioinformática.

CAPÍTULO 2

Justificación

El análisis filogenético es un método fundamental en biología que permite inferir las relaciones evolutivas entre diferentes especies o grupos de organismos. Utilizando datos morfológicos, bioquímicos o genéticos, este método reconstruye árboles filogenéticos basados en las similitudes y diferencias de las características de los organismos, que se presume tienen un ancestro común. Los análisis filogenéticos han ganado una mayor precisión y profundidad con los avances en tecnologías de secuenciación genética y métodos computacionales. Estos análisis se centran en la comparación entre marcadores moleculares individuales, como el ADN mitocondrial o ribosomal. Sin embargo, la literatura moderna sugiere que el uso de múltiples marcadores moleculares, también conocidos como análisis multi-locus, puede incrementar significativamente la resolución, permitiendo resolver relaciones más cercanas y complejas entre organismos que serían indetectables con un solo marcador (Dupuis *et al.*, 2012).

Este enfoque es especialmente valioso cuando se estudian taxones estrechamente relacionados, donde se han obtenido resultados positivos comparando genes mitocondriales, como citocromo b y citocromo c, oxidasa subunidad I (COI), en conjunto con genes del núcleo como el espaciador interno transcrito (ITS) y genes de ARN ribosomal (rRNA). Sin embargo, estas comparaciones no son exclusivas a estos genes y varían entre grupos de organismos por lo que una herramienta computacional flexible que permita especificar una cantidad de genes para el análisis, independientemente de cuales sean, sería de gran utilidad. Por ejemplo, el estudio "*Evaluating multi-locus phylogenies for species boundaries determination in the genus Diaporthe*" (Santos *et al.*, 2017), utilizó los genes ITS, CAL, HIS, TEF1 y TUB para discernir las relaciones taxonómicas entre especies del género *Diaporthe*. Mientras que el estudio "*Multilocus Sequence Analysis, a Rapid and Accurate Tool for Taxonomic Classification, Evolutionary Relationship Determination, and Population Biology Studies of the Genus Shewanella*"(Fang, Wang, Liu, Dai, Cai, Li *et al.*, 2019a), utilizó los genes gyrA, gyrB, infB, recN, rpoA, y topA para validar su herramienta MSLA con el género *Shewanella*. Estos ejemplos ilustran la aplicación práctica y la eficacia del análisis multi-locus en el discernimiento de relaciones filogenéticas complejas, destacando la importancia de adaptar los enfoques metodológicos a las características específicas de cada grupo taxonómico.

La capacidad de incorporar múltiples genes seleccionados de forma específica en los análisis no solo aumenta la precisión, sino que también refuerza la confiabilidad de las conclusiones filogenéticas. En los últimos años, los avances en tecnologías de secuenciación de siguiente generación (NGS) han hecho posible la secuenciación de un gran número de marcadores moleculares de forma simultánea (Foox *et al.*, 2021). Sin embargo, el análisis multi-locus presenta varios retos, incluyendo la necesidad de concatenar las secuencias de distintos marcadores y alinear estas secuencias concatenadas entre varios organismos. Además, gestionar la potencial discordancia entre las historias evolutivas de los distintos marcadores es un desafío significativo. Este último aspecto puede llevar a confusiones, como cuando secuencias no relacionadas son lo suficientemente similares como para causar falsos positivos en las alineaciones. Para afrontar estos retos, se han desarrollado diversas herramientas de software para el análisis genético, incluyendo programas para la alineación de secuencias (Caporaso *et al.*, 2010; Steenwyk *et al.*, 2020; Warris *et al.*, 2018), selección de marcadores (Bell, 2011; Maglott *et al.*, 2011a) e inferencia filogenética (Lewis *et al.*, 2015; Moreno *et al.*, 2022). No obstante, muchas de estas herramientas se ofrecen como paquetes de software individuales y el procedimiento para integrar una secuencia lógica entre ellas a menudo requiere la intervención de expertos en bioinformática. Por ello, existe la necesidad de una herramienta más flexible y amigable con el usuario que pueda manejar un amplio rango de datos multi-locus y permitir la elección de métodos de análisis en todos los pasos del proceso.

Una de las ventajas más grandes de este enfoque es la rentabilidad de realizar análisis multi-locus comparado con los costos de secuenciación de genoma completo. Mientras que la secuenciación de genoma completo puede proveer datos integrales sobre la composición genética de un organismo, muchas veces es demasiado cara. En contextos de bajos recursos como Guatemala, el acceso a secuenciación es limitado (Comunicación, 2023) en muchos casos requiere la tercerización de los procesos a laboratorios en el extranjero. Estos laboratorios tienen tarifas de cobro que incrementan con la longitud y el tiempo que toma realizarlos. El uso de varios marcadores moleculares puede proveer una forma más rentable de inferir relaciones evolutivas, ya que la secuenciación de múltiples fragmentos puede ser realizada en una sola ejecución del secuenciador.

Este proyecto desarrolla una herramienta de software capaz de analizar secuencias multi-locus para el análisis filogenético. El desarrollo de una herramienta de software para el análisis de secuencias multi-locus tiene el potencial de expandir el alcance e impacto del análisis filogenético, particularmente en campos como la ecología evolutiva, biología de conservación e investigación en biodiversidad en países de escasos recursos, pero ricos en diversidad de especies como Guatemala (CONAP, 2014).

CAPÍTULO 3

Objetivos

3.1. Objetivo general

- Desarrollar un *pipeline* para analizar datos multi-locus en análisis filogenético

3.2. Objetivos específicos

- Implementar una herramienta de reporte para la visualización interactiva y exploración de datos multi-locus.
- Permitir la alineación de secuencias multi locus y secuencias de referencia utilizando MUSCLE y Clustal Omega.
- Validar la herramienta utilizando secuencias de referencia disponibles en internet.

CAPÍTULO 4

Alcance

Este trabajo de graduación de licenciatura se limita al desarrollo de un pipeline funcional para los proceso de inferencia filogenética. Permite alinear, concatenar, inferir y visualizar los resultados de un estudio de filogenia. Para poder utilizar esta herramienta es necesario contar con la biblioteca de secuencias, con todos los procesos de verificación de calidad respectivos. Esta herramienta no verifica la calidad de las secuencias y asume que son de la mejor calidad posible.

4.1. Uso de algoritmos en el proceso

Los algoritmos seleccionados como opciones en las distintas fases del proceso provienen de programas informáticos diseñados por otros desarrolladores. Este trabajo no busca modificar o agregar capacidades adicionales a lo que las herramientas proveen. Esta herramienta busca diseñar un flujo optimizado entre las distintas herramientas, dandole al usuario la opción de parametrizar cada una de ellas con los parámetros específicos de cada herramienta.

CAPÍTULO 5

Marco teórico

5.1. Entendiendo filogenética y evolución

La evolución es el proceso mediante el cual todas las poblaciones alteran su material genético a través del tiempo, reflejando la adaptación de organismos a los cambios ambientales. Esta mutación genética ocurre de forma aleatoria en organismos individuales de una especie y dan lugar, a través del tiempo, a nuevos alelos. Estos alelos se transfieren de generación en generación a través de procesos de selección natural, en donde los organismos mejor adaptados para la supervivencia se reproducen exitosamente. Esto incrementa la presencia de alelos ventajosos en una población.

La evolución no es un proceso lineal, existen una infinidad de procesos, condiciones y estímulos, como la deriva y el flujo genético que pueden alterar la frecuencia de alelos de maneras inesperadas. Estos factores evolutivos y contextuales eventualmente pueden dar lugar a poblaciones lo suficientemente distintas entre sí para que estas puedan ser consideradas especies diferentes. Todas las especies comparten un ancestro común, que a su vez comparte un ancestro común con otras especies, por lo que es posible esclarecer las relaciones evolutivas de especies trazando los linajes hasta su origen. Es a través de esto que podemos decir con confianza que los humanos comparten ancestros comunes con otras especies de primates modernos y que existe un ancestro común entre plantas y animales (Baldauf, 2003; Strickberger, 2005).

Al estudio de estas relaciones evolutivas entre especies, organismos, o genes se le llama filogenética. Esta rama de la biología se basa en la comparación de ADN o secuencias de proteínas para estimar el pasado evolutivo (Baldauf, 2003; Emery, 2015). La idea de la filogenética y la representación de esta a través de árboles filogenéticos que relacionan a las especies entre sí, viene desde los tiempos de Charles Darwin. A estas primeras representaciones también se les conoce como cladogramas. Sin embargo, la filogenética molecular moderna se basa en métodos cuantitativos para estimar la distancia evolutiva entre organismos.

5.1.1. Árboles filogenéticos

Los árboles filogenéticos, o filogenias son una representación visual y cuantitativa de la relación genealógica entre organismos. Estos están compuestos de ramas (bordes) y nodos. Las ramas conectan nodos entre sí y un nodo es un punto en el que dos o más nodos se conectan. Estas ramas y nodos pueden ser externos o internos. Los nodos internos corresponden al último ancestro en común (LCA por sus siglas en inglés) y los nodos externos corresponden a los organismos del estudio, a estas también se les denomina unidades taxonómicas operacionales (OTUs por sus siglas en inglés). Los árboles filogenéticos se pueden representar de muchas formas, intuitivamente se representan de abajo hacia arriba como un árbol real. En donde el nodo raíz se encuentra abajo y los distintos nodos se sitúan hacia arriba. Sin embargo, esta representación, a pesar de ser intuitiva con la realidad, se torna compleja de representar cuando agregamos complejidad al árbol. Por lo que se recomienda trazarlas de izquierda a derecha, en donde el nodo raíz se encuentra del lado izquierdo (Yang y Rannala, 2012).

Los árboles filogenéticos moleculares, a diferencia de un cladograma, se dibujan usualmente, con ramas de tamaños proporcionales. Esto significa que los largos de las ramas representan de alguna forma la cantidad de evolución que ocurre entre nodos. Por tanto, mientras más larga la rama, más relativamente divergente son los organismos que se conectan a ellas (Baldauf, 2003; Yang y Rannala, 2012). La Figura 1 muestra una representación visual de un árbol filogenético dibujado de abajo hacia arriba con sus partes señalizadas.

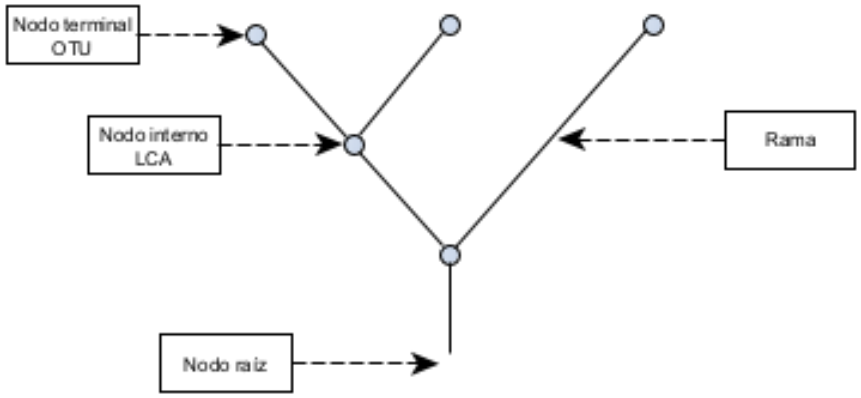

Figura 1: Árbol filogenético con partes señalizadas (Autoría propia)

En la actualidad, las filogenias se utilizan en casi todas las ramas de la biología. Además de representar la relación entre especies, se utilizan para describir las relaciones entre genes parálogos en una familia de genes, dinámicas evolutivas y epidemiológicas de patógenos, las relaciones genealógicas entre células somáticas durante la diferenciación y el desarrollo de cáncer, y en la clasificación metagenómica de secuencias, como las utilizadas en transcriptómica (O'Halloran, 2014; Yang y Rannala, 2012). Durante la pandemia de COVID-19 la filogenética fue de suma importancia para identificar variantes del virus SARS-COV-2 a través de la secuenciación y análisis de las proteínas spike y el código genético del virus (Attwood *et al.*, 2022). Las metodologías de cálculo de árboles filogenéticos son muchas y han dado lugar a ramas de estudio por si solas, como la filogenética estadística que se en-

carga de la creación y adaptación de métodos estadísticos, matemáticos y computacionales para resolver relaciones de distancia entre nodos, a través de metodologías basadas en la distancia o en los caracteres genéticos.

En las metodologías basadas en la distancia, se calcula la distancia entre cada par de secuencias del estudio y se imputa en una matriz. La matriz de distancia resultante se utiliza para la reconstrucción del árbol filogenético. Por ejemplo, el método de unión de vecinos utiliza un algoritmo de agrupamiento (conocido como *clustering*) a la matriz de distancias para determinar una filogenia completa.

Por su parte, los métodos basados en caracteres incluyen métodos de parsimonia máxima, máxima probabilidad de ocurrencia y métodos de inferencia Bayesiana. Estos acercamientos comparan de forma simultánea todas las secuencias en la alineación considerando un solo carácter de la secuencia y calcula una ponderación para cada árbol. Esta ponderación se refiere al número mínimo de cambios para el método de máxima parsimonia, al valor de probabilidad logarítmica para el método de máxima probabilidad de ocurrencia y a la probabilidad posterior para la inferencia Bayesiana. Para los tres métodos, el árbol con la mayor ponderación se puede obtener, en teoría, comparando todos los árboles posibles. En la práctica, debido a la gran cantidad de árboles posibles, esta clase de estudios no son computacionalmente viables, a menos que estemos trabajando con pocos datos por lo que se utilizan algoritmos de búsqueda heurística. Estos acercamientos utilizan un algoritmo rápido para generar un árbol inicial, luego computan reordenamientos locales para intentar incrementar la ponderación del árbol.

Para poder aplicar metodologías de generación de árboles filogenéticos moleculares necesitamos entender de dónde vienen los datos que se analizan. En la actualidad estos provienen de métodos de secuenciación de siguiente generación (conocidos también como NGS por sus siglas en inglés o métodos de secuenciación de alto rendimiento)(Yang y Rannala, 2012).

5.1.2. Secuenciación de alto rendimiento

En la última década ha habido un incremento dramático en los métodos disponibles para la recolección de datos genómicos. Comenzando con la pyrosecuenciación y las primeras máquinas de secuenciación de siguiente generación a inicios de los 2000, los investigadores han visto el costo por nucleótido disminuir por varios órdenes de magnitud. Para el estudio de la filogenética, esto ha significado un boom de posibilidades. Estos procesos de secuenciación son complejos y la ciencia detrás de las cada vez más portátiles máquinas secuenciadoras de nucleótidos varía de proveedor a proveedor. Estas soluciones comerciales incrementan el rendimiento de secuenciación a través de métodos diferentes. En la actualidad las tecnologías de secuenciación pueden dividirse en dos ramas principales: Los secuenciadores basados en plataformas de amplificación clonal y los basados en plataformas de secuenciación de molécula única (Lemmon y Lemmon, 2013; Reuter *et al.*, 2015).

Los basados en amplificación clonal (incluyendo a los secuenciadores de Illumina) cuentan con un proceso de amplificación en donde se incorporan en un medio, ya sea una celda fluida de vidrio copias de patrones de ADN, los cuales se pintan con tintes fluorescentes, se capturan por medio de imágenes y se cortan para remover el tinte y regenera la superficie para el siguiente ciclo. Un análisis de las imágenes en 4 colores sirve para identificar las bases.

Otras plataformas como Ion Torrent utilizan un semiconductor para determinar, a través de cambios de pH los cambios que ocurren por la liberación de ADN a través de procesos de extensión. Estos cambios se detectan por medio de sensores que se convierten a voltaje para identificar proporcionalmente el número de nucleótidos agregados por la enzima polimerasa (Reuter *et al.*, 2015).

Por otra parte, los métodos basados en secuenciación de molécula única como Pacific Bioscience y Oxford Nanopore, colocan una polimerasa única en un punto en el que los nucleótidos tengan que pasar por algún tipo de sensor, ya sea una cámara o un sensor de cambios de corriente eléctrica para obtener nucleótido por nucleótido secuencias completas (Reuter *et al.*, 2015).

Los secuenciadores como las plataformas de la marca Illumina realizan procesos de agrupación que permiten que varias secuencias se obtengan en paralelo e incluso que se inicien las reacciones en distintos puntos de una cadena de nucleótidos al mismo tiempo. Generando volúmenes de datos mayores en menor tiempo y con un porcentaje de error muy bajo. Sin embargo, el resultado de estos procesos de secuenciación es usualmente el mismo. Una secuencia de nucleótidos en formato de archivo de texto que representa los resultados. Es a través de estas secuencias resultado que la filogenética molecular se ha vuelto un método viable de responder a muchas preguntas biológicas. Sin embargo, también da lugar a nuevas fuentes de error y nuevas preguntas (Lemmon y Lemmon, 2013).

Errores derivados de la secuenciación en filogenética

Con el incremento del volumen de datos derivados de la secuenciación de alto rendimiento. También se incrementan los errores que se introducen a los análisis. Esto se deriva del error primario de la plataforma de secuenciación, pero también se deriva de la complejidad de los datos disponibles.

Los dos tipos de error filogenético principales son el error estocástico y el error sistemático. El error estocástico, o de falta de precisión, ocurre porque no se tiene suficiente información filogenética y se reduce si la cantidad de datos incrementa. El error sistemático, o de falta de exactitud, ocurre principalmente por la especificación de un modelo incorrecto y se incrementa si la cantidad de datos se incrementa, ya que el modelo se sesga. Esto significa que la probabilidad de obtener un estimado filogenético erróneo pero que tenga evidencia se incrementa si no se maneja el error sistemático (Lemmon y Lemmon, 2013).

Cuando se estima un árbol filogenético utilizando una sola región genómica, incrementar el número de sitios de análisis reduce el error estocástico. Sin embargo, es posible alcanzar un punto en donde incrementar el número de sitios comienza a introducir error sistemático si hay procesos importantes sin modelar. Esto parece indicar que la magnitud de los errores es altamente dependiente en las propiedades de los datos que se analizan, por lo que la forma más efectiva de incrementar la precisión filogenética es a través del submuestreo de los datos (Lemmon y Lemmon, 2013). Esta técnica estadística consiste en tomar una muestra más pequeña de una muestra más grande con el fin de reducir la cantidad de datos a analizar sin perder la representatividad de los datos originales.

El principal propósito del submuestreo es incrementar la proporción de información filo-

genética versus error sistemático. La clave es encontrar el subconjunto de datos con suficiente información para resolver la historia filogenética mientras se minimiza el error sistemático. El submuestreo puede ocurrir en varios puntos de un análisis filogenético, como en el proceso de selección de secuencias ortólogas y en la selección de sitios específicos dentro de la alineación de secuencias (Rannala y Yang, 2008; Rodréguez-Ezpeleta *et al.*, 2007).

5.2. Loci y sitios de análisis

En el contexto del análisis filogenético, un locus o loci en plural es una posición fija en un cromosoma en donde encontramos un gen o marcador genético particular. Un sitio de análisis se refiere a un locus especifico elegido por su variabilidad e importancia evolutiva para determinar las relaciones filogenéticas entre organismos. Estos sitios se seleccionan con base en su habilidad para proveer señales claras de divergencia evolutiva y típicamente son partes de genes o genomas que evolucionan a una tasa apropiada para distinguir las relaciones entre los taxones de interés (Hakeem *et al.*, 2019).

5.2.1. Análisis de secuencias multi-locus (MLSA)

La filogenética multi-locus ha emergido como una metodología poderosa para resolver las relaciones evolutivas complejas, especialmente entre taxones estrechamente relacionados. A diferencia del análisis de un solo marcador, MLSA utiliza múltiples loci genéticos, aumentando la resolución y confiabilidad de los árboles filogenéticos.

La selección de sitios de análisis apropiados es crucial para la precisión de un análisis de inferencia filogenética. Se espera que cada uno de los loci elegidos para un análisis MLSA sirva para esclarecer una señal evolutiva única, y que la combinación de los datos de varios loci ayude a crear un panorama lo suficientemente claro para esclarecer las relaciones evolutivas entre los organismos (Baldauf, 2003; Rodréguez-Ezpeleta *et al.*, 2007; Roy, 2014).

5.2.2. Criterios de selección de sitios de análisis

La selección de loci es crítica para el éxito de MLSA. Los loci seleccionados deben evolucionar a tasas adecuadas para los niveles taxonómicos de interés. Por ejemplo, genes altamente conservados pueden ser útiles para resolver relaciones profundas entre grupos de organismos, mientras que loci más variables pueden ser necesarios para distinguir entre especies estrechamente relacionadas. Para seleccionar sitios de análisis para un estudio filogenético se debe tener claridad de los siguientes criterios de selección:

- Conservación contra variabilidad: El sitio debe tener suficientes regiones conservadas para la alineación adecuada entre especies, pero suficiente variabilidad para distinguir entre ellas.
- Tasa evolutiva: El sitio debe evolucionar a una tasa apropiada para la precisión filogenética de interés. Esto quiere decir que se deben elegir sitios de evolución rápida

para especies relacionadas cercanamente y, sitios de evolución lenta para relaciones taxonómicas más lejanas.

- Restricciones funcionales: Para análisis filogenético se prefieren sitios que provean señales evolutivas consistentes. Por ejemplo, se utilizan genes de ARN ribosomal por su presencia en toda la vida celular y por su mezcla entre regiones conservadas y regiones variables.

- Viabilidad técnica: Se deben seleccionar sitios que se puedan amplificar y secuenciar correctamente, considerando factores como compatibilidad de PCR y calidad de la secuencia. Este aspecto, de forma indirecta también garantiza la presencia de secuencias de referencia en bases de datos. Si se eligen sitios difíciles de secuenciar, se encontrarán menos referencias en línea.

- Paralogía y duplicación genética: Se deben evitar regiones con duplicaciones de genes o parálogos, estos pueden confundir a los modelos de inferencia filogenética debido a la presencia de múltiples copias.

- Sitios de transferencia horizontal de genes (HGT por sus siglas en inglés): Se deben seleccionar sitios que tengan menor probabilidad de ser afectados por procesos de transferencia horizontal, ya que estos ofuscan relaciones evolutivas reales.

- Contexto genómico: Se debe considerar el contexto genómico de los loci de análisis para garantizar que sean representativos de la filogenia del organismo.

- Calidad de la secuencia: Es importante asegurar alta calidad de la secuencia que se utiliza para minimizar errores que puedan dar lugar a falsas divergencias.

- Reproducibilidad: Se deben elegir sitios que provean resultados consistentes en diferentes metodologías y estudios.

Cuando hablamos de MLSA, debemos considerar que estos criterios deben cumplirse para cada uno de los loci del análisis. Por tanto, un análisis multi-locus debe ser respaldado extensamente por literatura científica para cada uno de los loci para evitar incrementar el error sistemático (Rodréguez-Ezpeleta *et al.*, 2007).

5.3. Computación de análisis multi-locus

Para desarrollar un MLSA necesitamos recolectar información de referencia que pueda ser de utilidad para esclarecer la relación del organismo de interés. Estas secuencias de referencia dependerán de los sitios de interés seleccionados para el estudio, los resultados y calidad de la secuenciación de los resultados del organismo de interés y un toque de conocimiento empírico sobre el organismo que estamos estudiando. Esto quiere decir que no vamos a buscar secuencias de referencia de organismos con taxonomía demasiado divergente. Para esto, utilizaremos bases de datos de referencia de acceso público.

5.3.1. Bases de datos genómicas

Las bases de datos bioinformáticas son el principal recurso para la recolección de la información genómica que alimenta el avance científico bioquímico. Estas bases de datos, en su mayoría, son de acceso público y existen con el propósito de difundir datos y conocimiento en formatos estandarizados por la comunidad científica, producidos por investigaciones realizadas por equipos académicos universitarios, centros de salud internacionales, casas médicas y otras iniciativas privadas como fundaciones y empresas que deciden colaborar.

Estas bases de datos continúan creciendo diariamente y en la actualidad son recursos indispensables para la lucha contra la pandemia del coronavirus, dando acceso a genomas completos, secuencias de proteínas de membrana, variantes genéticas y recopilando de forma sistemática avances científicos e investigaciones relacionadas (Agarwala *et al.*, 2018).

Las bases de datos genómicas y transcriptómicas permiten a investigadores en distintas ramas de las ciencias de la vida, aprovechar el conocimiento colectivo e impulsar investigaciones propias o incluso descubrimientos que han cambiado la forma en la que entendemos nuestros cuerpos, los organismos que nos rodean, la forma en la que interactuamos con agentes patogénicos y con las diversas maquinarias internas que nos dan vida.

Estas bases de datos son extremadamente amplias ya que son el producto de esfuerzos colectivos internacionales. Para propósitos de esta investigación nos enfocaremos en las bases de datos producto de estos esfuerzos colectivos, especialmente las establecidas por consorcios internacionales de colaboración científica especialmente enfocadas en las correspondientes a genómica humana y transcriptómica humana más conocidas y actualizadas (Agarwala *et al.*, 2018).

Estas bases de datos han alcanzado el reconocimiento que tienen debido a su accesibilidad para usuarios de la comunidad científica con poco conocimiento computacional a través de extensas bibliotecas de herramientas computacionales interconectadas, gratuitas y en muchas ocasiones disponibles como servicios web que aprovechan recursos computacionales de centros de investigación remotos a través de computación en la nube (Johnson *et al.*, 2008; Karsch-Mizrachi *et al.*, 2018; Maglott *et al.*, 2011b).

Sin embargo, estas bases de datos cuentan con limitaciones técnicas para el usuario promedio, requieren conocimiento de términos técnicos avanzados y usualmente están acompañadas de interfases de usuario poco intuitivas y herramientas de ayuda inexistentes. Los investigadores con conocimiento computacional más avanzado pueden aprovechar de estas herramientas a través de accesos por interfases de programación de aplicaciones (API por sus siglas en inglés). Estas herramientas permiten descargar grandes volúmenes de datos de forma sistemática y ordenada para integrarlos de forma directa al desarrollo de tecnologías nuevas, herramientas de análisis y motores de búsqueda.

Consorcios científicos

Las bases de datos bioinformáticas se han enfrentado a varios problemas desde su concepción en los años 90. Los equipos que las mantienen y desarrollan las herramientas de acceso a las mismas encontraron que la sistematización, unificación y tamizaje de nuevo

conocimiento, era un proceso extremadamente complejo. Para tener una base de datos centralizada, actualizada, utilizable y relevante, es necesario recopilar los esfuerzos de todos los equipos científicos trabajando en distintas ramas de las ciencias de la vida y ninguna institución tenía, ni tiene, la capacidad para abarcar todo el planeta. Esto dio origen a consorcios científicos. Agrupaciones de organizaciones, usualmente sin fines de lucro, que, a través de estándares, protocolos unificados y colaboración abierta, han comunicado bases de datos con recursos en idiomas distintos, formatos distintos y requisitos de información distintos.

Estas bases de datos se alimentan constantemente entre sí, usualmente de forma diaria y comparten toda la información estableciendo un formato único internacional accesible desde cualquiera de los sistemas afiliados.

El principal consorcio es la **base de datos de colaboración internacional de secuencias de nucleótidos (INDSC)**. Este proyecto inició hace más de 30 años y tiene como meta la captura, preservación y el acceso abierto a secuencias de nucleótidos y sus metadatos asociados. Este consorcio científico está compuesto por la base de datos de ADN de japón (DDBJ) del instituto nacional de genética de Mishima, Japón, el archivo europeo de nucleótidos (ENA) del instituto de bioinformática europeo del laboratorio europeo de biología molecular (EMBL-EBI) en Hinxton, Reino unido y GenBank en el centro nacional para la información en biotecnología (NCBI), de la biblioteca nacional de medicina y el instituto nacional de salud en Bethesda, Maryland, Estados Unidos (Karsch-Mizrachi *et al.*, 2018).

Los laboratorios miembros de INDSC trabajan en conjunto para que la información de secuencias de nucleótidos que se deposita en alguno de los tres sitios se preserve como parte del historial científico de la humanidad y que esté disponible en formatos estandarizados a través de cualquiera de los sitios. Los archivos de INDSC colaboran para responder ante el surgimiento de nuevas tecnologías de secuenciación para poder adaptar los sitios y características de los formatos para mantenerse al día con los cambios en la ciencia. El alcance de los datos disponibles en los sitios miembros incluye secuencias crudas, alineamientos de secuencias en formato SRA y secuencias con anotaciones funcionales con referencia al estudio que las obtuvo y datos sobre los resultados encontrados.

La estructura de metadatos estandarizada sirve para describir la muestra biológica incluyendo información taxonómica, diseño experimental, alcance del proyecto, comentarios de los investigadores y referencias bibliográficas. Esta estructura se agrega a cada una de las secuencias disponibles para proveer contexto y para permitir a investigadores utilizar las herramientas de búsqueda de los sitios para encontrar información útil.

Cada uno de los centros provee herramientas para la comunidad que las utiliza para poder agregar nuevas secuencias de nucleótidos. Estas herramientas están en constante mejoría y han iterado desde herramientas en consola de Linux hasta páginas web guiadas para que personas con poco conocimiento computacional puedan contribuir de forma correcta. Asegurando que los requisitos mínimos se cumplan y que los datos sean sintáctica y semánticamente válidos(Agarwala *et al.*, 2018; Karsch-Mizrachi *et al.*, 2018). La decisión de trabajar con bases de datos provenientes de INDSC surge principalmente por las herramientas comunitarias diseñadas por cada uno de los sitios para acceder a los datos. Cada uno de los sitios tiene sus propias herramientas, sus propios públicos objetivo, sin embargo, la información presente en alguno de ellos es la misma que se encuentra en los otros. Esto permite acceder

al triple de información utilizando solo una interfaz de programación.

5.3.2. Secuencias en formatos computables (archivos Fasta y FastQ)

En el análisis filogenético multi-locus (MLSA), los formatos de archivo FASTA y FASTQ juegan un papel crucial en la gestión y análisis de datos de secuenciación. Estos formatos son fundamentales para almacenar y manipular secuencias de ADN, permitiendo a los investigadores realizar análisis precisos y eficientes. **Formato FASTA** El formato FASTA es uno de los más utilizados en bioinformática para almacenar secuencias de nucleótidos o aminoácidos. Cada secuencia en un archivo FASTA se representa con una línea de encabezado que comienza con el carácter `">"` seguida de una breve descripción o identificador de la secuencia, y luego la secuencia de nucleótidos o aminoácidos en sí (Pearson y Lipmant, 1988).

- Encabezado: La línea que comienza con `">"` contiene un identificador único y opcionalmente una descripción de la secuencia.
- Secuencia: Las líneas subsecuentes contienen la secuencia de ADN o proteínas.

El formato FASTA es simple y fácil de manejar, lo que lo convierte en una opción preferida para almacenar y compartir secuencias biológicas. En el contexto de MLSA, los archivos FASTA se utilizan para almacenar secuencias concatenadas de múltiples loci, lo que facilita su alineación y análisis filogenético. **Formato FASTQ** El formato FASTQ extiende la funcionalidad del formato FASTA al incluir información sobre la calidad de las secuencias, lo cual es esencial para los datos generados por la secuenciación de nueva generación (NGS). Un archivo FASTQ contiene cuatro líneas por cada secuencia.

- Línea de encabezado: Comienza con `"@"` seguido de un identificador único para la secuencia.
- Línea de secuencia: Contiene la secuencia de nucleótidos.
- Separador: Una línea con un solo "+", esta puede tener caracteres adicionales, pero se ignoran.
- Línea de calidad: Contiene los valores de calidad de cada nucleótido en la secuencia, representados como caracteres ASCII.

Los archivos FASTQ son particularmente útiles para evaluar la precisión y confiabilidad de las secuencias obtenidas mediante NGS (Cock *et al.*, 2009). En MLSA, los archivos FASTQ se utilizan en las primeras etapas del análisis para verificar la calidad de las secuencias antes de su alineación y procesamiento.

5.4. Alineación de secuencias

La alineación de secuencias es el proceso mediante el cual se organizan secuencias de ADN, ARN o proteínas para identificar regiones de similitud que puedan ser consecuencia

de una relación funcional, estructural o evolutiva entre secuencias. La meta primaria de los algoritmos de alineación es encontrar la mejor forma de emparejar caracteres en dos o más secuencias para maximizar la similitud entre ellos. Existen distintos tipos de algoritmo para alinear secuencias:

- Alineación por pares: Alinea secuencias de dos en dos, esto implica que la comparación puede hacerse para dos secuencias o sistemáticamente se puede aplicar para más de dos, sin embargo, los errores incurridos en los pasos iniciales se propagan entre alineaciones ya que la comparación por pares no reajusta cuando encuentra una mejor alineación.
- Alineación de secuencias múltiples (MSA por sus siglas en inglés): Alinea secuencias en grupos de tres o más para determinar regiones de conservación entre ellas. MSA es más computacionalmente intenso, ya que realiza comparaciones en grupos de mayor tamaño y la mayoría de los algoritmos tienen proceso de reajuste cuando encuentran alineaciones de mayor similitud.

5.4.1. Algoritmos de alineación de secuencias múltiples (MSA)

Esta investigación se enfoca en los algoritmos de alineación de secuencias múltiples. Estos funcionan mejor para los MLSA ya que devuelven resultados más precisos en las alineaciones individuales de secuencias pequeñas que se requieren. En la actualidad los algoritmos más utilizados son:

MUSCLE

El algoritmo de secuenciación MUSCLE (Multiple Sequence Comparison by Log Expectation). Es un algoritmo MSA que opera en tres etapas: Inicialización, alineación progresiva y refinamiento. Cada uno de los pasos está diseñado para mejorar la precisión y eficiencia del proceso de alineación de secuencias.

En el paso de inicialización el algoritmo MUSCLE inicia contando k-meros (subsecuencias de longitud K) en cada secuencia y computa distancias pareadas entre las secuencias. Esto significa que la distancia se calcula basándose en el número de k-meros que comparten entre sí. Esto se utiliza para realizar un árbol de guía de las relaciones iniciales entre las secuencias que se espera alinear y sirve como plantilla para el proceso de alineación progresiva.

En el paso de alineción progresiva el algoritmo MUSCLE emplea un algoritmo de clustering para determinar el orden en el que las secuencias se alinean entre sí. Eso quiere decir que las secuencias más similares entre sí se colocan en el mismo cluster. Estas secuencias se alinean en el orden de prioridad del árbol resultante del proceso de clustering y en cada paso se utiliza un algoritmo de alineación por pares (usualmente el algoritmo perfil-perfil). El cual calcula una ponderación de alineación utilizando una matriz de punteo y agrega espacios vacíos progresivamente para mejorar este punteo.

Luego de completar este proceso de alineación progresiva, ocurre un proceso de refinación iterativa en donde se recalculan las distancias por pares de la alineación resultante y se vuelve a hacer un cluster de priorización y se inicia el proceso nuevamente. Este proceso se repite

hasta que el algoritmo no puede encontrar más optimizaciones del puntaje que pueda realizar o hasta que se llegue a un número máximo de iteraciones (Edgar, 2022).

Ventajas

- Alta precisión: El algoritmo MUSCLE es conocido en la comunidad científica por su precisión. El proceso de refinamiento iterativo ayuda a mejorar la alineación continuamente resultando en alineaciones precisas y confiables. Es un buen candidato para la construcción de *pipelines* de análisis como anotación funcional y construcción de árboles filogenéticos (Ranwez y Chantret, 2020).
- Velocidad y eficiencia: El algoritmo MUSCLE está diseñado para ser computacionalmente eficiente. La decisión de utilizar conteo de k-meros y alineación perfil-perfil lo vuelve capaz de alinear sets de datos voluminosos en poco tiempo (Ranwez y Chantret, 2020).
- Versatilidad: MUSCLE puede ser utilizado para alinear tanto secuencias de nucleótidos como secuencias de proteínas. Tiene soporte para varios tipos de archivos de entrada y salida haciéndolo una herramienta que se puede utilizar como parte de *pipelines* o flujos de trabajo diversos (Ranwez y Chantret, 2020).

Desventajas

- Uso de memoria: MUSCLE es eficiente en términos de velocidad, pero puede ser ineficiente en uso de memoria, especialmente cuando se trata de conjuntos de datos muy grandes (Ranwez y Chantret, 2020).
- Menos eficiente para secuencias divergentes: MUSCLE tiene problemas para alinear secuencias altamente divergentes o que tienen muchas inserciones o eliminaciones. En estos casos el árbol guía inicial puede no ser suficiente para capturar relaciones evolutivas reales (Ranwez y Chantret, 2020).

Clustal Omega

Clustal Omega (Cluster alignment workbench) es un programa utilizando ampliamente para la alineación MSA. Utiliza un método progresivo de alineación en donde las secuencias se alinean secuencialmente basándose en un árbol guía. En el primer paso se crea una alineación por pares utilizando el algoritmo de Needleman-Wunsch (Sievers y Higgins, 2018). Estas alineaciones generan una matriz de distancia que representa la disimilitud entre cada par de secuencias. Con base en la matriz de distancia se aplican penalizaciones por la presencia de vacíos para considerar inserciones y eliminaciones.

En el segundo paso se utiliza la matriz de distancia para construir el árbol guía que es una primera representación de relaciones evolutivas entre secuencias. Es importante notar que al igual que MUSCLE esta relación es preliminar y no es multi-locus. Para cada uno de los loci a utilizar se debe realizar un proceso de alineación. Para la construcción de este árbol, se utilizan o el algoritmo UPGMA (Unweighted Pair Group Method with Arithmetic Mean) o el algoritmo Neighbor-Joining. Estos se discutirán más adelante en este protocolo

ya que son los mismos algoritmos que se utilizan para la creación de un árbol filogenético multi-locus (Sievers y Higgins, 2014).

Con base a este árbol preliminar se realiza un proceso de alineación progresiva de acuerdo con el orden de los nodos y ramas del árbol filogenético (Walker, 2021).

Ventajas

- Amplia aceptación: Los métodos implementados por Clustal Omega han sido ampliamente validados y existen numerosos estudios que lo utilizan. Haciéndolo confiable para investigadores de todos los niveles (Sievers y Higgins, 2018).
- Altamente personalizable: Clustal Omega permite configurar los parámetros de forma amplia para mejorar los resultados de los procesos de alineación a pesar de su simplicidad de uso, tiene la capacidad de realizar análisis muy precisos a las necesidades del investigador.

Desventajas

- Computacionalmente intensivo: Los pasos de alineación por pares y construcción del árbol guía pueden requerir de mucho poder computacional. Especialmente para conjuntos de datos complejos. Muchos investigadores optan por realizar análisis de Clustal Omega en clusters de computadores privados o públicos como el disponible por EMBL (Madeira *et al.*, 2024).
- Falta de refinamiento iterativo: A diferencia de MUSCLE, no incluye procesos de refinamiento iterativo. Esto quiere decir que una vez se completa la alineación ya no se optimiza más. Lo que significa que puede obtener peores punteos de alineación en datasets cuyas diferencias iniciales son muy amplias y se propagan a través del proceso de alineación progresiva. La falta de reajuste implica que estos errores nunca se resuelven.

5.5. Concatenación de secuencias

Por la naturaleza de los MLSA, las regiones que se alinean son de menor tamaño y el costo computacional se divide en varias alineaciones. Esto quiere decir que, en lugar de alinear genomas completos con millones de nucleótidos, se alinean pequeños bloques entre sí y luego se realiza un proceso de concatenación entre ellas para crear una sola secuencia por organismo. Para realizar un proceso de concatenación debemos anteriormente haber realizado la alineación de todos los loci que vamos a utilizar en el estudio filogenético. Esto quiere decir que, para cada una de las regiones de interés, tenemos un archivo que contiene todas secuencias de distintos organismos correspondientes a esa región. Es importante que este proceso de concatenación considere los siguientes puntos:

- Debe mantenerse el orden de los loci en la secuencia compuesta para todos los organismos. Esto quiere decir que luego de concatenar podamos tener un archivo con

la alineación de secuencias de varios loci de todos los organismos que usaremos para construir el árbol filogenético.

- Se debe poder manejar la inserción de vacíos para mantener la consistencia entre las concatenaciones, pero también para considerar secuencias faltantes en alguno de los organismos. Esto quiere decir que deben poder insertarse "x" cantidad de vacíos cuando no tenemos la secuencia para alguna región de interés de alguno de los organismos sin alterar la alineación compuesta entre todos los organismos. Esto es de suma importancia para no alterar los resultados de la alineación agregando regiones confusas de "diferencia" entre secuencias.

- Al igual que en todo proceso computacional, se debe asegurar que las alineaciones que se concatenan sean de buena calidad. Si las alineaciones son pobres el resultado de la concatenación va a ser pobre.

5.5.1. Algoritmos de concatenación de secuencias

Existen varias herramientas y algoritmos de concatenación. Algunos de ellos han sido implementados como software independiente y otros como librerías de Python. Las siguientes son dos de ellas que se implementan en este *pipeline*.

AMAS

El paquete *Alignment Manipulation and Summary* (AMAS por sus siglas en inglés es una herramienta versátil y eficiente praa el procesamiento y manipulación de alineaciones de secuencias múltiples en distintos formatos incluyendo FASTA. Su fortaleza principal está en la capacidad para concatenar secuencais alineadas de distintos loci, creando una matriz generalizada que puede usarse para análisis filogenéticos (Borowiec, 2016).

Ventajas

- Eficiencia: Amas es conocido por su velocidad para manejar conjuntos de datos de gran tamaño, permitiendo la concatenaciones de múltiples alineanciones sin gastar tantos recursos computacionales. Esto es util cuando se trabaja con genomas completos.

- Facilidad de uso: Al ser un paquete de python la instalación es muy sencilla, al igual el paquete provee de herramientas para trabajar con formatos de entrada diversos sin la necesidad de realizar conversiones previas. Esto reduce el tiempo requerido para poder iniciar a usar la herramienta.

- Resumen de alineación: Una de las principales ventajas del uso de AMAS es que produce estadísticas de resumen del proceso de concatenación. Esto permite analizar rápidamente los resultados de la concatenación para decidir si se puede trabajar con el resultado o si debe hacerse nuevamente mejorando los datos.

- Manejo de datos faltantes: Uno de los principales retos del análisis de MLSA es la completitud de todas las secuencias de todos los organismos con los que trabajamos.

Completar las librerias de datos puede ser muy tardado y en ocasiones puede ser imposible, especialmente si se está trabajando con organismos o familias poco estudiadas que no tienen muchas secuencias de referencia en sitios como Genbank. AMAS permite realizar el proceso de concatenación incluso con datos faltantes.

Desventajas

- Manejo limitado de secuencias divergentes: AMAS No realiza procesos de refinamiento de alineaciones. Esto es necesario cuando las secuencias son de organismos altamente divergentes entre sí. Si esto no se identifica a tiempo puede resultar en matrices de concatenación que deshacen el trabajo realizado en el paso de alineación de secuencias (Kück y Longo, 2014).
- Falta de visualización de resultados: AMAS está diseñado para ser un paso intermedio en un proceso más complejo por lo que no provee ninguna herramienta para la visualización del resultado de la concatenación.

SeqKit

SeqKit es un conjunto de herramientas diseñadas para la manipulación eficiente de archivos FASTA/FASTQ. Es uno de los paquetes esenciales de *pipelines* bioinformáticos. Provee herramientas par la extracción, filtro, concatenación y resumen de secuencias (Shen *et al.*, 2016).

Ventajas

- Velocidad y eficiencia: SeqKit está optimizado para la velocidad de concatenación de secuencias de conjuntos de datos grandes.
- Facilidad de uso: SeqKit tiene comandos fáciles de utilizar para la concatenación. El usuario simplemente provee los archivos de entrada para cada locus y SeqKit los combina en uno solo, asegurando el orden correcto y agregando espacios vacíos en donde sea necesario.
- Consistencia locus a locus: SeqKit asegura que las secuencias concatenadas mantenan el mismo orden en todos los loci utilizados. Esto es esencial para generar alineaciones coherentes de varios organismos un paso fundamental para MLSA.

Desventajas

- Personalización limitada para espacios vacíos: SeqKit no tiene herramientas para mejorar la inserción de espacios vacíos. No tiene estrategias especiales de inserción. Trabaja con la información caso a caso y no realiza ningún algoritmo complejo de inserción de espacios por lo que los resultados pueden no ser los necesarios para un análisis complejo.

- Refinamiento de alineaciones: SeqKit no revisa ni mejora las alineaciones antes de concatenar. Asume que los archivos de entrada corresponden a alineaciones de buena calidad.
- Herramienta de línea de comandos: SeqKit es una herramienta de línea de comandos por lo que puede ser una limitante de uso para usuarios no familiarizados con trabajar en terminales.

5.6. Inferencia filogenética

La inferencia filogenética se refiere al proceso de construcción de árboles filogenéticos o redes que representan las relaciones evolutivas entre un conjunto de organismos o genes. Estos árboles permiten ilustrar las conexiones genealógicas entre especies, poblaciones o incluso genes dentro de un mismo genoma, posibilitando el rastreo de sus orígenes evolutivos. En el contexto del Análisis de Secuencias Multilocus (MLSA, por sus siglas en inglés), la inferencia filogenética desempeña un papel fundamental en la resolución de relaciones entre especies o cepas estrechamente relacionadas. Al combinar información de múltiples loci genéticos, el MLSA proporciona una estimación más robusta y precisa de las historias evolutivas en comparación con los análisis basados en un solo gen.

5.6.1. Métodos de inferencia filogenética

La inferencia filogenética puede abordarse a través de diversos métodos computacionales, cada uno con sus fortalezas y limitaciones. Estos métodos se dividen generalmente en dos grandes categorías: métodos basados en distancias y métodos basados en caracteres.

Métodos basados en distancias

Los métodos basados en distancias estiman árboles evolutivos convirtiendo los datos de secuencias en una matriz de distancias, que representa la divergencia evolutiva entre pares de secuencias (Huang y Li, 2024). Algunos de los métodos más comunes en esta categoría son:

- Unión de Vecinos (*Neighbor Joining*, NJ): Este método de agrupamiento construye un árbol minimizando la longitud total de las ramas. Es eficiente computacionalmente y adecuado para grandes conjuntos de datos, lo que lo convierte en una opción popular para construir filogenias a partir de matrices de distancias. Sin embargo, asume que la tasa de evolución es constante entre los linajes, lo que no siempre es cierto.
- UPGMA (Método de Agrupamiento No Ponderado con Media Aritmética): Este método también construye un árbol a partir de una matriz de distancias, pero asume un reloj molecular constante, es decir, que todos los linajes evolucionan a la misma velocidad. Aunque es sencillo, esta suposición puede llevar a inexactitudes en conjuntos de datos más complejos, donde las tasas de evolución varían entre linajes.

Métodos Basados en Caracteres

Los métodos basados en caracteres, a diferencia de los basados en distancias, analizan directamente los caracteres nucleotídicos o aminoacídicos individuales en cada sitio de la alineación de secuencias. Utilizan modelos estadísticos para estimar la estructura del árbol más probable que explique los datos observados (Ronquist *et al.*, 2012). Entre los métodos más comunes se encuentran:

- Máxima Parsimonia (MP): Este método busca encontrar el árbol que requiera el menor número de cambios evolutivos para explicar los datos observados. Aunque los árboles parsimoniosos suelen alinearse con las expectativas evolutivas, este método puede volverse computacionalmente costoso con grandes conjuntos de datos y no tiene en cuenta las variaciones en las tasas de evolución entre sitios.
- Máxima Verosimilitud (ML): La aproximación de máxima verosimilitud construye árboles seleccionando la topología que maximiza la probabilidad de observar los datos de secuencia bajo un modelo evolutivo específico. Este método es muy flexible y permite la variación en las tasas evolutivas entre sitios, y suele producir árboles más precisos que los métodos más simples. No obstante, ML es intensivo en términos computacionales, especialmente para grandes conjuntos de datos.
- Inferencia Bayesiana: La inferencia bayesiana en filogenética es un enfoque probabilístico que aplica los principios del teorema de Bayes para estimar las relaciones evolutivas entre especies o secuencias de ADN. A diferencia de los métodos clásicos de inferencia filogenética, que buscan un solo árbol óptimo (como en la Máxima Parsimonia o la Máxima Verosimilitud), la inferencia bayesiana genera una distribución de probabilidad para un conjunto de árboles, lo que permite incorporar la incertidumbre en la estimación filogenética. Esta metodología no solo estima el árbol más probable, sino que también evalúa la credibilidad de múltiples árboles posibles, proporcionando una representación más completa de las posibles relaciones evolutivas. En filogenética, la inferencia bayesiana suele realizarse mediante algoritmos de Monte Carlo con cadenas de Markov (MCMC). Este enfoque permite muestrear árboles a partir de la distribución de probabilidad posterior, lo que hace posible generar un conjunto de árboles que reflejan diferentes hipótesis filogenéticas con sus respectivas probabilidades.La inferencia bayesiana evalúa el conjunto completo de árboles posibles y calcula sus probabilidades posteriores, integrando tanto la información contenida en los datos como los conocimientos previos. Este enfoque es útil cuando se quiere tomar en cuenta la incertidumbre en la estimación de la filogenia (Ronquist *et al.*, 2012).

5.6.2. Herramientas para la inferencia filogenética

Existen varias herramientas bioinformáticas diseñadas para implementar los métodos descritos. Por ejemplo, *Bio.Phylo* y *DendroPy* son bibliotecas de Python que permiten construir y visualizar árboles filogenéticos utilizando diversos métodos, como la Unión de Vecinos y Máxima Verosimilitud. Estas bibliotecas ofrecen flexibilidad en la construcción de flujos de trabajo para la inferencia filogenética y son ampliamente utilizadas en proyectos de MLSA (Huang y Li, 2024).

Bio.Phylo

Bio.Phylo, un módulo de *Biopython*, proporciona funciones para leer, escribir y manipular árboles filogenéticos. Soporta la construcción de árboles a partir de matrices de distancias mediante métodos como la Unión de Vecinos. Aunque es relativamente sencillo de usar, sus limitaciones en el manejo de grandes conjuntos de datos y la realización de inferencias más complejas (como la inferencia bayesiana) pueden requerir el uso de herramientas más especializadas (Kim *et al.*, 2016).

DendroPy

DendroPy es una biblioteca completa para la computación filogenética, que soporta una amplia gama de tareas, desde la construcción de árboles hasta la simulación y comparación de árboles. Una de las fortalezas de *DendroPy* es su capacidad para manejar la estimación de árboles por Máxima Verosimilitud, así como su soporte para la manipulación avanzada de datos. La flexibilidad de esta herramienta la hace ideal para flujos de trabajo más complejos de MLSA (Huang y Li, 2024).

MrBayes

MrBayes es una herramienta especializada en la inferencia bayesiana de filogenias, utilizada frecuentemente cuando los investigadores necesitan estimar árboles filogenéticos que tengan en cuenta la incertidumbre en los parámetros del modelo y la topología del árbol. Es ampliamente utilizada por su capacidad para integrar modelos evolutivos y calcular probabilidades posteriores de los árboles. Sin embargo, requiere recursos computacionales elevados, especialmente para conjuntos de datos grandes (Ronquist *et al.*, 2012).

CAPÍTULO 6

Marco metológico

La metodología de este proyecto se separa en fases para mejorar la comprensión del proceso secuencial de desarrollar un *pipeline* bioinformático. La fase 1 corresponde al diseño de un *pipeline* de MLSA.

6.1. Fase 1: Diseño e implementación de un *pipeline*

Esta primera fase comprende todo el planteamiento teórico de los pasos a seguir para completar un análisis filogenético multi locus y la implementación de la herramienta. Para esto se definieron tecnologías a utilizar en cada uno de los pasos del proceso, al igual que las opciones e interacciones de los usuarios con la herramienta y se implementaron a una herramienta funcional que provee un producto mínimo viable para hacer análisis MLSA.

6.1.1. Carga de archivos

El primer paso, corresponde a la carga de archivos en un formato estándar para el correcto funcionamiento de la herramienta. Se definió que la herramienta iba a ser capaz de leer archivos en formato FASTA y FASTQ. Estos archivos se alimentarán a la herramienta de forma individual, colocando un archivo para cada locus y cada organismo. Esto permitirá la modularización de las secuencias. Es decir, un usuario puede realizar cambios a la secuencia utilizada simplemente cambiando el archivo. Para que el programa pueda identificar adecuadamente cada una de las secuencias. El usuario debe llenar un archivo en excel llamado ***input.xlsx*** este archivo contendrá en forma de matriz los nombres de los archivos de secuencias que se estarán trabajando en la herramienta.

La Tabla 1 muestra un ejemplo de la estructura de entrada diseñada para la herramienta.

Organism	16S rRNA	recA	rpoB
Escherichia coli	E_coli_16S.fasta	E_coli_recA.fasta	E_coli_rpoB.fasta
Bacillus subtilis	B_subtilis_16S.fasta	B_subtilis_recA.fasta	B_subtilis_rpoB.fasta
Saccharomyces cerevisiae	S_cerevisiae_16S.fasta	S_cerevisiae_recA.fasta	S_cerevisiae_rpoB.fasta

Cuadro 1: Ejemplo de ***input.xlsx*** para la entrada de datos en la herramienta MLSA-pipeline

En ésta, la primera columna contiene un listado de organismos, para este listado el usuario elige los nombres que quiere de cada organismo. En la primera fila se colocan los nombres de los genes que se estarán comparando en el análisis y en cada posición *(x,y)* se colocan los nombres de los archivos con su extensión correspondiente. El programa asume que los archivos de secuencias se encuentran en la carpeta seq del directorio de trabajo.

6.1.2. Alineación de secuencias

El segundo paso del diseño corresponde a la alineación de secuencias. La Figura 2 muestra un diagrama de flujo del proceso a seguir para interactuar con la herramienta. El primer paso que se realiza es la selección de uno de los dos algortimos de alineación que se proveen en la herramienta. Estos pueden ser **ClustalOmega** o **MUSCLE**, luego se realiza un paso de verificación de la instalación correcta de la herramienta seleccionada. Ya que estas herramientas son externas al *pipeline*. No son desarrolladas internamente. Si todo está funcionando correctamente se inicia el proceso de parametrización, de lo contrario se levanta una alerta para que el usuario pueda revisar que ha realizado una instalación correcta. En el paso de parametrización se le dan dos opciones al usuario:

1. Elegir la parametrización por defecto: Se provee una parametrización mínima que permite al usuario ejecutar la herramienta de forma correcta.
2. Parametrización experta: La herramienta permite al usuario proveer su propio archivo de configuraciones que sobreescribe por completo al archivo de configuración por defecto. Esto permite a usuarios con más experiencia colocar todos los parámetros que requieran para el análisis.

Luego de tener todas las configuraciones, la herramienta va a cada una de las columnas del archivo ***input.xlsx*** y realiza el proceso de alineación. Como resultado, la herramienta entrega una carpeta llamada *align* que contiene un archivo .FASTA para cada una de las columas correspondientes. Esto alinea todas las secuencias de ese gen de todos los organismos. Además provee un archivo *log* con las salidas de la herramienta seleccionada. Estas salidas no tienen ningún tipo de interpretación adicional por el *pipeline* y se entregan al usuario tal y como las produce la herramienta original. Con esto se completa el paso de alineación y el usuario termina con archivos que pueden ser utilizados por el paso de concatenación de forma adecuada.

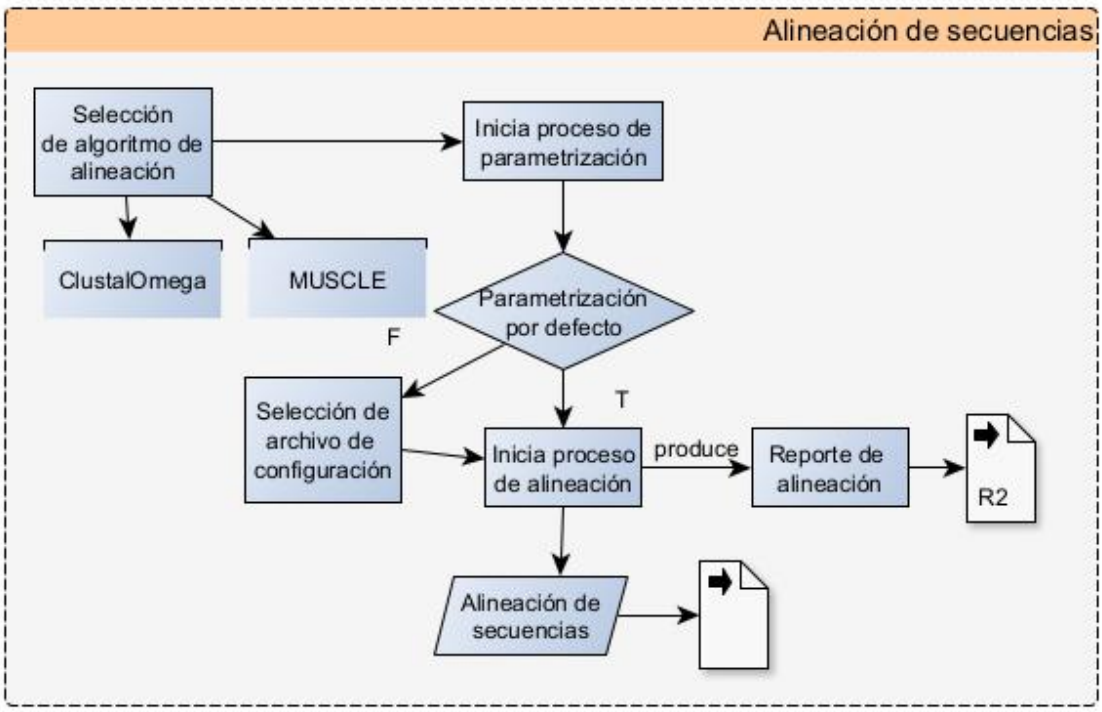

Figura 2: Diagrama de flujo de alineación de secuencias

6.1.3. Concatenación de secuencias

El siguiente paso correspode a la concatenación de las secuencias. En este paso se toman todos los archivos resultantes del paso anterior y se concatenan a un solo archivo .FASTA. Cómo se puede observar en la Figura 3 las secuencia de pasos es muy similar a la alineación. Se elige una herramienta de análisis, se verifica que esta esté instalada correctamente, se parametriza según las necesidades del usuario (por defecto o experta) y luego se ejecuta el algoritmo de concatenación sobre las secuencias disponibles en el directorio *align*. Esto produce un archivo *log* con las salidas del programa seleccionado y un archivo llamado **concatenate.FASTA** en el directorio *concatenate* del directorio de trabajo.

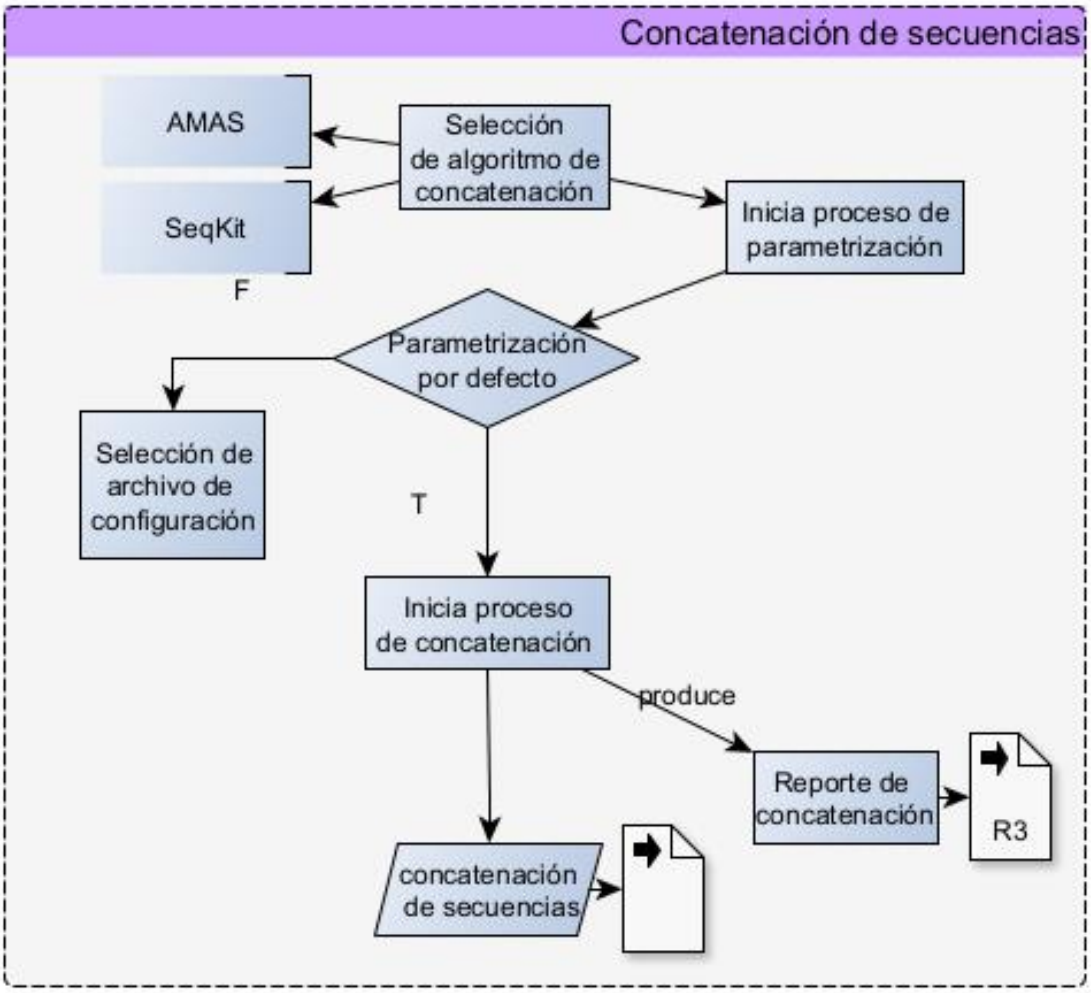

Figura 3: Diagrama de flujo de concatenación de secuencias

6.1.4. Inferencia filogenética

El siguiente paso corresponde a la inferencia filogenética. En este paso se realizan todos los análisis que producen un árbol filogenético de las secuencias realizadas, el producto final de este *pipeline*. En este paso, se sigue la misma lógica que en los pasos anteriores como puede observarse en la Figura 4. En este paso los resultados (tanto el archivo *log*, como el archivo de resultados de la inferencia) se almacenan en el directorio *inference* del directorio de trabajo. A diferencia de los pasos anteriores, este paso produce un reporte interactivo, el cual contiene el árbol filogenético resultante. Esta herramienta de visualización se describirá a detalle en la siguiente fase de la herramienta.

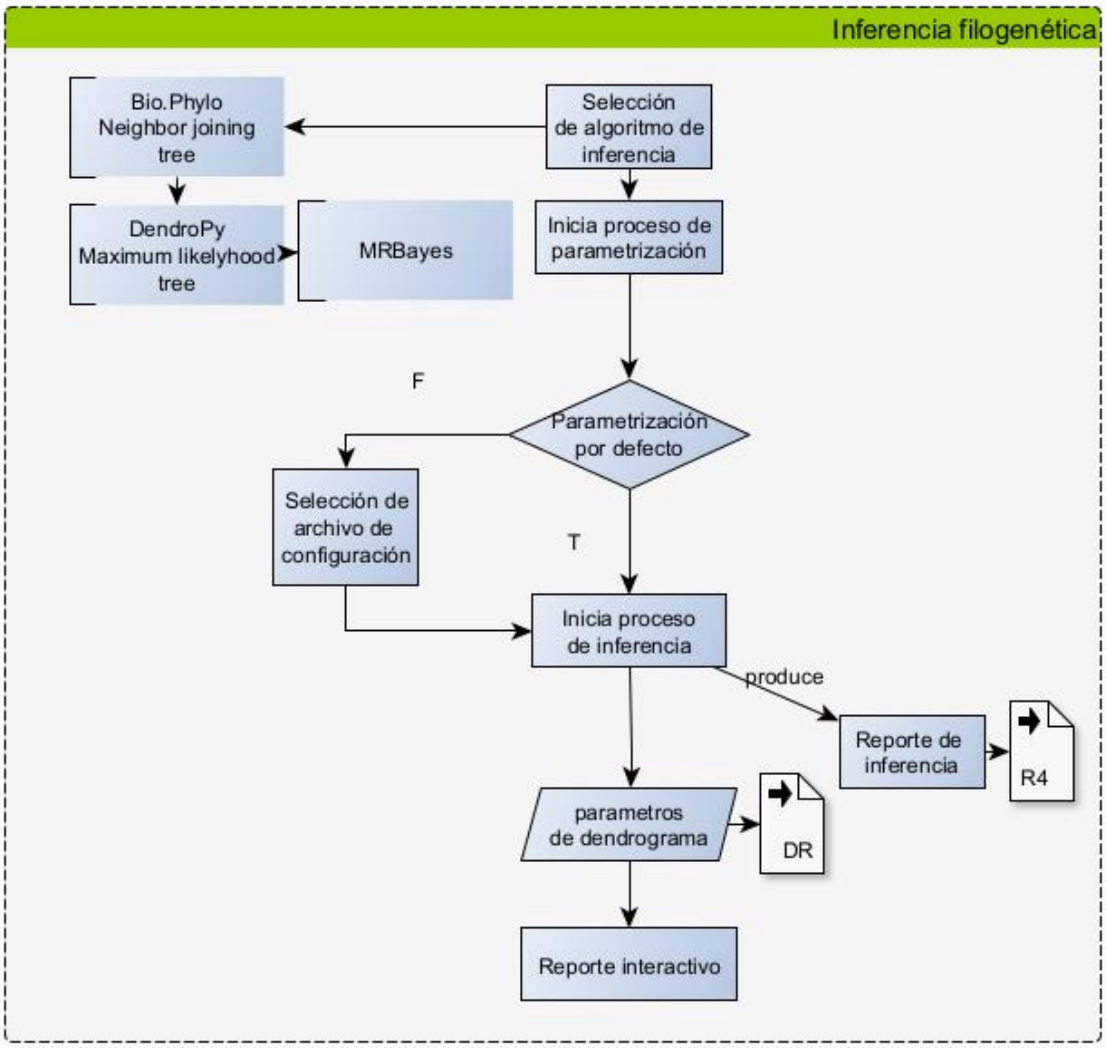

Figura 4: Diagrama de flujo de inferencia filogenética

6.1.5. Implementación de la herramienta

El desarrollo de un *pipeline* bioinformático cuenta con varias partes que deben ser orquestradas entre sí. Para implementar esta herramienta, se trabajó con tecnologías estándar en la industria actual. Cómo primer paso de la implementación se desarrolló un contenedor de **Docker**. Esta es una herramienta que permite hacer ambientes autocontenidos que tienen todas las herramientas necesarias para que se ejecute adecuadamente. El contenedor de docker utilizado es una instalación del sistema operativo Ubuntu Linux 20.04, compatible con todas las herramientas descritas anteriormente. En este contenedor se instalaron todas las herramientas necesarias incluyendo **Python**, los paquetes de **Python** que se requieren para los pasos 2 y 3 del *pipeline*, **Snakemake**, además de inicializar la herramienta con la cantidad de núcleos del computador. Este contenedor de **Docker** se debe inicializar especificando un directorio de montaje de archivos y el comando de inicialización de **Snakemake**.

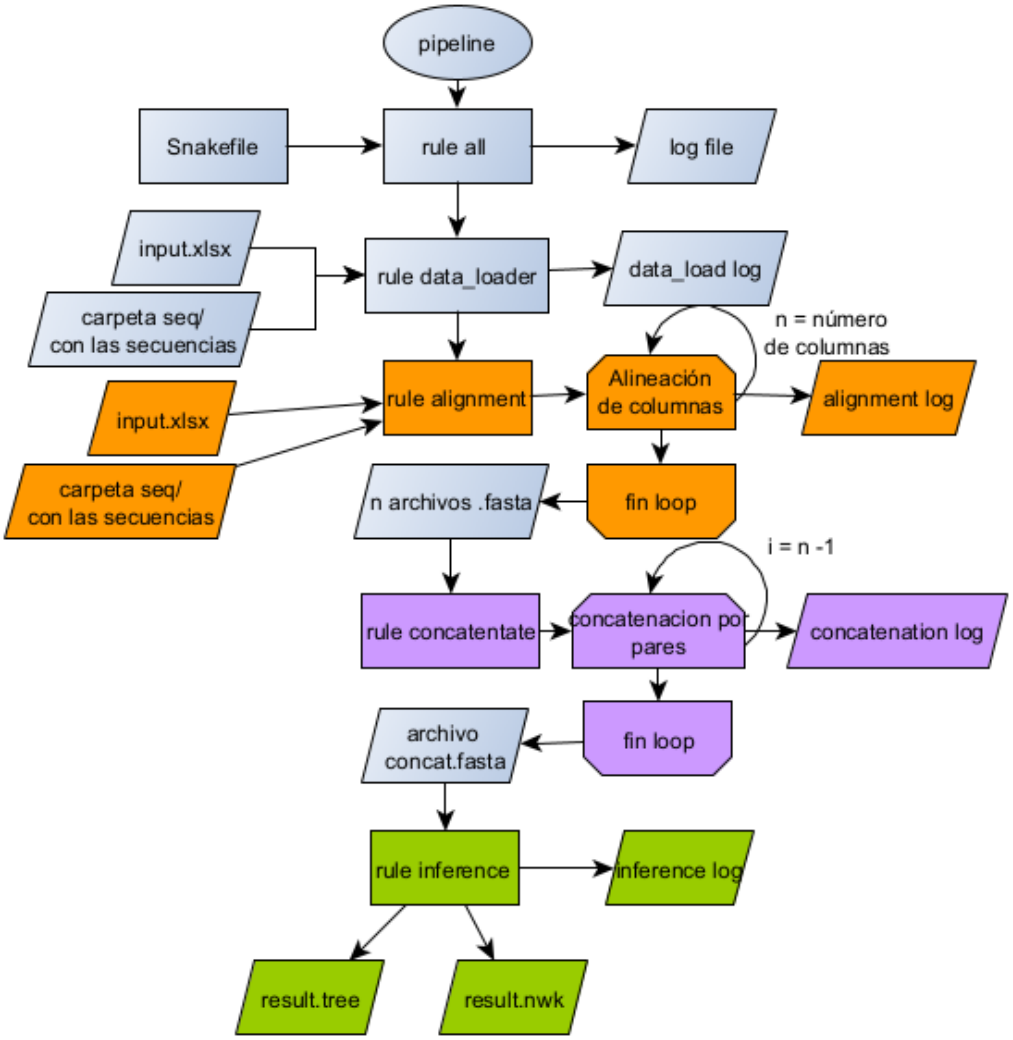

Figura 5: Secuencia de procesos implementados en **Snakemake**, cada proceso lleva un archivo de control llamado *rule*

Para la implementación de los pasos de **Snakemake** se inicializó un ***Snakefile***, esta es la sinmtaxis de la herramienta, escrita en Python para configurar los pasos. El árbol de procesos de **Snakemake** está representado en Figura 5 en donde se muestran las entradas y salidas del *pipeline* y cada uno de los pasos. **Snakemake** utiliza reglas de operación llamadas *rules*. Cada una de estas reglas referencia a un archivo en Python del mismo nombre que contiene todos los pasos necesarios para que funcione el programa específico. El diagrama muestra estas reglas para cada uno de los pasos y muestra la cantidad de iteraciones que se realizan en cada paso para obtener el resultado esperado. Uno de los pasos más importantes de este *pipeline* es la correcta determinación de los formatos de salida de cada una de las herramientas. Por defecto, estas herramientas usualmente tienen formatos propios de salida (.afa para **MUSCLE**, .clw para **ClustalW**, etc) por lo que debe estandarizarse la salida en cada uno de los pasos. Para algunas de las herramientas la estandarización es un simple cambio de parámetros en el proceso de ejecución pero para otras, que no proveen

esa facilidad se implementó un convertidor del formato específico a formato .fasta. La salida final de la herramienta se entrega en formato Newick, utilizando dos extensiones distintas. Una extensión genérica (.tree) y una extensión específica del formato (.nwk). Este formato devuelve un grafo anidado el cual es fácil de diagramar para la visualización de los resultados.

Todo el código implementado en esta herramienta está disponible en el repositorio de github: https://github.com/Rafalp190/MLSA-pipeline. Para su fácil uso y modificación. Se trabajó solamente con herramientas de uso libre y gratuitas para garantizar la longevidad del proyecto. El repositorio se encuentra bajo una licencia *GNU General Public License v3.0* por lo que el proyecto puede ser utilizado para fines comerciales. Cualquier resultado del mal uso de la herramienta no es responsabilidad del autor original.

6.2. Fase 2: Implementación de la herramienta de reporte

Para la implementación de la herramienta de reporte se pensó en una herramienta visual que permita a los usuarios explorar los resultados de los distintos pasos de la herramienta. Haciendo un enfoque especial en el dendrograma resultante. Esta herramienta se desarrolló en el *framework* **shiny** para Python, para realizar las visualizaciones interactivas se utilizó la librería **plotly**. La herramienta de visualización se ejecuta automáticamente al terminar la ejecución y queda disponible como un sitio web local desde el contenedor de **docker**. Se puede acceder al sitio en cualquier momento que el contenedor de docker esté activado y que el proceso se haya completado satisfactoriamente en el URL: http://localhost:1995

6.3. Fase 3: Validación de la herramienta

Para la validación de la herramienta se decidió replicar el análisis *Multilocus Sequence Analysis, a Rapid and Accurate Tool for Taxonomic Classification, Evolutionary Relationship Determination, and Population Biology Studies of the Genus Shewanella* realizado por Fang, Yujie, et al., en 2019 (Fang, Wang, Liu, Dai, Cai, Li *et al.*, 2019b). En este estudio se realizó un análisis MLSA del género *Shewanella* utilizando seis genes HKG: *gyrA, gyrB, infB, recN, rpoA, y topA*. Se eligió este estudio por la disponibilidad de las secuencias de referencias a través de una matriz con los enlaces de acceso a Genbank. La matriz de entrada puede accederse en el repositorio de github.

CAPÍTULO 7

Resultados y Discusión

La presentación de resultados de esta herramienta se centra en la evaluación de las tecnologías utilizadas para cada fase del proceso de implementación. Se desarrolló una herramienta con varias partes que requirió la integración de distintas herramientas tanto para la creación del ambiente, el *pipeline*, la alineación y concatenación, la inferencia filogenética y la herramienta de reporte. Adicionalmente, se validó el correcto funcionamiento de la herramienta, replicando un estudio filogenético bien documentado y con disponibilidad de acceso a los datos de referencia utilizados para realizar el estudio.

7.1. *Pipeline* para análisis filogenético multi-locus

La primera de las herramientas a discutir es la herramienta **Docker**. Al usar Docker, se creó un ambiente auto contenido que permitió instalar un sistema operativo Linux. Se optó por instalar Ubuntu 20.04, para mantener compatibilidad con todas las herramientas que se requirieron para la investigación. No se utilizó una versión superior de Ubuntu ya que se tuvo complicaciones para instalar la herramienta SeqKit. Se determinó que este error ocurría en cualquier versión superior a 20.xx de Ubuntu, por lo que esta herramienta limitó utilizar una versión más nueva. Existe la posibilidad que sea un problema de compatibilidad que los desarrolladores de SeqKit solucionen en el futuro. Al momento que se redactó este documento, la versión estable más reciente de Ubuntu disponible como imagen en los repositorios de Docker, es la versión 24.04. Se instaló la versión más reciente de Python estable disponible en los repositorios de APT de Debian Linux, no se especifica la versión de instalación en el archivo de configuración (*Dockerfile*) para permitir que se continúe actualizando automáticamente cada vez que se construya un nuevo repositorio. Las herramientas **ClustalW** y **MUSCLE**, para alineación de secuencias, también se descargaron directamente de los repositorios de APT. Esto asegura la compatibilidad directa con linux y facilita el proceso de instalación.

SeqKit y MrBayes no tienen una versión en APT, entonces siguieron los procesos indicados en la documentación de cada una de ellas para instalarlas. El proceso implicó descargar la versión más reciente del repositorio de github de las herramientas, extraerlas, instalarlas y asegurar que los permisos de usuario fuesen los correctos para que se pueda acceder a estas herramientas desde la consola del contenedor. El siguiente paso consisitió en instalar todos los paquetes de Python requeridos para la herramienta. Esto incluyó algunos paquetes para la lectura de datos, una herramienta de concatenación desarrollada en Python, BioPython para tener herramientas generales para trabajar con información Bioinformática y otras dependencias requeridas. Los requerimientos completos instalados se pueden encontrar en el archivo requirements.txt disponible en el repositorio de github del proyecto.

Docker permitió solucionar una de las complicaciones principales de desarrollar esta herramienta: La instalación de los distintos programas necesarios para el uso del *pipeline*. Al instalar todo dentro de un contenedor de docker, los usuarios no deben preocuparse del proceso de instalación. Todos las complicaciones generadas por versionamiento de programas, descarga de instaladores y compatibilidad de sistema operativo se han resuelto por ellos. Esto facilita el uso del *pipeline* y de las herramientas que este *pipeline* contiene. Otra de las inconveniencias de uso que mejoró es el mantenimiento de un directorio de trabajo. Este directorio se mantiene en el computador del usuario y este se monta virtualmente al contenedor. Esto significa que el usuario puede utilizar la interfaz de manejo de archivos de su sistema operativo de preferencia para interactuar con los archivos que entran y salen del sistema. Esto facilita la introducción de nuevas sequencias a la matriz de ***input.xlsx*** y al directorio ***seq/***. Sin embargo, Docker no soluciona el problema de tener una herramienta en consola, ya que esta herramienta no tiene una interfaz gráfica. Los contenedores son un pequeño sistema operativo al que accedemos desde una terminal. La interfaz gráfica de *Docker for Windows* mejora un poco esta interacción (Docker, Inc., 2024) pero al mismo tiempo incrementa la curva de aprendizaje para entrar y utilizar un contenedor. Por lo que se recomienda que esta herramienta se utilice desde una terminal o desde un ambiente integrado de trabajo que tenga acceso a una terminal del sistema operativo como Visual Studio Code o Notepad ++.

Otro de los problemas que no soluciona, especialmente para sistemas Windows, es que se requiere activar una funcionalidad de Windows que no viene activa por defecto, el subsistema de Linux para Windows (WSL por sus siglas en ingles). Esta herramienta permite tener un pequeño sistema Linux en el computador, y es a través de esta capabilidad que Docker puede crear una máquina virtual de linux y ejecutar comandos de linux desde un sistema operativo Windows. Activar el subsistema es un proceso que se ha vuelto más sencillo en Windows 11 (Microsoft Corporation, 2024) pero que puede requerir de pasos adicionales en computadores con Windows 10 ((Microsoft Corporation, 2024)). Adicionalmente, WSL fue implementado en Windows 10, eso quiere decir que la herramienta no es compatible con versiones anteriores de Windows ((Docker, Inc., 2024)) y que nunca podrá serlo.

La siguiente de las herramientas fundamentales para este *pipeline* es **Snakemake**, esta herramienta, desarrollada en python, está especialmente diseñada para el desarrollo de *pipelines* y permitió hilar todos los procesos de forma coherente y secuencial, permitiendo utilizar código de Python y todas las herramientas de manipulación de datos del paquete Pandas, para realizar las transformaciones necesarias para convertir la salida de una herramienta en la entrada del siguiente paso. Como se puede observar en la Figura 5 todos

los pasos producen una salida que debe ser entregada al siguiente paso del proceso para su utilización. Esto puede ser hilado a través de la manipulación de archivos en el directorio de trabajo montado en el contenedor de **Docker** mencionado anteriormente.

El programa **Snakemake**, también permite a los usuarios especificar la cantidad de núcleos de sistema operativo con los que desea ejecutar el *pipeline*, permitiendo, en conjunto con el contenedor de**Docker**, restringir la cantidad de recursos que se asignan a la ejecución de los procesos de un análisis MLSA. Comparado con procesos tradicionales de implementación de *pipelines* como Bash, **Snakemake** reduce el tiempo de ejecución de procesos de forma significativa, especialmente cuando se trabaja con una menor cantidad de núcleos de sistema operativo (Loecker y Ewing, 2021). Esto es beneficioso para contextos de escasos recursos ya que, a pesar de que las diferencias en ejecución se miden en segundos, estas cuentan más cuando las computadoras no son tan poderosas. En Guatemala no existen estadísticas sobre el poder de procesamiento promedio de computadores, pero al ser un país en vías de desarrollo (Inter-American Development Bank, 2024), el acceso a la tecnología de última generación es más limitado y en general llega al país más tarde. Pensar en formas de optimizar recursos cumple con uno de las problemáticas resaltadas en la justificación de esta investigación y es una de las principales razones por las que se eligió **Snakemake** por encima de otras tecnologías para el diseño de *pipelines*.

Otra de las razones por las que se eligió **Snakemake**, es que al igual que Python, es una herramienta de código abierto y de uso libre. Lo que permite ofrecer este *pipeline* sin ningún costo asociado a su instalación, uso o modificación. Lo que también ayuda a solucionar la barrera de ingreso más grande en contextos de bajos recursos: el precio de las herramientas y los costos asociados a su mantenimiento (Kapitsaki *et al.*, 2015).

En general, la mayoría de desventajas asociadas con el uso de **Snakemake** como herramienta para la implementación de *pipelines* se asocian al proceso de desarrollo. Esta herramienta tiene una curva de aprendizaje elevada, que complejiza desarrollar *pipelines* nuevos, no ofrece ayudas para el manejo de paquetes ni dependencias, es complicada de escalar a sistemas paralelos y requiere una clara identificación de las entradas y salidas de cada proceso de ejecución. Sin embargo, esos problemas no son notorios para el usuario final del *pipeline*. el usuario se enfrenta a un contenedor de Docker que ya tiene todo previamente configurado y siempre que las entradas del paso 1 de ejecución estén correctas, el *pipeline* debería encargarse de que todos los procesos ocurran de forma coherente (Köster y Rahmann, 2012).

A pesar de esto, una de las dificultades con las que el usuario puede enfrentarse es con el manejo de errores. La herramienta desarrollada hace todo lo posible por hacer la lectura de los archivos de log de cada paso accesible al usuario. Sin embargo, en caso de que ocurra un error no considerado, este puede ser complejo de interpretar. Considero que esta es una de las principales debilidades de la tecnología utilizada para esta herramienta. La interpretación de errores es bastante compleja cuando se trabaja con **Snakemake** y es posible que estos errores sean trasladados al usuario sin ninguna forma de proveerle ayuda adicional para resolverlos.

El proceso de revisión de datos faltantes es el paso inicial de la ejecución del *pipeline* de análisis. Para este proceso la herramienta realiza un proceso de lectura del archivo ***input.txt*** que contiene las secuencias. Para cada una de las secuencias se leyó el nombre del archivo .fa,

.fasta, .fq o .fastq correspondiente y se utilizó el paquete Bioconductor para cargar el archivo a una variable temporal. Este paquete tiene funcionalidades para determinar si un archivo en estos formatos es válido. Devolviendo un archivo de *log* llamado ***data_load_FECHAHORA.txt*** cómo el que se muestra en la Figura 6. Este archivo contiene el número de organismos que se revisaron, cuántos locus existen por organismo, garantiza que todos los archivos estén presentes e informa si todas las secuencias que se tienen son válidas. En caso que alguna secuencia no sea válida o que no esté presente se indica que no se encontró y se interrumpe el proceso de ejecución luego de notificar que una o más no son correctas. Este proceso garantiza que se entre a la alineación de secuencias con información coherente y completa. Es importante notar que este proceso no almacena en memoria a largo plazo las secuencias que lee y revisa, a simple vista esto parece generar una doble ejecución del proceso, ya que en el paso de alineación se deberan leer una por una las columnas del excel para poder alinearlas entre sí. Se decidió separar los procesos y leer dos veces a pesar del incremento que esto pudo tener en el tiempo de ejecución ya que nos permite compartimentalizar los distintos pasos del *pipeline*, permitiendo que el usuario decida saltarse pasos en ejecuciones subsiguientes. Cada uno de los pasos es autocontenido y a pesar de que requieren de la salida del paso anterior, no hace falta que falle un proceso de alineación que es más costoso en recursos para identificar errores en la entrada de datos (Cock *et al.*, 2009).

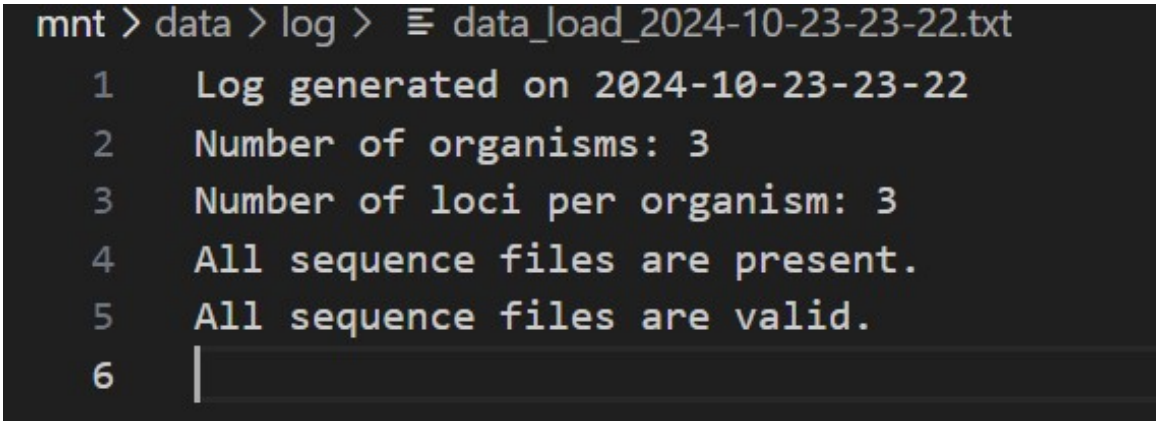

Figura 6: Ejemplo de salida de ejecución del *log del proceso de lectura y verificación de archivos.*

Cuando se trabaja con muchas secuencias, este proceso de verificación contempla un porcentaje mínimo del tiempo de ejecución, comparado con los tiempos de alineación y con los otros pasos del proceso completo y a la larga ahorra tiempo al usuario identificando errores temprano sin que ocurra por ejemplo, en la sexta alineación de un proceso de alineación de siete loci. Por la forma en la que funciona **Snakemake**, esto implicaría que se debe ejecutar nuevamente las seis alineaciones que ya se ejecutaron para corregir la séptima.

Para implementar el proceso de alineación se tuvo que lidiar con varias condiciones. Según lo planteado en la metodología, la herramienta se implementó con la capacidad de cambiar de algoritmo y de tener parámetros por defecto o parámetros especializados. Esto se logró a través de un archivo YAML de configuración (***config.yaml***) que se colocó en la raíz *mnt/data* del directorio de trabajo del proyecto. Éste archivo de configuración permite al usuario hacer modificaciones a la ejecución del paso de alineación. Ya que se provee la opción de cambiar de herramienta de alineación para el proceso, también contiene un parámetro ***aligner*** que permite elegir cual herramienta utilizar. Por defecto se eligió utilizar ***MUSCLE***.

Como se muestra en el Bloque de código 7.1, el archivo de configuración inicia con la selección del alineador con el parámetro ***aligner***, el segundo parámetro es la cantidad de cores de ejecución del *pipeline*. Luego se introduce la sección ***muscle params*** que permite elegir los parámetros para alinear con **MUSCLE**. Los parámetros configurables son la cantidad de iteraciones (16, por defecto), si se desea alinear diagonales (*false*, por defecto), el costo de creación de nuevos espacios vacíos en la alineacion con ***gapopen***, el costo de extensión de los espacios vacíos con el parámetro ***gapextend***, la cantidad de *megabytes* de memoria ram que la herramienta utilizará para el análisis con el parámetro ***maxmb***. Por defecto este parámetro se especifica como *null*, esto quiere decir que la herramienta utilizará todos los recursos disponibles para realizar el proceso de alineación. Adicionalmente la herramienta **MUSCLE** da la opción de elegir un algoritmo de agrupación que utiliza en el proceso. Por defecto esta utiliza el algoritmo ***neighbor joining***.

Bloque de código 7.1: Ejemplo de archivo ***config.yaml*** que contiene las configuraciones necesarias para alinear la herramienta

```
# Alignment step
aligner: "clustalw"     # "clustalw" or "muscle"
## Muscle5 alignment parameters available to set
## for more information check:
muscle_params:
  perturb: 0            # Integer random number seed
  perm: "none"          # Guide tree permutation, default "none"
  stratified: false     # Generate stratified ensemble
  diversified: false    # Generate diversified ensemble
  replicates: 4         # Number of replicates
  consiters: 2          # Number of consistency iterations
  refineiters: 100      # Number of refinement iterations

## Clustalw parameters available to set
## for more information check:
clustalw_params:
  gapopen: 15.0         # Gap open weight cost
  gapextend: 0.2        # Gap extend weight cost
  numiter: 5            # Number of iterations
  matrix: "BLOSUM"      # Matrix to use
```

Cuando el usuario elige realizar la alineación con **ClustalW**, también se incluyen los parámetros específicos para esta herramienta. Se incluyen los mismos parámetros de ***gapopen*** y ***gapextend*** pero con la sintaxis de **ClustalW**, se incluye el parámetro de cantidad de iteraciones: ***numiter***, el parámetro de matriz de alineación que se utiliza: ***matrix***. Cada uno de estos parámetros permite mejorar o empeorar el rendimiento de la herramienta (Cashman *et al.*, 2023).

Los parámetros que se eligieron por defecto para el proceso de alineación, son los parámetros recomendados por ambas herramientas para realizar un análisis de calidad promedio. El usuario puede modificar los parámetros a su gusto utilizando los archivos de configuración. En teoría, utilizar los parámetros por defecto debería ser lo suficientemente bueno para obtener un árbol filogenético en el proceso de inferencia (Cashman *et al.*, 2023). La para-

metrización de la herramienta en este paso es uno de las principales formas en las que se mejora la calidad del análisis final (Ashkenazy *et al.*, 2018). Proveer a los usuarios de esta funcionalidad adicional en el proceso de alineación agrega flexibilidad al proceso de alineación tanto con ***MUSCLE*** como con ***ClustalW***. Esto cumple con el objetivo específico 2 de los objetivos planteados en esta investigación, permitir al usuario alinear secuencias multi locus y secuencias de referencia utilizando las herramientas mencionadas.

El paso de concatenación es mucho más sencillo al paso de alineación, pero se siguió la misma lógica. Se implementaron dos herramientas: **Seqkit** y **AMAS** y se permitió a través del archivo de configuración ***config.yaml*** elegir entre ellas y parametrizar cada una de ellas. Los procesos de concatenación no requiren de tanta parametrización adicional ya que solamente pegan las secuencias alineadas entre sí. Sin embargo, se especificaron algunos de los parámetros disponibles en cada una de las herramientas.

Es importante notar que no se incluyen parámetros que permitan cambiar el formato de salida, esto para poder garantizar que los algoritmos puedan ser ejecutados en secuencia. En todos los pasos en donde es posible se parametrizó la salida a formato Fasta. Tanto las herramientas del proceso de alineación como las de concatenación permiten la salida en fasta.

Bloque de código 7.2: Ejemplo de archivo ***config.yaml*** que contiene las configuraciones necesarias para concatenar la herramienta

```
# Concatenation step
concatenator: "seqkit"  # Choose between "amas" or "seqkit"

## AMAS parameters available to set
## for more information check:
amas_params:
  check_align: false

## SEQKIT parameters available to set
## for more information check:
seqkit_params:
  full: true
```

Como se muestra en el Bloque de código 7.2 este segmento del archivo config permite elegir entre AMAS y Seqkit y parametrizar cada uno de ellos por defecto o utilizando sus parámetros. Al ser uno de los pasos más sencillos del proceso, las herramientas no tienen tántos parámetros que elegir. Sin embargo, se incluyen algunos que incluyen revisiones adicionales (para AMAS) o que permiten trabajar con secuencias incompletas agregando las faltantes al archivo concatenado (Mccauley *et al.*, 2008). El paso de concatenación tambien es el momento ideal para generar la imagen de las secuencias alineadas y concatenadas que mostrará la herramienta de reporte. Así que se incorporó en este paso esa funcionalidad. Sin embargo, esto incrementa el tiempo de ejecución de la herramienta.

En el proceso de inferencia filogenética los algoritmos que se implementaron fueron: *Neighbor joining tree (NJT)* del paquete ***BioPhylo***, el algoritmo *Maximum likelyhood tree (MLT)* del paquete ***DendroPy*** y la herramienta MrBayes, con la opción de seleccionar algoritmos distintos de los implementados en ésta herramienta. La elección de permitir al usuario

elegir entre modelos determinísticos como NJT y MLT dan la opción de realizar un análisis mucho más veloz. Los algoritmos probabilísticos de ***MRBayes*** son mucho más costosos de realizar y toman más tiempo (Ronquist *et al.*, 2012) pero obtienen resultados mejores. La flexibilidad de selección ayuda a que la herramienta pueda ejecutarse en contextos donde no se tienen computadores tan potentes o que pueda escalarse a ejecución en sistemas más poderosos.

Para la parametrización del proceso de inferencia se siguió la misma lógica, primero se elige que herramienta utilizar con el parámetro ***inferer*** y luego se pueden configurar cada una de ellas por aparte utilizando configuraciones por defecto o cambiando algunos parámetros para empeorar o mejorar los resultados.

Como se observa en el Bloque de código 7.3, el proceso de inferencia permite la selección de distintos algoritmos con sus respectivas opciones configurables. Por ejemplo, para el algoritmo ***Neighbor Joining*** (NJT) del paquete ***BioPhylo***, se permite seleccionar el ***distance metric*** para definir el modelo de cálculo de distancias, con opciones como *jukes cantor* y *kimura* que pueden ajustarse según el tipo de datos. Además, se incluye la opción de establecer un grupo externo con el parámetro ***outgroup***, el cual puede ser útil en el análisis para enraizar el árbol en un punto específico.

Bloque de código 7.3: Ejemplo de archivo ***config.yaml*** que contiene las configuraciones necesarias para el paso de inferencia

```
inferer: "mrbayes"
neighbor_joining_params:
    distance_metric: "jukes_cantor"
    outgroup: null

maximum_likelihood_params:
    substitution_model: "GTR"
    bootstrap_replicates: 100

mrbayes_params:
    algorithm: "mcmc"
    ngen: 100000
    samplefreq: 100
    nchains: 4
    burnin: 25000
```

Para el algoritmo ***Maximum Likelihood Tree*** (MLT) del paquete ***DendroPy***, el archivo de configuración permite especificar el modelo de sustitución de nucleótidos (***substitution model***), incluyendo modelos comunes como el modelo *GTR* que se establece como predeterminado, y el número de réplicas ***bootstrap*** que el usuario desee para obtener una estimación del soporte de los nodos en el árbol resultante.

Finalmente, para la herramienta ***MrBayes***, se pueden especificar opciones avanzadas para el análisis probabilístico. El parámetro ***algorithm*** permite elegir entre algoritmos implementados en MrBayes como *MCMC* o *Metropolis-coupled MCMC (MC3)*, ofreciendo mayor flexibilidad en los análisis de cadenas múltiples, especialmente para cadenas acopladas en contextos de distribuciones de probabilidad complejas. Adicionalmente, parámetros como

el número de generaciones (***ngen***), la frecuencia de muestreo (***samplefreq***), y la cantidad de cadenas (***nchains***) ayudan a optimizar el proceso para obtener resultados más detallados y específicos. El parámetro ***burnin*** se utiliza para descartar un número inicial de muestras y asegurar que se obtiene un muestreo representativo de la distribución posterior.

La flexibilidad de selección entre modelos determinísticos como NJT y MLT, y modelos probabilísticos como los algoritmos de ***MrBayes***, ofrece una ventaja significativa (Hall, 2004). Los modelos determinísticos proporcionan una ejecución rápida, lo cual es beneficioso para análisis preliminares o en entornos de cómputo limitados. Por otro lado, los algoritmos probabilísticos, aunque más costosos en términos de tiempo de cómputo, tienden a producir resultados más robustos y precisos (Hall, 2004). Esta flexibilidad es particularmente útil en contextos donde los recursos computacionales pueden ser limitados, permitiendo al usuario realizar inferencias rápidas con modelos determinísticos o realizar análisis detallados con algoritmos más complejos en computadoras de mayor potencia.

La parametrización en el archivo ***config.yaml*** permite, además, ajustar la precisión y el detalle del análisis en función de las necesidades del proyecto y los recursos disponibles. Esto asegura que la herramienta sea aplicable en una amplia variedad de contextos de investigación, desde análisis rápidos hasta inferencias filogenéticas complejas y detalladas.

7.2. Herramienta de reporte

Para el desarrollo la herramienta de reporte se diseñó primero un prototipo no funcional que fue presentado a actores clave, quienes proporcionaron realimentación sobre la información que la herramienta debería mostrar. En general, los cuatro expertos consultados decidieron los siguientes aspectos:

- La herramienta debía mostrar el árbol filogenético y permitir que este sea descargado
- La herramienta debía mostrar los puntajes de la inferencia filogenética de forma clara y permitir descargar el archivo .nwk.
- La herramienta debía poder mostrar el resultado de la alineación y concatenación de secuencias, preferiblemente en un visualizador interactivo del archivo fasta concatenado y permitir la descarga del mismo.
- La herramienta debía permitir descargar los archivos de *log* generados en cada uno de los procesos.

Los resultados completos del proceso de prototipado pueden ser consultados en la Sección 11.1. Siguiendo estas indicaciones, se procedió a desarrollar la herramienta de reporte utilizando el paquete *Shiny* para Python (Kasprzak *et al.*, 2021). *Shiny* es una herramienta originalmente desarrollada para realizar páginas web de visualización de datos en el lenguaje de programación R, pero se ha implementado una versión en Python que está activa desde el año 2022 (Kasprzak *et al.*, 2021). Para mantener consistencia entre las tecnologías utilizadas en este *pípeline* se decidió utilizar la versión en Python, a pesar que la versión en R es más robusta y tienen más años de implementación. Se realizó una evaluación de las capacidades

necesarias dado el listado de solicitudes de los expertos y se determinó que *Shiny* en python era capáz de solventar todos los requerimientos de esta herramienta. Para el usuario final, este será un cambio imperceptible, pero si implicó en el proceso de desarollo, la instalación de menos herramientas en el contenedor de *Docker* haciéndolo más compacto ya que no se tuvo que instalar R y todas las dependencias que *Shiny* requiere para funcionar.

Con respecto a la visualización del dendrograma se decidió utilizar un dendrograma jerárquico hacia abajo. Como se muestra en la Figura 7, en el dendrograma las distancias se representan por el largo del segmento en el eje Y, y las especies se encuentran en el eje X.

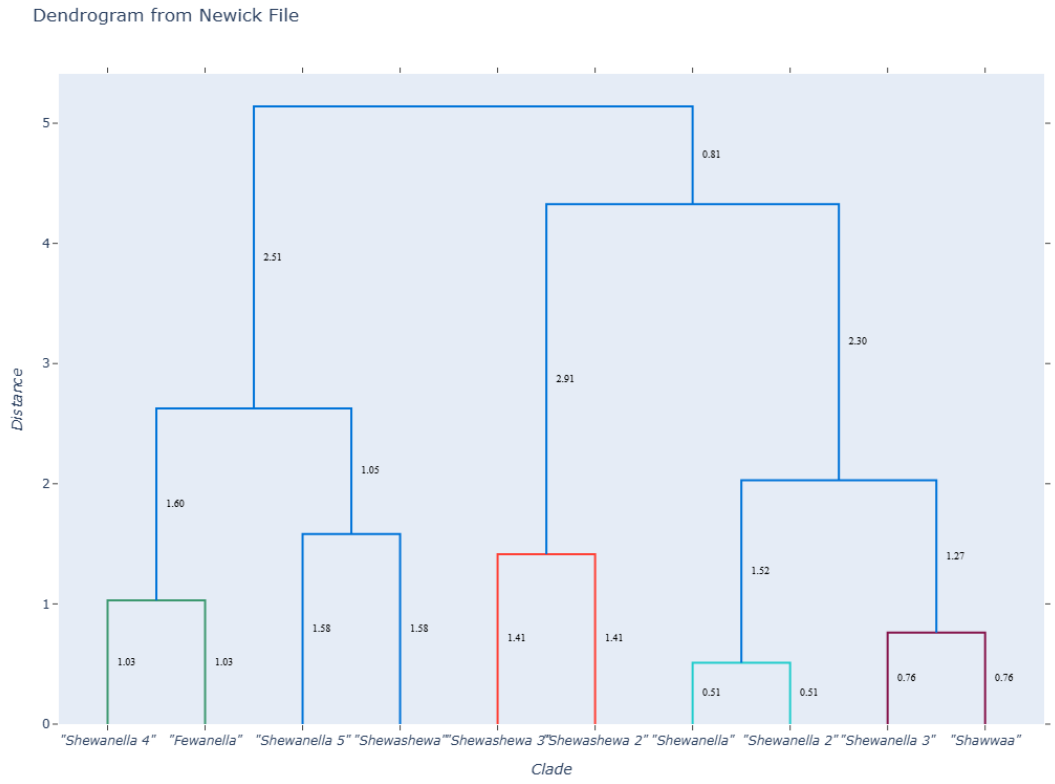

Figura 7: Ejemplo de dendrograma producido por el sitio web de *Shiny* del proyecto.

El dendrograma se realizó utilizando el paquete *Plotly*, el cual permite realizar gráficas de forma sencilla y agrega funcionalidades como la habilidad para hacer zoom a la imagen y tomar capturas de pantalla del gráfico en pantalla, lo que le da mucha flexibilidad al usuario para tomar capturas de pantalla de partes específicas del dendrograma (Schneider *et al.*, 2024). Se optó por utilizar colores para las últimas ramas del dendrograma para representar las parejas de organismos más cercanos entre si de forma más clara. Con esta implementación se cumplió el primer requerimiento de los usuarios. Adicionalmente se agregó un botón en la plataforma que descarga el archivo .nwk resultado del proceso de inferencia, cumpliendo con el segundo requerimiento. Esto permitirá que los usuarios puedan utilizar el archivo para hacer análisis adicional en otras herramientas si así lo requieren.

Para el requerimiento tres y cuatro, se crearon pestañas en la herramienta que permi-

ten visualizar las secuencias alineadas y concatenadas utilizando pyMSaViz (Nakatsu *et al.*, 2024). Esta representación de la alineación se presenta como una imagen estática, descargable que muestra los concensos de alineación y colorea los nucleótidos que se alinean al concenso. El visualizador permite al usuario descargar el archivo fasta de la alineación y también permite descargar la imagen generada. Se optó por utilizar esta herramienta ya que está ampliamente documentada y se utiliza en publicaciones académicas para mostrar alineaciones (Gomaz y Štefanić, 2024). Se provee el archivo de alineación y concatenación para usuarios que deseen modificar la visualización agregando anotaciones y otras capacidades de MSAViz. Si el usuario reemplaza la imagen generada en el directorio de salida puede tener un reporte con más detalle en la alineación. Esto puede ser de especial utilidad para anotar la alineación y mostrar zonas de interés.

Para la descarga de logs, se agregó una pestaña adicional que permite visualizar los archivos TXT de cada log y descargarlos fácilmente. Se optó por incluir un visualizador para facilitar a los usuarios el proceso de análisis de logs dentro de la herramienta. Este visualizador es rudimentario y ocupa solamente la impresión de los archivos de texto en pantalla, pero una posible mejora para el futuro sería desarrollar una sintaxis estandarizada de logs para poder agregar formatteo de código al visualizador del informe. Esto sobrepasa el alcance de esta investigación, ya que la inclusión del visualizador va más allá de las funcionalidades requeridas. Con la inclusión de los botones de descarga, se cumple lo solicitado por las personas a las que se les prototipó la herramienta. El desarrollo de la herramienta de reporte cumple con el objetivo específico 1: *Implementar una herramienta de reporte para la visualización interactiva y exploración de datos multi-locus.* permitiendo a los usuarios visualizar el dendrograma de forma interactiva a través del gráfico desarrollado en Plotly y la exploración de los resultados utilizando el visualizador de la alineación de secuencias y *logs* de la herramienta.

7.3. Validación de la herramienta

Para la validación de la herramienta se descargaron todas las secuencias utilizadas por el estudio *Multilocus Sequence Analysis, a Rapid and Accurate Tool for Taxonomic Classification, Evolutionary Relationship Determination, and Population Biology Studies of the Genus Shewanella*(Fang, Wang, Liu, Dai, Cai, Li *et al.*, 2019b). Utilizando la matriz de acceso disponible como anexo de la herramienta. Dicha matriz contenía los códigos de acceso en genbank para cada una de las secuencias utilizadas en el estudio. El estudio original evalúa 62 organismos, para este estudio se seleccionó solamente las primeras 15 secuencias del dendrograma resultante del estudio original. Estas se pueden observar en la Figura 8.

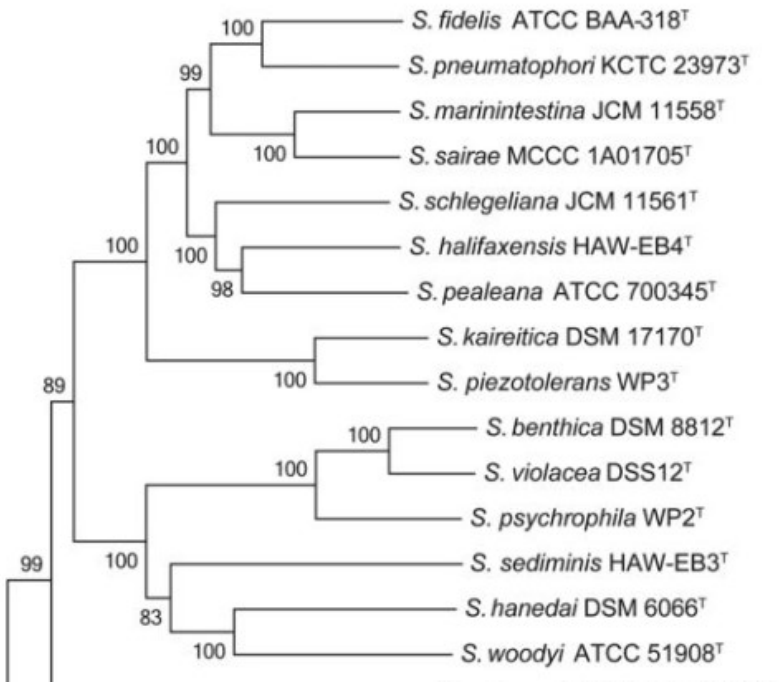

Figura 8: Dendrograma resultado del artículo *Multilocus Sequence Analysis, a Rapid and Accurate Tool for Taxonomic Classification, Evolutionary Relationship Determination, and Population Biology Studies of the Genus Shewanella*(Fang, Wang, Liu, Dai, Cai, Li *et al.*, 2019b)

Las secuencias se buscaron y se descargaron de forma manual desde Genbank. Se creó un archivo ***input.xlsx*** correspondiente a las secuencias descargadas y se procedió a ejecutar el *pipeline* utilizando dos combinaciones de herramientas distintas. Para elegir estas combinaciones se optó por ejecutarlas con la configuración por defecto y eligiendo dos categorias de configuración. La primera correspondió a un análisis rápido, utilizando en cada paso los algoritmos menos costosos en términos de recursos, como se estableció anteriormente en esta discusión esto implicó que para los distintos pasos se eligieron las siguientes herramientas:

1. Alineación: *MUSCLE Align*
2. Concatenación: *Seqkit*
3. Inferencia: *DendroPy Maximum Likelyhood tree (MLT)*

Los resultados de esta primera ejecución devolvieron un dendrograma que se resultó como el que se muestra en la Figura 9. En este se pudo observar que la mayoría de organismos quedaron en la misma posición que el diagrama del artículo de referencia. Ninguno de los punteos es igual a la matriz del artículo de referencia.

Como diferencias se pudo observar que en primer lugar, *S. marinintestina* se ha agrupado junto con *S. pneumatophori*, mientras que en el árbol original estaba agrupado directamente con otras especies en una posición distinta. Esta nueva ubicación introduce una variación en el orden de ramificación.

Además, *S. pealeana* y *S. kaireitica* se han ubicado en una posición ligeramente diferente,

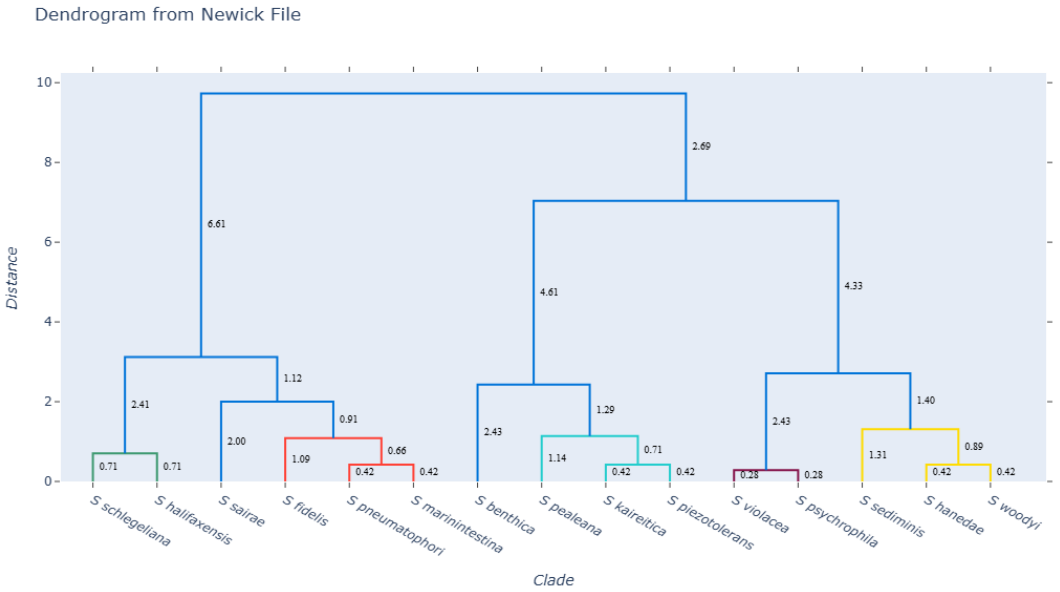

Figura 9: Resultados de la ejecución de análisis multi locus utilizando MUSCLE, Seqkit y DendroPy con configuración por defecto

manteniendo su relación cercana pero alterando la secuencia en la estructura del árbol. En el árbol original, estas especies se encuentran en una relación más directa con otras especies, mientras que aquí ha cambiado su nivel de ramificación.

La agrupación de *S. benthica* también se ha modificado ligeramente, cambiando su posición en relación con *S. sediminis* y otras especies cercanas. En la versión modificada, *S. benthica* aparece más cercana a *S. pealeana* y *S. kaireitica*.

Por último, la estructura de ramificación que incluye a *S. violacea*, *S. psychrophila*, *S. sediminis*, *S. hanedae*, y *S. woodyi* ha sido alterada sutilmente en el nuevo archivo Newick, cambiando ligeramente el orden en que aparecen y sus niveles de ramificación.

Estos cambios mantienen un 80 % de similitud con la estructura original, ya que a pesar de que algunas de las ramas aparecen en otra posición. En general, la mayoría de las especies se ubicaron en la misma rama en la que ya se encontraban originalmente. Esto puede atribuirsele principalmente al algoritmo de inferencia utilizado.

Una de las principales limitaciones de este método es su alta sensibilidad a la calidad de los datos y a las secuencias utilizadas. Incluso pequeñas variaciones en la alineación de las secuencias o en la selección de modelos evolutivos pueden influir significativamente en la estructura del árbol generado. Otra de las limitantes es la tendencia a convergencia hacia óptimos locales en lugar de una solución global, lo que puede distorsionar las relaciones filogenéticas y producir agrupaciones que no reflejan fielmente las relaciones evolutivas reales entre las especies (Felsenstein, 2004). Este último factor podría explicar por qué algunas de las especies quedaron en otros grupos.

Para la segunda ejecución se optó por realizar un análisis utilizando todas las herramientas que requieren más recursos para los distintos pasos. Sin embargo, se trabajó con la

configuración por defecto. Las herramientas utilizadas fueron:

1. Alineación: *ClustalW*
2. Concatenación: *AMAS*
3. Inferencia: *MRBayes MCMC*

Como se puede observar en la Figura 10 Las posiciones del dendrograma resultante utilizando esta configuración son las mismas que las resultantes en el diagrama del artículo de referencia. Sin embargo, los punteos asignados a cada rama no son iguales. Esto se puede explicar por la naturaleza no determinística del algoritmo MCMC de MRBayes. Un análisis con MRBayes podría asignar diferentes pesos en la matriz filogenética debido a su naturaleza estocástica, a diferencia de los métodos deterministas. MRBayes utiliza un enfoque bayesiano para la inferencia de árboles, donde se basa en un muestreo de Monte Carlo por cadenas de Markov (MCMC) para explorar el espacio de posibles árboles y modelos evolutivos. Esto significa que el algoritmo no sigue un único camino fijo hacia una solución, sino que genera múltiples árboles posibles basándose en distribuciones de probabilidad, permitiendo evaluar la incertidumbre y variabilidad en las relaciones filogenéticas (Huelsenbeck y Ronquist, 2001).

Debido a este enfoque, los pesos o probabilidades asignados a diferentes ramas en la matriz filogenética pueden variar entre ejecuciones, incluso con los mismos datos, ya que el proceso MCMC explora distintas configuraciones evolutivas en función de la probabilidad a posteriori. Esto contrasta con los métodos deterministas, como la máxima verosimilitud, que proporcionan un solo árbol óptimo basado en una única solución. La estocasticidad de MRBayes permite incorporar incertidumbre y obtener una visión más completa de la variabilidad en la estructura del árbol (Ronquist y Huelsenbeck, 2003).

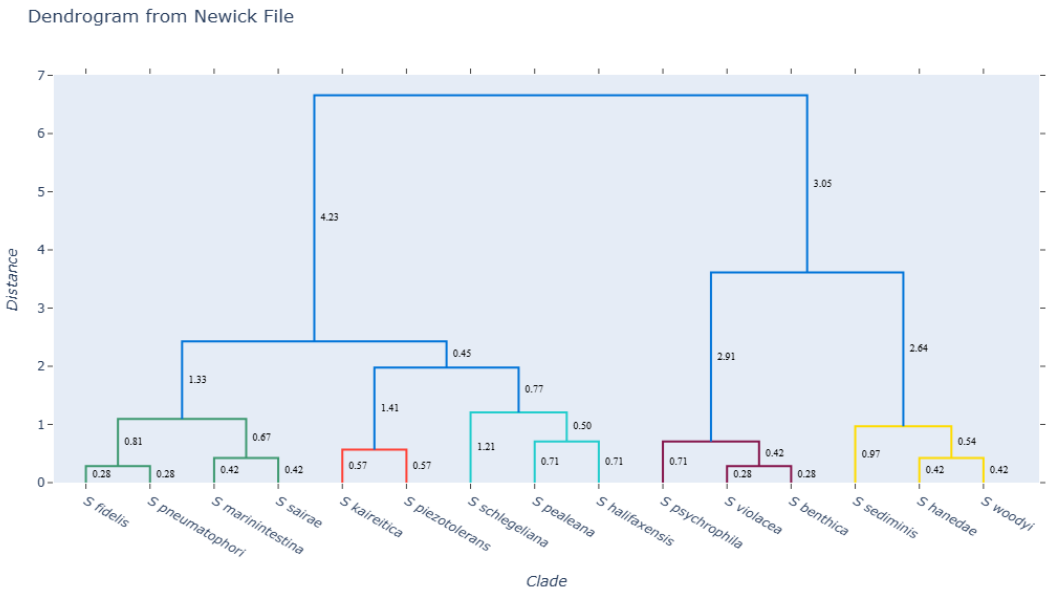

Figura 10: Resultados de la ejecución de análisis multi locus utilizando ClustalW, AMAS y MRBayes con configuración por defecto

Estas validaciones de la herramienta cumplen con el objetivo especifico 3 en donde se evalúa la herramienta comparandola con un estudio con bases de datos disponibles en internet. El *pipeline* desarrollado en Python *Snakemake* que implementa distintas tecnologías para alineación, concatenación, inferencia filogenética y tiene un visualizador se desarrolló exitosamente y puede ser consultado en el repositorio de github del proyecto: https://github.com/Rafalp190/MLSA-pipeline. El repositorio también contiene instrucciones para la descarga, instalación y uso de la herramienta.

CAPÍTULO 8

Conclusiones

- Se desarrollo un *pipeline* para analizar datos multi-locus en análisis filogenético. Permitiendo a los usuarios seleccionar entre distintas herramientas y parámetros para obtener análisis filogenéticos de la calidad que requieran para su análisis.
- Se creó una herramienta lo suficientemente flexible para que los usuarios puedan alinear secuencias de ADN utilizando MUSCLE y ClustalW de forma intercambiable. El *pipeline* también permite seleccionar herramientas distintas para los pasos de concatenación y análisis filogenético.
- Se creó una herramienta de visualización utilizando *Shiny Python* que permite interactuar con un dendrograma, analizar los resultados de la alineación de secuencias y descargar los archivos de resultados de los distintos pasos del proceso para utilizarlos en otros análisis.
- Se validó la herramienta replicando un estudio de análisis filogenético multilocus. Se determinó que los resultados utilizando MUSCLE, SeqKit y DendroPy se parece en un 80 % al resultado obtenido en el estudio original, mientras que el uso de ClustalW, AMAS y MRBayes obtuvo las mismas categorizaciones que el estudio original, difiriendo únicamente en los punteos asignados a cada rama del árbol.
- La utilización de la herramienta *Docker* para la creación del ambiente de desarrollo agilizó el proeso de implementación y redujo la cantidad de procesos de instalación que los usuarios deben realizar para que la herramienta funcione en sus computadores.

CAPÍTULO 9

Recomendaciones

Esta herramienta es un primer acercamiento al desarrollo de un *pipeline* bioinformático multi locus para análisis filogenético, por lo que se limitó la cantidad de tecnologías que se pueden elegir en cada uno de los pasos. Se recomienda para futuras investigaciones en este ámbito que intenten implementar otras herramientas o que sean más pedagógicos para los usuarios en la selección de parámetros para cada herramienta. Adicionalmente, se recomienda que se incluya mediación pedagógica adicional a los mensajes de error para que los usuarios tengan una mejor capacidad de resolverlos sin tener que acudir a fuentes secundarias de información. Esta implementación no parametriza los errores de las distintas herramientas ni homogeniza como esos errores se le transmiten al usuario final, por lo que puede existir una dificultad de interpretación si el usuario no tiene al menos un poco de experiencia en las herramientas que eligió para cada paso de la ejecución.

Se recomienda fuertemente utilizar docker para la distribución de *pipelines* bioinformáticos complejos. Esto permite flexibilidad de implementación en distintos sistemas y con las funcionalidades más modernas como Kubernetes y ambientes paralelizados en Docker, podría dar lugar al desarrollo de *pipelines* paralelizables que pueden ser implementadas directamente en servidores tipo cluster o en infraestructuras en la nube.

CAPÍTULO 10

Bibliografía

Agarwala, R., Barrett, T., Beck, J., Benson, D. A., Bollin, C., Bolton, E., Bourexis, D., Brister, J. R., Bryant, S. H., Canese, K., Cavanaugh, M., Charowhas, C., Clark, K., Dondoshansky, I., Feolo, M., Fitzpatrick, L., Funk, K., Geer, L. Y., Gorelenkov, V., ... Zbicz, K. (2018). Database resources of the National Center for Biotechnology Information. *Nucleic Acids Research*, *46*, D8-D13. https://doi.org/10.1093/nar/gkx1095

Ashkenazy, H., Sela, I., Levy Karin, E., Landan, G., & Pupko, T. (2018). Multiple Sequence Alignment Averaging Improves Phylogeny Reconstruction. *Systematic Biology*, *68*(1), 117-130. https://doi.org/10.1093/sysbio/syy036

Attwood, S. W., Hill, S. C., Aanensen, D. M., Connor, T. R., & Pybus, O. G. (2022, septiembre). Phylogenetic and phylodynamic approaches to understanding and combating the early SARS-CoV-2 pandemic. https://doi.org/10.1038/s41576-022-00483-8

Baldauf, S. L. (2003, junio). Phylogeny for the faint of heart: A tutorial. https://doi.org/10.1016/S0168-9525(03)00112-4

Bell, C. G. (2011). Accessing and Selecting Genetic Markers from Available Resources. En *Genetic Markers in Epidemiology* (pp. 1-17). Springer. https://doi.org/10.1007/978-1-61779-176-5_1

Borowiec, M. L. (2016). AMAS: a fast tool for alignment manipulation and computing of summary statistics. *PeerJ*, *4*, e1660. https://doi.org/10.7717/peerj.1660

Caporaso, J. G., Bittinger, K., Bushman, F. D., DeSantis, T. Z., Andersen, G. L., & Knight, R. (2010). PyNAST: a flexible tool for aligning sequences to a template alignment. *Bioinformatics*, *26*(2), 266-267. https://doi.org/10.1093/bioinformatics/btp636

Cashman, M., Cohen, M. B., Marsh, A. L., & Cottingham, R. W. (2023). Dissecting Complexity: The Hidden Impact of Application Parameters on Bioinformatics Research. *bioRxiv*. https://doi.org/10.1101/2022.12.20.521257

Cock, P. J. A., Antao, T., Chang, J. T., Chapman, B. A., Cox, C. J., Dalke, A., Friedberg, I., Hamelryck, T., Kauff, F., Wilczynski, B., & de Hoon, M. J. L. (2009). Biopython: freely available Python tools for computational molecular biology and bioinformatics. *Bioinformatics (Oxford, England)*, *25*(11), 1422-1423. https://doi.org/10.1093/bioinformatics/btp163

Cock, P. J., Fields, C. J., Goto, N., Heuer, M. L., & Rice, P. M. (2009). The Sanger FASTQ file format for sequences with quality scores, and the Solexa/Illumina FASTQ variants. *Nucleic Acids Research*, *38*, 1767-1771. https://doi.org/10.1093/nar/gkp1137

Comunicación, G. (2023, abril). *Secuencia genética en Guatemala: Avances y retos*. https://www.galileo.edu/secuencia-genetica-en-guatemala

CONAP. (2014). V INFORME NACIONAL DE CUMPLIMIENTO A LOS ACUERDOS DEL CONVENIO SOBRE LA DIVERSIDAD BIOLÓGICA.

Docker, Inc. (2024). *Docker Desktop for Windows Installation Guide* [Accessed: 2024-10-29]. https://docs.docker.com/desktop/install/windows-install/

Dupuis, J. R., Roe, A. D., & Sperling, F. A. H. (2012). Multi-locus species delimitation in closely related animals and fungi: One marker is not enough. *Molecular Ecology*, *21*(18), 4422-4436. https://doi.org/10.1111/j.1365-294X.2012.05642.x

Edgar, R. C. (2022). High-accuracy alignment ensembles enable unbiased assessments of sequence homology and phylogeny. https://doi.org/10.1101/2021.06.20.449169

Emery, L. (2015). Phylogenetics: An introduction. https://doi.org/10.6019/tol.phyl.2015.00001.1

Fang, Y., Wang, Y., Liu, Z., Dai, H., Cai, H., Li, Z., *et al.* (2019a). Multilocus Sequence Analysis, a Rapid and Accurate Tool for Taxonomic Classification, Evolutionary Relationship Determination, and Population Biology Studies of the Genus Shewanella. *Applied and Environmental Microbiology*, *85*(11). https://doi.org/10.1128/AEM.03126-18

Fang, Y., Wang, Y., Liu, Z., Dai, H., Cai, H., Li, Z., Du, Z., Wang, X., Jing, H., Wei, Q., Kan, B., Wang, D., Fang, C. Y., & Nojiri, E. H. (2019b). Multilocus Sequence Analysis, a Rapid and Accurate Tool for Taxonomic Classification, Evolutionary Relationship Determination, and Population Biology Studies of the Genus Shewanella. https://doi.org/10.1128/AEM

Felsenstein, J. (2004). *Inferring Phylogenies*. Sinauer Associates.

Foox, J., Tighe, S. W., Nicolet, C. M., *et al.* (2021). Performance assessment of DNA sequencing platforms in the ABRF Next-Generation Sequencing Study. *Nature Biotechnology*, *39*(9), 1129-1140. https://doi.org/10.1038/s41587-021-01049-5

Gomaz, B., & Štefanić, Z. (2024). Oligomeric Symmetry of Purine Nucleoside Phosphorylases. *Symmetry*, *16*(1). https://doi.org/10.3390/sym16010124

Hakeem, K. R., Shaik, N. A., Banaganapalli, B., & Elango, R. (2019, enero). *Essentials of bioinformatics, volume III: In silico life sciences: Agriculture*. Springer International Publishing. https://doi.org/10.1007/978-3-030-19318-8

Hall, B. G. (2004). Comparison of the Accuracies of Several Phylogenetic Methods Using Protein and DNA Sequences. *Molecular Biology and Evolution*, *22*(3), 792-802. https://doi.org/10.1093/molbev/msi066

Huang, S., & Li, B. (2024). Common Methods for Phylogenetic Tree Construction and Their Implementation in R. *Bioengineering*, *11*(5), 480. https://doi.org/10.3390/bioengineering11050480

Huelsenbeck, J. P., & Ronquist, F. (2001). MRBAYES: Bayesian inference of phylogenetic trees. *Bioinformatics*, *17*(8), 754-755.

Inter-American Development Bank. (2024). *Borrowing Member Countries* [Accessed: 2024-10-29]. https://www.iadb.org/en/who-we-are/how-we-are-organized/borrowing-member-countries

Johnson, M., Zaretskaya, I., Raytselis, Y., Merezhuk, Y., McGinnis, S., & Madden, T. L. (2008). NCBI BLAST: a better web interface. *Nucleic acids research*, *36*. https://doi.org/10.1093/nar/gkn201

Kapitsaki, G. M., Tselikas, N. D., & Foukarakis, I. E. (2015). An insight into license tools for open source software systems. *Journal of Systems and Software*, *102*, 72-87. https://doi.org/https://doi.org/10.1016/j.jss.2014.12.050

Karsch-Mizrachi, I., Takagi, T., & Cochrane, G. (2018). The international nucleotide sequence database collaboration. *Nucleic Acids Research*, *46*, D48-D51. https://doi.org/10.1093/nar/gkx1097

Kasprzak, P., Mitchell, L., Kravchuk, O., & Timmins, A. (2021). Six Years of Shiny in Research - Collaborative Development of Web Tools in R. *The R Journal*, *12*(2), 155-162. https://doi.org/10.32614/RJ-2021-004

Kim, N., Hahn, M., & Lee, S.-J. (2016). Bayesian and Maximum Likelihood Phylogenetic Analyses of Protein Sequence Data. *BMC Ecology and Evolution*, *16*(1), 10-23. https://doi.org/10.1186/s12862-016-0684-2

Köster, J., & Rahmann, S. (2012). Snakemake—a scalable bioinformatics workflow engine. *Bioinformatics*, *28*(19), 2520-2522. https://doi.org/10.1093/bioinformatics/bts480

Kück, P., & Longo, G. C. (2014). FASconCAT-G: extensive functions for multiple sequence alignment preparations concerning phylogenetic studies. *Molecular Phylogenetics and Evolution*, *60*(2), 472-476. https://doi.org/10.1016/j.ympev.2013.11.014

Lemmon, E. M., & Lemmon, A. R. (2013). High-throughput genomic data in systematics and phylogenetics. https://doi.org/10.1146/annurev-ecolsys-110512-135822

Lewis, P. O., Holder, M. T., & Swofford, D. L. (2015). Phycas: Software for Bayesian Phylogenetic Analysis. *Systematic Biology*, *64*(3), 525-531. https://doi.org/10.1093/sysbio/syu132

Loecker, J., & Ewing, P. (2021). Benefits of the Snakemake Workflow Management Software in Comparison to Traditional Programming [South Dakota State University Honors Capstone Projects. Accessed: 2024-10-29]. https://openprairie.sdstate.edu/honors_isp/8/

Madeira, F., Madhusoodanan, N., Lee, J., Eusebi, A., Niewielska, A., Tivey, A. R. N., Lopez, R., & Butcher, S. (2024). The EMBL-EBI Job Dispatcher sequence analysis tools framework in 2024. *Nucleic Acids Research*. https://doi.org/10.1093/nar/gkae241

Maglott, D., Ostell, J., Pruitt, K. D., & Tatusova, T. (2011a). Entrez Gene: gene-centered information at NCBI. *Nucleic Acids Research*, *39*(Database), D52-D57. https://doi.org/10.1093/nar/gkq1237

Maglott, D., Ostell, J., Pruitt, K. D., & Tatusova, T. (2011b). Entrez gene: Gene-centered information at NCBI. *Nucleic Acids Research*, *39*. https://doi.org/10.1093/nar/gkq1237

Mccauley, R., Fitzgerald, S., Lewandowski, G., Murphy, L., Simon, B., Thomas, L., & Zander, C. (2008). Debugging: A review of the literature from an educational perspective. *Computer Science Education*, *18*. https://doi.org/10.1080/08993400802114581

Microsoft Corporation. (2024). *Windows Subsystem for Linux Documentation* [Accessed: 2024-10-29]. https://learn.microsoft.com/en-us/windows/wsl/

Moreno, M. A., Dolson, E., & Ofria, C. (2022). hstrat: a Python Package for phylogenetic inference on distributed digital evolution populations. *Journal of Open Source Software*, *7*(80), 4866. https://doi.org/10.21105/joss.04866

Nakatsu, K., Jijiwa, M., Khadka, V., Nasu, M., & Deng, Y. (2024). sRNAfrag: a pipeline and suite of tools to analyze fragmentation in small RNA sequencing data. *Briefings in Bioinformatics*, *25*(1). https://doi.org/10.1093/bib/bbad515

O'Halloran, D. (2014). A practical guide to phylogenetics for nonexperts. *Journal of Visualized Experiments*. https://doi.org/10.3791/50975

Pearson, W. R., & Lipmant, D. J. (1988). Improved tools for biological sequence comparison (amino acid/nucleic acid/data base searches/local similarity). https://www.pnas.org

Rannala, B., & Yang, Z. (2008). Phylogenetic inference using whole genomes. https://doi.org/10.1146/annurev.genom.9.081307.164407

Ranwez, V., & Chantret, N. N. (2020). Strengths and Limits of Multiple Sequence Alignment and Filtering Methods. https://hal.inria.fr/PGE.

Reuter, J. A., Spacek, D. V., & Snyder, M. P. (2015, mayo). High-Throughput Sequencing Technologies. https://doi.org/10.1016/j.molcel.2015.05.004

Rodréguez-Ezpeleta, N., Brinkmann, H., Roure, B., Lartillot, N., Lang, B. F., & Philippe, H. (2007). Detecting and overcoming systematic Errors in genome-scale phylogenies. *Systematic Biology*, *56*, 389-399. https://doi.org/10.1080/10635150701397643

Ronquist, F., & Huelsenbeck, J. P. (2003). MrBayes 3: Bayesian phylogenetic inference under mixed models. *Bioinformatics*, *19*(12), 1572-1574.

Ronquist, F., van der Mark, P., & Huelsenbeck, J. P. (2012). *Bayesian phylogenetic analysis using MrBayes* (A.-M. V. Philippe Lemey Marco Salemi, Ed.). Cambridge University Press. https://doi.org/10.1017/CBO9780511819049.009

Roy, S. R. A. (2014). Molecular Markers in Phylogenetic Studies-A Review. *Journal of Phylogenetics & Evolutionary Biology*, *02*. https://doi.org/10.4172/2329-9002.1000131

Santos, L., Alves, A., & Alves, R. (2017). Evaluating multi-locus phylogenies for species boundaries determination in the genus Diaporthe. *PeerJ*, *5*, e3120. https://doi.org/10.7717/peerj.3120

Schneider, K., Venn, B., & Mühlhaus, T. (2024). Plotly.NET: A fully featured charting library for .NET programming languages [[version 2; peer review: 1 approved, 1 approved with reservations]]. *F1000Research*, *11*, 1094. https://doi.org/10.12688/f1000research.123971.2

Shen, W., Le, S., Li, Y., & Hu, F. (2016). SeqKit: A cross-platform and ultrafast toolkit for FASTA/Q file manipulation. *PloS one*, *11*(10), e0163962. https://doi.org/10.1371/journal.pone.0163962

Sievers, F., & Higgins, D. G. (2014). Clustal Omega. *Current Protocols in Bioinformatics*, *2014*, 3.13.1-3.13.16. https://doi.org/10.1002/0471250953.bi0313s48

Sievers, F., & Higgins, D. G. (2018). Clustal Omega for making accurate alignments of many protein sequences. *Protein Science*, *27*, 135-145. https://doi.org/10.1002/pro.3290

Steenwyk, J. L., Buida, T. J., Li, Y., Shen, X.-X., & Rokas, A. (2020). ClipKIT: A multiple sequence alignment trimming software for accurate phylogenomic inference. *PLOS Biology*, *18*(12), e3001007. https://doi.org/10.1371/journal.pbio.3001007

Strickberger, M. W. (2005). *Evolution*. Jones & Bartlett Learning.

Walker, J. M. (2021). Methods In Mlecular Biology. http://www.springer.com/series/7651

Warris, S., Timal, N. R. N., Kempenaar, M., *et al.* (2018). pyPaSWAS: Python-based multi-core CPU and GPU sequence alignment. *PLOS ONE*, *13*(1), e0190279. https://doi.org/10.1371/journal.pone.0190279

Yang, Z., & Rannala, B. (2012, mayo). Molecular phylogenetics: Principles and practice. https://doi.org/10.1038/nrg3186

CAPÍTULO 11

Anexos

11.1. Anexo 1: Instrumento de primera validación de prototipo

Preguntas para Potenciales Usuarios

Características y Capacidades

- ¿Qué características o capacidades considerarías esenciales en un pipeline de MLSA?
- ¿Hay alguna característica avanzada (por ejemplo, alineamiento automatizado, construcción de árboles filogenéticos) que te gustaría ver incluida?

Manejo de Datos

- ¿Cómo se deberían descargar los datos?

Integración y Compatibilidad

- ¿Con qué formatos de archivo (por ejemplo, FASTA, NEXUS, PhyloXML) trabajas comúnmente, y debería el pipeline soportar estos formatos?

Personalización y Flexibilidad

- ¿Cuánta personalización y flexibilidad necesitas en el pipeline?

Interfaz de Usuario

- ¿Prefieres una interfaz de línea de comandos, una interfaz gráfica de usuario, o ambas para el pipeline de MLSA?
- ¿Qué tan importante es para ti la facilidad de uso y la documentación?

Rendimiento y Velocidad

- ¿Qué tan crítico es la velocidad del análisis para tu investigación?
- ¿Estás dispuesto a sacrificar algo de precisión por resultados más rápidos, o es la precisión tu máxima prioridad?

Colaboración y Compartir

- ¿Serían valiosas para ti las características que faciliten el compartir y el trabajo colaborativo?

Comentarios recopilados de entrevistas

Para la herramienta, las entrevistas con profesionales de la biología y bioinformática parecen indicar que la herramienta tiene utilidad y es de interés para personas que trabajan en taxonomía. Algunos comentarios que se obtuvieron recalcan los siguientes puntos:

- La herramienta no debe limpiar las secuencias, ya que el proceso de limpieza es muy diferente para cada secuencia y debería hacerse como un proceso aparte.
- La descarga de reportes de progreso es sumamente importante.
- Solo debe haber un momento de configuración en donde se parametrizan todos los pasos.
- La instalación debe ser lo más sencilla posible.
- Los formatos de entrada deben ser compatibles con salidas usuales de secuenciadores (Formatos FASTA, FASTQ, FASTQC).
- Los formatos de salida deben ser compatibles con software de edición y modificación de árboles filogenéticos.
- La herramienta se beneficiaría de una herramienta de procesamiento rápido para corridas preliminares.
- La documentación debe ser clara en los algoritmos que se utilizan, pero no debe tener absolutamente toda la herramienta documentada. Es suficiente tener un enlace a la documentación oficial de los distintos algoritmos.

11.2. Anexo 2: Dockerfile de la herramienta

Bloque de código 11.1: Archivo Dockerfile de la herramienta

```
# Use Ubuntu 20.04 as the base image
FROM ubuntu:20.04

# Prevent interactive prompts during package installation
ENV DEBIAN_FRONTEND=noninteractive

# Update package lists and install system dependencies
RUN apt-get update && apt-get install -y \
    python3 \
    python3-pip \
    git \
    curl \
    build-essential \
    make \
    wget \
    clustalw \
    muscle \
    libopenmpi-dev \
    && rm -rf /var/lib/apt/lists/*

# Install SeqKit (for concatenation)
RUN wget https://github.com/shenwei356/seqkit/
releases/download/v2.3.0/seqkit_linux_amd64.tar.gz \
    && tar -zxvf seqkit_linux_amd64.tar.gz \
    && chmod +x seqkit \
    && mv seqkit /usr/local/bin/seqkit \
    && rm seqkit_linux_amd64.tar.gz

# Install MrBayes (for Bayesian phylogenetic inference)
RUN wget https://github.com/NBISweden/MrBayes/
archive/v3.2.7a.tar.gz \
    && tar -xvzf v3.2.7a.tar.gz \
    && cd MrBayes-3.2.7a \
    && ./configure --enable-mpi \
    && make \
    && make install \
    && cd .. \
    && rm -rf MrBayes-3.2.7a v3.2.7a.tar.gz

# Install Python packages: BioPhylo, DendroPy, and AMAS
COPY requirements.txt /app/requirements.txt
COPY Snakefile /app/Snakefile

RUN pip3 install --no-cache-dir -r /app/requirements.txt
```

```
RUN pip3 install --no-cache-dir snakemake
# Copy the src directory into the container
# (contains the Python scripts)
COPY src/ /app/src/

# Set the working directory to /app
WORKDIR /app

# Specify the entry point for Snakemake
CMD ["sh", "-c", "snakemake --snakefile Snakefile --cores ${CORES:-1}"]
```