Universidad de la República
PEDECIBA - Biología - Área Genética

Institut Pasteur de Montevideo

Tesis de Doctorado

Extendiendo el concepto de pangenoma a taxones superiores mediante la implementación de nuevas herramientas bioinformáticas

Ignacio Ferrés Cáceres

Tutor
Dr. Pablo Fresia

Co-tutor
Dr. Gregorio Iraola

14 de julio de 2026

El trabajo que se describe en este documento fue realizado en el Laboratorio de Genómica Microbianas del Institut Pasteur de Montevideo, entre 2018 y 2023, bajo la tutoría de los doctores Pablo Fresia y Gregorio Iraola. La defensa se realizó el 29 de Junio de 2026 en el Salón de Actos «Guillermo Dighiero» del Institut Pasteur de Montevideo. El tribunal estuvo conformado por los doctores Pablo Smircich (presidente), María Inés Fariello, y Federico Rosconi.

Esta tesis fue financiada por la Agencia Nacional de Investigación e Innovación (ANII) a través de una beca de Doctorado, código POS_NAC_2018_1_151494. El doctorado se enmarca en el PEDECIBA - Programa de Desarrollo de las Ciencias Básicas, de la Universidad de la República.

«Omnis cellula e cellula»

— ***Rudolf Ludwig Carl Virchow***

Poder decir adiós
es crecer.

— ***Gustavo Cerati***

Motomami motomami.

— ***La Rosalía***

Agradecimientos

A mi familia y mis amigos, por estar siempre.

A los compañeros y compañeras del LGM, por bancar cabeza y motivarme.

A mis tutores, Pablo y Gregorio, por darme el lugar y la financiación para desarrollar mis investigaciones, y principalmente por darme la libertad para apropiarme de un tema de investigación nuevo para el grupo.

Al tribunal, por los valiosos aportes.

A la IA de Google y a mi psicólogo, por ayudarme a desarmar el trancazo que tuve durante unos años desde que terminé el trabajo y defendí la tesis.

A todos quienes fueron parte de este camino.

Gracias.

Prólogo

Esta tesis de doctorado fue desarrollada enteramente en el Institut Pasteur de Montevideo, bajo la tutela de los doctores Pablo Fresia y Gregorio Iraola. Los primeros bocetos de este trabajo se desarrollaron en la Unidad de Bioinformática durante mi maestría, dirigida por el Dr. Hugo Naya y por el Dr. Iraola, entre 2015 y 2018. Posteriormente se conformó el Laboratorio de Genómica Microbiana, donde realicé la mayor parte del trabajo. A partir del segundo año de la pandemia de COVID-19, el grupo pasa a formar parte del Centro de Innovación en Vigilancia Epidemiológica (CiVE) del Instituto Pasteur de Montevideo.

Para la introducción decidí hacer una revisión bibliográfica relativamente exhaustiva sobre los temas que de alguna forma tienen que ver con esta tesis, aunque muchas secciones pueden ser consideradas como laterales ya que hacen a la historia del desarrollo del concepto de pangenoma. De todas formas, pienso que estas secciones hacen más liviana la lectura y permiten comprender mejor algunas ideas.

También decidí dejar algunos anglicismos para palabras que pierden un poco su significado si se traducen al castellano, o que no encontré buenas sinónimas, y para no volver muy repetitivo el texto me fui un poco de lo canónico. Por ejemplo, para la palabra «grupo» podemos usar *cluster*, o *subset*, en algunos contextos en los que se entiende mejor de qué estamos hablando en lugar de usar «grupo» muchas veces. Tal vez el significado que les demos sean muy del día a día de cada uno, pero me pareció que reflejaban mejor lo que quería decir en cada momento.

Las frases o citas que separan las secciones son meramente un toque personal. Algunas de alguna forma en mi mente refieren al capítulo que las preceden, o al que las suceden, otras son resultado del momento en que escribía esta tesis. No hay significados ocultos, salvo para mi.

Resumen

El pangenoma, desde el punto de vista bioinformático, es una estructura de datos que representa la presencia y ausencia (o abundancia) de familias génicas en una colección de genomas filogenéticamente relacionados. Se conceptualiza en base a la observación de que un sólo genoma de referencia no es suficiente para representar la variabilidad genómica de una especie bacteriana. Históricamente el concepto se ha aplicado a genomas pertenecientes a una misma especie ya que por razones metodológicas y de oportunidad científica, es donde las diferencias entre cepas se manifiestan de forma más evidente. El concepto, sin embargo, es intuitivamente extendible para relacionar organismos filogenéticamente más distantes. En esta tesis se explora esa posibilidad desarrollando tres bibliotecas escritas en lenguaje R para 1) simular pangenomas a distintos tiempos evolutivos, 2) reconstruir pangenomas distantes, y 3) analizar las reconstrucciones mediante una interfaz simple e intuitiva.

Abstract

The pangenome, from a bioinformatic perspective, is a data structure that represents the presence and absence (or abundance) of gene families in a collection of phylogenetically related genomes. It is conceptualized based on the observation that a single reference genome is not sufficient to represent the genomic variability of a bacterial species. Historically, the concept has been applied to genomes belonging to the same species because, due to methodological reasons and scientific opportunity, it is where the differences between strains manifest most evidently. The concept, however, is intuitively extendable to relate phylogenetically more distant organisms. This thesis explores that possibility by developing three libraries written in the R language to 1) simulate pangenomes at different evolutionary times, 2) reconstruct distant pangenomes, and 3) analyze the reconstructions through a simple and intuitive interface.

Índice general

Índice de figuras

Índice de cuadros

Lista de acrónimos

NGS tecnologías de secuenciación de nueva generación. 16

N_e tamaño poblacional efectivo. 20, 23, 70

HGT transferencia horizontal de genes. 22, 24

ADN Ácido desoxirribonucleico. 26

SIT Secuenciación por Inserción de Transposones. 29, 30

ARN Ácido ribonucleico. 29

GWAS Estudios de Asociación de Genoma Completo. 30, 31

SNP Polimorfismos Nucleotídicos Simples. 30, 31

Glosario

genoma accesorio Regiones genómicas presentes en algunas, pero no en todas, las cepas de determinada especie. 8, 16, 21, 22, 23, 24, 27

pangenoma Del inglés: *pangenome* ('pan' - $\pi\alpha\nu$: 'todo' en griego). Repertorio global de genes de una especie bacteriana. 8, 9, 11, 16, 17, 19, 20, 21, 22, 23, 24, 25, 26, 27, 30, 31, 32, 35, 36, 37, 38, 39, 41, 46, 47, 48, 59, 60, 61, 76, 80, 81

fluidez Del inglés: *fluidity*. Promedio del ratio de familias génicas únicas entre la suma de familias génicas en pares de genomas seleccionados al azar de los N genomas que conforman el pangenoma. 11, 19, 20, 21, 22, 26

genoma núcleo Del inglés: *coregenome* (*core*: núcleo). Regiones genómicas compartidas por todas las cepas de determinada especie. 16, 20, 21, 31

pangenoma abierto La secuenciación de nuevos genomas aporta nuevos genes a una tasa promedialmente constante. 18, 19, 20

pangenoma cerrado El número de nuevos genes con cada nueva secuenciación tiende a cero. 18, 19, 20

esencialoma Genes que son esenciales en al menos una cepa de la especie. 30

Capítulo 1

Introducción: El pangenoma bacteriano

Si bien se considera el inicio de la era genómica en 2003 a partir de la publicación del primer borrador del genoma humano completo[1,2], el primer genoma completo de un organismo vivo se publicó en 1995 y se trató del genoma de *Heamophilus influenzae*[3], dando comienzo a la era genómica microbiana. En el año 2000, dos trabajos pioneros de vacunología reversa en *Neisseria meningitidis* a partir de un único genoma disponible demostraron la utilidad de contar con la información genómica[4,5]. Un consorcio conformado por los investigadores de los trabajos citados en *Neisseria* se propuso volver a aplicar estas técnicas para abordar el problema de las infecciones neonatales del Grupo B de *Streptococcus*. En el año 2001 estos autores publican el primer genoma completo de *Streptococcus agalactiae*[6], y al año siguiente otro grupo independiente reportaba el genoma completo de otra cepa clínica de la misma especie[7]. Ya en este momento algunos indicios sugerían que existía alta variabilidad en el contenido genético de cepas de una misma especie bacteriana, si bien desde principios de los años noventa ya se observaba una discrepancia en organismos que tenían alta similitud de secuencia a nivel del gen de la subunidad 16S del ARNr, pero que en experimentos de hibridación ADN-ADN se observaba una distancia mucho mayor a la esperada[8]. En este punto también cabe mencionar un trabajo de 2002, por Welch y colaboradores, donde constataban que al comparar 3 cepas de *Escherichia coli*, sólo el 39,2 % del agrupamiento combinado de las proteínas eran comunes a las 3 cepas[9].

En el año 2005 Tettelin y colaboradores, quienes habían conformado el consorcio para secuenciar el primer genoma de *Streptococcus agalactiae* unos años antes, publican seis nuevos genomas de la misma especie y junto a los otros dos ya disponibles realizan un estudio comparativo[10]. Es en este trabajo donde se acuña el concepto de pangenoma como el repertorio global de genes de una especie bacteriana, y se distingue entre genoma núcleo de genoma accesorio o dispensable, es decir aquellos genes compartidos por todas las cepas, o aquellos presentes en algunas pero no en todas las cepas, respectivamente.

Casualmente, ese mismo año se comienzan a comercializar las tecnologías de secuenciación de nueva generación (NGS), lo cual catalizó una explosión exponencial en estudios comparativos con la perspectiva pangenómica que continúa hasta

la fecha, como se observa en la figura 1.1.

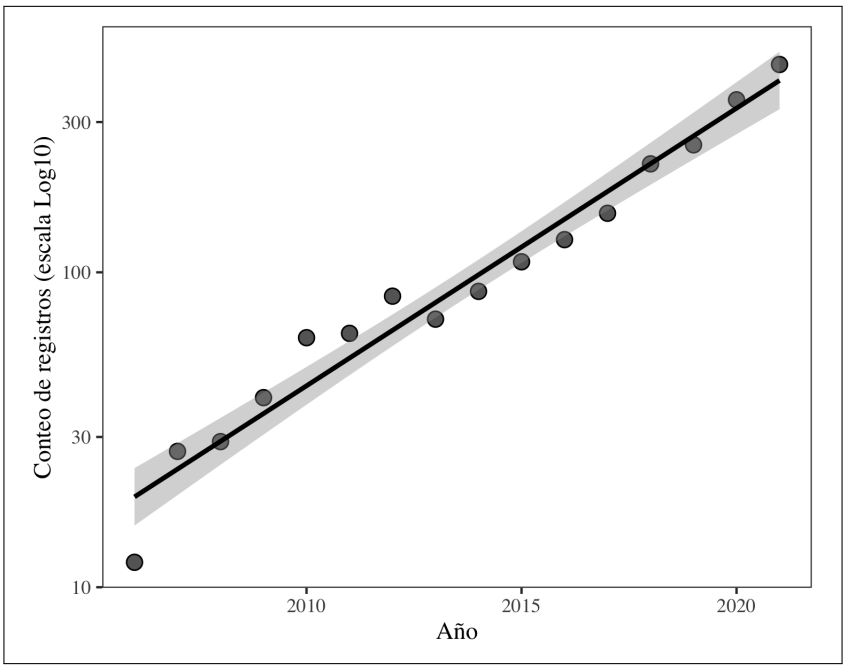


Figura 1.1: Conteo de registros bibliográficos con el término «pangenome» en PubMed desde 2005, en escala logarítmica. La recta tendencial sugiere crecimiento exponencial; el intevalo de confianza de 0,95 se muestra en gris.

El concepto de «pangenoma» guarda reminiscencias con el «progenote» de Woese y Fox de 1977[11], una entidad teórica en la cual la relación genotipo-fenotipo no se encontraba aún madura, en el sentido de que la información genética puede no compartir el orígen con el citoplasma que lo contiene. Tales «progenotes» serían entidades genéticas, pero no genómicas, ya que en este momento evolutivo no existiría la estabilidad necesaria para mantener de forma robusta un genoma completo. En un revisión posterior, Woese discute si en bacterias tiene sentido hablar de genealogía y taxonomía, o si debido a la transferencia horizontal de genes, se trata de quimeras genéticas con historias evolutivas independientes[12] (ver sección: «*Do bacteria have a genealogy and meaningful taxonomy?*», del artículo de 1987). Un concepto similar desarrolla Kendler en 1993, con sus pre-células («*pre-cells*»)[13,14], y Lawrence en 1999 en un capítulo sobre los genomas mínimos, conceptualiza la meta-célula («*meta-cells*») estudiando simulaciones de micelos que comparten un conjunto de genes cooperativos, pero donde ningún micelo es capaz de contener a todos los genes al mismo tiempo[15]. Podría decirse que, al menos en teoría, los pangenomas existen desde antes del establecimiento de genomas individuales, como un recurso compartido.

1.1. Descripción estadística de los pangenomas

Cuando este grupo de investigadores, liderado por Tettelin, se preguntó cuántos genomas distintos eran necesarios para capturar la diversidad genética presente en *S. agalactiae*, surgió la pregunta sobre cómo estimar el número potencial de genes presentes. Una asunción en esta pregunta fue no considerar microdiversidad, es decir, tomar a los distintos alelos de un gen como la misma entidad, o dicho de otro modo, permitir cierto número de polimorfismos al definir un gen. A esta escala la anterior asunción resulta muy conveniente y no merece la discusión sobre cuándo dos alelos dejan de ser el mismo gen. Más adelante en esta tesis se hablará sobre el concepto de ortología y esta discusión se volverá relevante al considerar parálogos y genomas más distantes.

En un primer análisis del número de nuevos genes en función del número de genomas muestreados, se ajustaron los parámetros a una función de decrecimiento exponencial:

$$F(N) = \kappa_c^{-N/\tau_c} + \tan(\theta) \tag{1.1}$$

donde $F(N)$ es el número de atributos distintos, en este caso nuevos genes; N es el número de entidades, o sea genomas; κ_c y τ_c son parámetros libres, y $\tan(\theta)$ mide el número de genes específicos para $N \to \infty$. Contrariamente a lo esperado, el parámetro $\tan(\theta)$ no se ajustaba a 0 sino a 33, lo cual implicaba que sin importar el esfuerzo de muestreo, en promedio apareceían unos 33 genes nuevos por cada genoma muestreado[10], haciendo tender a infinito este valor con dicho esfuerzo. Unos años después, con el aumento de genomas de *S. agalactiae* secuenciados, se observó que este comportamiento se ajustaba mejor a una Ley Potencial derivada de la ley de Heaps[16]:

$$F(N) = \kappa_c N^{-\alpha_c} \tag{1.2}$$

donde N y $F(N)$ representan los mismos parámetros que en la ecuación 1.1, y κ_c y α_c son constantes determinadas empíricamente. Con este nuevo marco teórico para describir la diversidad génica y nuevos genomas de *S. agalactiae*, se volvió al problema y se observó que el exponente seguía siendo menor que un valor crítico ($0 < \alpha_c < 1$) por lo que el número potencial de genes continuaba sin cota superior[17]. Ambos modelos, tanto decrecimiento exponencial como ley potencial, implicaban que estas bacterias podían incorporar genes de un suministro potencialmente infinito. Otras especies, sin embargo, mostraban una cota bien definida (Figura 1.2).

Todo lo anterior llevó a definir los conceptos de pangenoma abierto o pangenoma cerrado como propiedades biológicas intrínsecas de las especies bacterianas. Un pangenoma abierto es tal si en una muestra no sesgada de genomas de la especie,

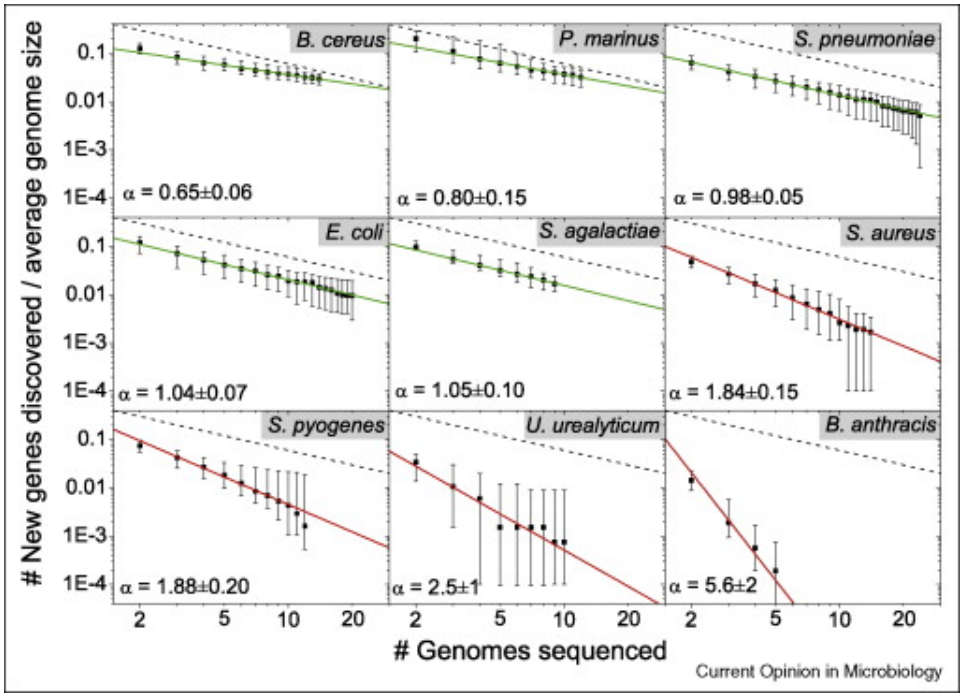


Figura 1.2: Ejemplos de pangenomas abiertos (verde) y cerrados (rojo). Reimpreso de *Current Opinion in Microbiology*, «Comparative genomics: the bacterial pangenome», Vol 11 (5), Tettelin, et al., pag 472-477, Copyright (2008), con permiso de Elsevier.

el parámetro $\alpha < 1$ por lo que no es posible caracterizar su repertorio genético, y es un pangenoma cerrado si dicho parámetro es $\alpha > 1$ lo cual implica que con un número suficiente de genomas muestreados sí es posible caracterizarlo. De lo anterior se desprende la pregunta de si el ambiente contiene suficientes genes (virtualmente infinitos) para ajustarse a la predicción en el caso de los pangenomas abiertos, lo cual ya ha sido abordado por estudios posteriores[18,19]. El concepto, sin embargo, es útil desde el punto de vista biológico dado que ofrece información sobre la flexibilidad genómica de las especies para intercambiar material genético con el ambiente, y su capacidad de adaptarse a nuevos nichos. Recientemente, por ejemplo, se observó que el *openness* (es decir, qué tan abierto está un pangenoma) está vinculado al lugar taxonómico de la especie, y que la relación entre función génica y posición en el espectro de frecuencias está conservada evolutivamente[20].

Otro estadístico utilizado para entender un pangenoma es el de fluidez (del inglés *fluidity*, ϕ), introducida por Kislyuk y colaboradores en 2011[21] para describir la disimilitud entre los genomas a nivel génico. Es el promedio de la razón de familias génicas únicas entre la suma de familias génicas en pares de genomas seleccionados al azar de los N genomas que conforman el pangenoma:

$$\phi = \frac{2}{N(N-1)} \times \sum_{\substack{k,l=1\ldots N \\ k<1}} \frac{U_k + U_l}{M_k + M_l} \tag{1.3}$$

donde U_k y U_l son el número de familias génicas encontradas sólo en los genomas k y l, respectivamente, y M_k y M_l son el número total de familias génicas encontradas en k y l respectivamente.

Pangenomas más variables tienden a tener mayor fluidez, observándose esa misma propiedad en pangenomas considerados abiertos. Lo contrario ocurre en pangenomas cerrados. Recientemente se ha observado también que la fluidez está muy relacionada al tamaño poblacional efectivo (N_e)[22,23].

1.2. Mecanismos que moldean el pangenoma

A continuación sigue una breve reseña de conceptos teóricos que fueron desarrollándose en las últimas décadas alrededor de la idea de los pangenomas bacterianos, para introducir al lector en el estado del arte de esta relativamente nueva rama de la ciencia.

1.2.1. Evolución del pangenoma, ¿neutral?: Modelo de Genes Infinitos

Uno de los primeros modelos para describir los parámetros que moldean a los pangenomas es el Modelo de Genes Infinitos, descrito por Baumdicker y colaboradores en 2012[24]. El mismo está en línea con el Modelo de Alelos Infinitos[25] y el Modelo de Sitios Infinitos[26], divisados por Kimura y Crow y aplicados desde hace décadas, para describir la diversidad genética. Se parte del coalescente, es decir un árbol ultramétrico de n organismos el cual representa la genealogía «verdadera». Los parámetros que luego describen al pangenoma son el tamaño poblacional efectivo N_e, el cual puede interpretarse como el número de generaciones desde el ancestro común más cercano hasta el tiempo presente, el tamaño de genes indispensables (o genoma núcleo) C; la probabilidad de ganar un gen dispensable υ, y la probabilidad de que un gen dispensable existente se pierda ν. De lo anterior se definen $\theta = 2N_e\upsilon$, y $\rho = 2N_e\nu$, los cuales se corresponden al número promedio de genes ganados o perdidos en $2N_e$ generaciones, respectivamente. Así el tamaño esperado del genoma de un individuo, G^1 se desprende de la siguiente ecuación:

$$\mathbb{E}_{\theta,\rho,C}[G^1] = C + \frac{\theta}{\rho} \tag{1.4}$$

mientras que el el número esperado de genes distintos en toda la muestra de tamaño n se obtiene de la siguiente forma:

$$\mathbb{E}_{\theta,\rho,C}[G^n] = C + \theta \sum_{i=0}^{n-1} \frac{1}{i+\rho} \tag{1.5}$$

Experimentalmente, el desafío se encuentra en estimar ρ y θ ($\hat{\rho}$ y $\hat{\theta}$, respectivamente), lo cual se consigue a partir del coalescente calibrado y el espectro de frecuencias genéticas, maximizando una función de verosimilitud (ver Baumdicker, Hess y Pfaffelhuber, 2012). Este modelo ha sido testeado exitosamente[27], y luego extendido para incorporar la transferencia horizontal de genes[28], y se trata de un modelo de evolución neutral.

Otro estudio clásico[a] que soporta la hipótesis de que la evolución del pangenoma es neutral es el de Andreani y colaboradores[22] (2017) donde observan que existe una correlación entre la fluidez de las distintas especies (ϕ), y la diversidad nucleotídica sinónima de los genes *core* (π_{syn}), como se observa en la figura 1.3. Algo que también llamó la atención en este estudio es que la intersección de la relación entre ϕ y π_{syn} es significativamente distinta de cero, lo cual implicaría que el genoma accesorio diverge antes de que la variación sinónima aparezca en el genoma núcleo.

En la presente tesis, el Modelo de Genes Infinitos tiene particular relevancia en los capítulos 2 y 4.

1.2.2. Evolución del pangenoma, ¿purificadora?: Carrera armamentística en el genoma accesorio

En biología evolutiva se conoce como Hipótesis de la Reina Roja a la propuesta de 1973 de Leigh van Valen[29] para explicar la observación (Ley de extinción, o Ley de van Valen) de que la probabilidad de extinción no depende de la longevidad de la especie, sino que es un proceso al azar cuya tasa depende de la ecología del taxón. El nombre es una referencia a la obra de Lewis Carrol «*A través del espejo y lo que Alicia encontró allí*» publicada en 1871 como secuela de «*Alicia en el País de las Maravillas*», donde la Reina Roja le explica a Alicia que era necesario correr lo más rápido posible tan sólo para mantenerse en el mismo lugar. Se trata de una metáfora para explicar que las especies deben «correr» (es decir, evolucionar rápidamente) simplemente para «mantenerse» (no extinguirse).

Llevado al plano de la pangenómica, una hipótesis para explicar el mantenimiento de los pangenomas es que la mayor parte de los genes accesorios serían

[a]Dada la juventud de esta ciencia, llamamos «clásico» a cualquier estudio con más de 4 o 5 años.

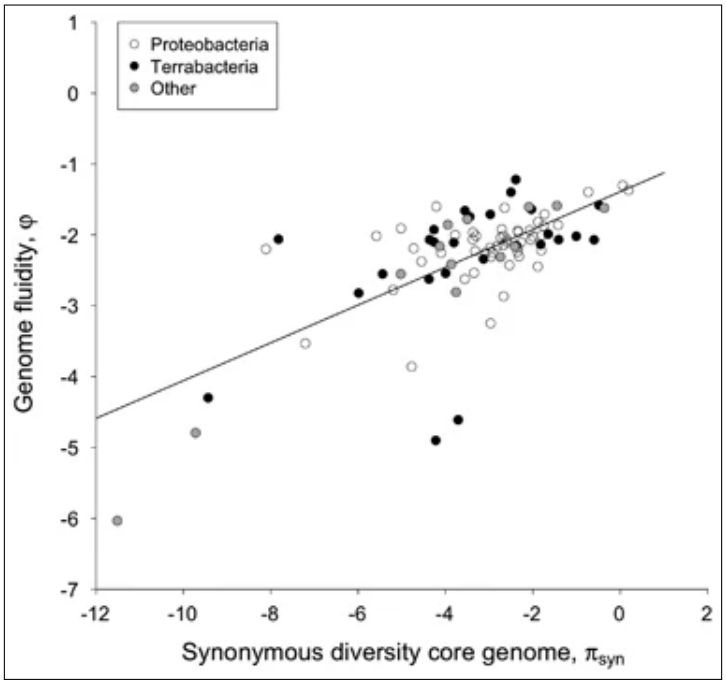


Figura 1.3: Correlación entre la fluidez de los pangenomas y la diversidad sinónima en genes core para 90 especies bacterianas seleccionadas. Tomado y adaptado de Andreani, et al. (2017) «Prokaryote genome fluidity is dependent on effective population size»[22], Licencia: Creative Commons CC BY 4.0.

mayoritariamente deletéreos o parasíticos, y que, tal como se planteaba con la hipótesis de la Reina Roja, la evolución de los pangenomas se explica como una carrera armamentística contra estos elementos.

Vos y colaboradores argumentan que si bien este es el caso, es difícil de observarlo dado que no se puede detectar lo que ya no está[30], por lo que hay que analizar genomas muy cercanos para estimar las tasas de transferencia horizontal de genes (HGT) en primera instancia. Lee y Marx (2012) analizando reducciones genómicas en un experimento que involucraba 32 poblaciones de *Methylobacterium extorquens* durante 1500 generaciones en 4 regímenes nutricionales distintos, llegan a la conclusión que dichas reducciones no eran neutrales sino que desde las poblaciones ancestrales había existido una convergencia guiada por selección purificadora[31].

1.2.3. Evolución del pangenoma, ¿adaptativa?: Selección en el genoma accesorio

Si bien podría explicar la existencia de profagos y pseudogenes, la idea de que la selección purificadora es la más relevante en el mantenimiento de los pan-

genomas, sin embargo, no encuentra demasiado sostén en la evidencia dado que existen múltiples elementos adaptativos en el genoma accesorio, tales como genes metabólicos, o genes de resistencia a antibióticos[32,33].

Por otro lado, un análisis exhaustivo de distribuciones de frecuencias génicas en distintos grupos bacterianos, por parte de Lobkovsky, Wolf, y Koonin (2013), arrojó que la hipótesis neutralista, bajo ciertas asunciones, podía ser rechazada de forma consistente[34]. Similarmente, en 2016, Sela y colaboradores se proponen testear varios modelos evolutivos para explicar la compacidad de los genomas procariotas en cuanto a contenido génico. Sugieren que sólo modelos adaptativos logran explicar la adquisición de genes aunque la selección positiva que gobierna el proceso es probablemente débil. La explicación biológica que ofrecen sería que los procariotas se exponen brevemente a material genético en promedio beneficioso, y que luego tiende a seguir una reducción genómica lenta[35].

Bobay y Ochman (2018) proponen que las especies con un bajo N_e tenderán a eliminar aquellos genes que no aporten una ganancia adaptativa significativa que pueda vencer el efecto de la deriva. Poblaciones con alto N_e, en cambio, pueden mantener genes con una contribución al *fitness* más modesta[23]. Este modelo asume que la mayoría de los genes accesorios son adaptativos, y está muy en línea con lo propuesto por McInerney y colaboradores en un año antes[36]. Respecto a esto último, James O. McInerney viene defendiendo la hipótesis adaptativa del pangenoma, en particular centrado en que buena parte de los genes accesorios se encuentran en un contexto ecológico favorable a las actividades biológicas que proveen[37-40].

1.2.4. ¿...entonces? Seleccionismo vs Neutralismo, otra vez.

La cuestión de si los pangenomas evolucionan de forma neutral o de forma adaptativa en una falsa dicotomía ya que la discusión no se da en estos términos absolutos, sino en el *grado* en que se da una u otra cosa. Eduardo Rocha en 2018 se refería a esto:

> «*We do know that at least some of these genes are adaptive whereas others are deleterious. The fundamental question is: How many? The concomitant operational issue is: Can we identify them?*»
>
> Eduardo P. C. Rocha, 2018. *Neutral Theory, Microbial Practice: Challenges in Bacterial Population Genetics*[41]

Se trata de una discusión ya conocida para la biología evolutiva, que existe desde la introducción de la teoría neutralista de la evolución molecular por Motoo Kimura en 1968[42], sólo que en aquel caso referida a la evolución alélica.

Se han realizado esfuerzos, sin embargo, para reconciliar ambas hipótesis. En un artículo de opinión, por ejemplo, Shapiro (2017)[43] enfatizaba en que la idea

de utilizar la teoría de genética de poblaciones clásica no era del todo correcto, al referirse al trabajo de Andreani y col. (2017)[22], donde defendían la hipótesis neutralista, y al de McInerney y col. (2017)[36], donde se inclinaban por la hipótesis adaptativa, ambos trabajos comentados en las secciones 1.2.1 y 1.2.3, respectivamente. El argumento de Shapiro se centraba en que las hipótesis deberían contemplar la posibilidad de HGT entre poblaciones, o sea, no necesariamente limitado a intercambios intra-especies, por lo que las asunciones de las teorías de genética de poblaciones no eran del todo realistas. Como sugerencia, recomendaba estudiar la posibilidad de aplicar la teoría tomando a los genes como unidad de selección, en lugar de a las especies. Más recientemente, Douglas y Shapiro (2021) desarrollan esta idea[44] enfocándose en los elementos genéticos móviles como entidades egoístas que, si bien no le confieren ninguna ventaja (o incluso le confieren deventaja) al portador, dichos elementos logran diseminarse efectivamente por las poblaciones.

Lo que es claro es que ambas hipótesis coexisten, y que incluso aunque la hipótesis adpatativa parezca tener más evidencia a su favor, la hipótesis neutralista continuará explicando una parte del genoma accesorio, y servirá, en todo caso, como hipótesis nula para testear nuevos modelos adaptativos.

1.2.5. La Reina Negra, y el pangenoma como un recurso genético compartido

La Hipótesis de la Reina Negra es un claro guiño a la hipótesis de van Valen (ver sección 1.2.2), aunque también hace referencia a un juego de naipes, *Hearts*, donde el objetivo es obtener la mínima cantidad de puntos evitando ciertas cartas, en especial a la Q de Picas (la *Reina Negra*) ya que esta vale como la suma de todo el resto de las cartas combinadas, pero en todo momento algún jugador debe poseerla. Fue propuesta por Morris, Lenski y Zinser en 2012[45,46] para explicar que ciertos genes (o funciones biológicas), representan un costo fisiológico importante para quienes los portan, pero son un bien público para la comunidad, por lo que por un lado existen presiones selectivas para ser eliminados de los genomas, y al mismo tiempo es necesario que un sub grupo de los individuos que componen la comunidad los mantengan (alguien, después de todo, debe poseer a la *Reina Negra*).

Existe también la versión «fuerte» de esta hipótesis, en la cual esas funciones biológicas no están presentes en algunas pocas cepas *keystone*, sino que se encuentran distribuídas en toda la comunidad y cada cepa es mutuamente dependiente de todas las demás[47], algo similar a lo que se mencionaba en la sección 1 con los «progenotes» de Woese, las pre-células de Kendler, y las meta-células de Lawrence.

Ya desde hacía unas décadas que la HGT tomaba relevancia como medio sobre el cual los recursos genéticos eran compartidos, y como fuerza evolutiva de porte al

menos a nivel bacteriano. Incluso previo al desarrollo del concepto del pangenoma, en 2002, Woese manifestaba:

> «*Only a decade ago HGT [Horizontal Gene Transfer] was generally considered a relatively benign force, which had sporadic and restricted evolutionary impact. However, the HGT that genomics reveals is not of this nature. It would seem to have the capacity to affect the entire genome, and given enough time could, therefore, completely erase an organismal genealogical trace. This is an evolutionary force to be reckoned with, comparable in power and consequence to classical vertical evolutionary mechanisms.*»
>
> Carl R. Woese, 2002. *On the evolution of cells*[48]

por lo que era necesario asumir la evolución comunal, más que la individual, ya que la comunidad era la fuente principal de innovación genética. Esta idea continuó afianzándose en los años posteriores.

1.2.6. Hipótesis de bienes públicos

Otros autores han abordado la discusión del pangenoma como un recurso compartido desde el plano económico. En 2011, McInerney y colaboradores plantean la hipótesis de los bienes públicos para explicar la evolución biológica en la tierra[49]. Si bien en esta ocasión no lo incorporaron, el concepto de pangenoma calzaba naturalmente en la hipótesis desde un inicio, y posteriores desarrollos lo incluyeron[50,51].

La teoría económica de los bienes públicos, en la cual se basaron estos investigadores, fue propuesta por Paul Samuelson, premio Nobel en economía en 1970, quien propuso que los bienes de consumo colectivo se definían como bienes no rivales («*no-rivarly*»), es decir que todos los individuos lo podían disfrutar en común, y que el consumo por uno de ellos no implicaba sustracciones del mismo consumo por parte de otros individuos[52] (si bien él no lo llamó así, es el término que se utiliza hoy en día para referirse a este concepto). Pocos años después, Musgrave introduce un criterio alternativo, la excluibilidad («*excludability*»), definida como la capacidad de prevenir a los individuos de consumir cierto bien[53]. Bajo estas definiciones, un bien es público si es no rival y no-excluible. Los bienes rivales y excluibles son típicamente bienes privados (por ejemplo, una casa). También existen bienes rivales y no-excluíbles (recursos comunes, imaginar un bosque común del que todos pueden sacar leña, pero el bosque es finito por lo que puede darse una situación de tragedia de los comunes)[54], conceptualizados por Elinor Ostrom (premio Nobel en 2009), y bienes que son no rivales pero sí excluíbles (bienes de grupos, por ejemplo una subscripción a un club, sólo los socios pueden acceder pero una vez que se es

socio se tiene libre acceso), estudiados por James M. Buchanan, premio Nobel en 1986[55]. El anterior párrafo puede sintetizarse en el siguiente cuadro:

Cuadro 1.1: Resumen de tipos de bienes económicos.

	Rivales	**No rivales**
Excluíbles	Bienes privados	Bienes de grupos (clubes)
No excluíbles	Recursos comunes	Bienes públicos

El Ácido desoxirribonucleico (ADN) que queda libre en el ambiente tiene la propiedad de ser no-excluíble, es decir, todos los organismos de dicho ambiente pueden acceder a él, y es no rival en el sentido práctico de que las múltiples copias del mismo no pueden ser usadas por un único organismo o especie, excluyendo al resto. Bajo estas condiciones, el ADN ambiental puede considerarse un bien público, aunque en algunos casos podría decirse que se comportan más como bienes grupales o de clubes[49].

La principal implicancia de esta hipótesis es abandonar el pensamiento en términos de árbol en evolución («*tree-thinking*»), donde todos los genes, o secuencias de ADN en general, de una especie se comportan como bienes privados (excluíbles y rivales) y comparten una misma historia evolutiva, y en cambio considerar la idea de redes para describirla bajo el paradigma de bienes públicos («*goods-thinking*»).

Los autores que defienden esta hipótesis también declaran que dada la fluidez de los genomas, causada por la ganancia y pérdida de genes accesorios, el análisis de los pangenomas debería pensarse más en términos de redes (o grafos) que en términos de árboles[51].

1.3. Categorización de las familias génicas

La clasificación clásica de las familias génicas en un pangenoma es agruparlas en 2 categorías: genes *core*, y genes accesorios o dispensables[10]. Estas definiciones, *sensu stricto*, dividen a las familias génicas entre aquellas presentes en el 100 % de los genomas considerados, de aquellas presentes en $\leq 99\,\%$. Existe también la versión *sensu lato* de estas definiciones, que implica tener en cuenta la posibilidad de que ciertas familias génicas no estén presentes en los datos, pero no por no existir realmente, sino por sesgos metodológicos y/o errores de secuenciación, permitiendo que algunas familias génicas de muy alta frecuencia, pero no presentes en el 100 % de las cepas, sean consideradas *core*. Usualmente esta definición laxa de genes core puede incluir a aquellos presentes en $\geq 99\,\%$, o incluso a veces se relaja esta condición a $\geq 95\,\%$.

Otros optan por subdividir esta clasificación de genes *core sensu lato* en genes *core*, para aquellos presentes en $\geq 99\,\%$, y genes *softcore*, presentes entre $\geq 95\,\%$ y $< 99\,\%$ de las mismas; así como también subdividir el genoma accesorio en genes *shell*, y genes *cloud*, para aquellos genes presentes entre $< 95\,\%$ y $\geq 15\,\%$, y aquellos entre $> 0\,\%$ y $< 15\,\%$, respectivamente[56].

El genoma «único» consiste en genes presentes únicamente en una sola cepa, también llamados *singletons*, clasificación que tiende a coincidir con el genoma *cloud*, y muchas veces se usan ambos términos para refereirse a genes cepa-específicos indistintamente.

1.3.1. Definición analítica de las categorías genéticas

Más recientemente, Hyun y colaboradores (2022)[20] desarrollaron definiciones analíticas para categorizar a las familias génicas en genes *core*, *shell*, y *cloud*.

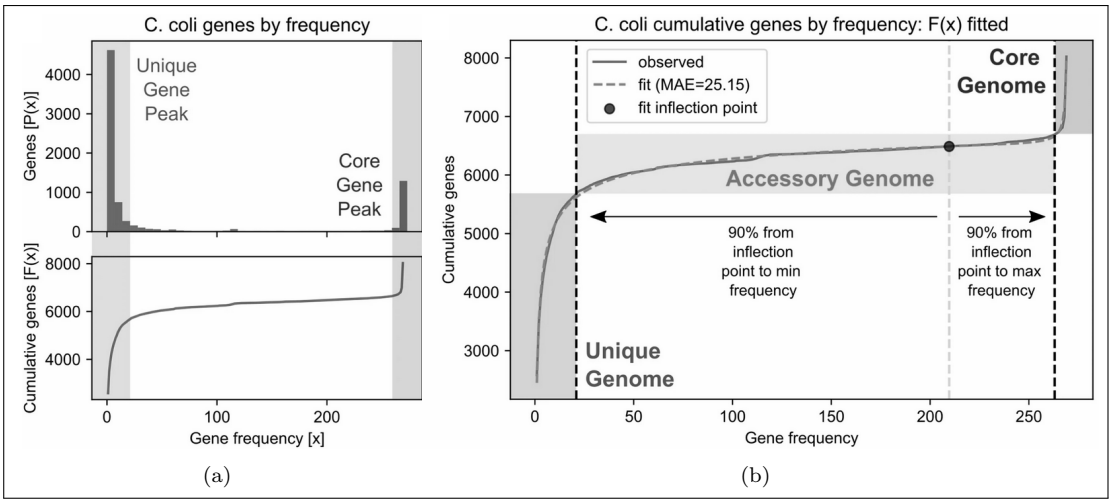


Figura 1.4: Pangenoma de *Campylobacter coli*. (a) Gráficos de frecuencias génicas y su suma acumulada. (b) Descripción gráfica del método para identificar las distintas categorías génicas. Tomado y adaptado de Hyun, et al. (2022) «Comparative pangenomics: analysis of 12 microbial pathogen pangenomes reveals conserved global structures of genetic and functional diversity»[20]. Licencia: Creative Commons CC BY 4.0.

Para describir matemáticamente el histograma de frecuencias génicas (figura 1.4a, arriba), los autores proponen la siguiente ecuación:

$$P(x) = c_1 x^{-\alpha_1} + c_2 (N + 1 - x)^{-\alpha_2} \tag{1.6}$$

donde $x = 1, 2, \ldots, N$, siendo N el número de genomas considerados. Los parámetros c_1, α_1, c_2, y α_2, son parámetros libres cuya determinación se comenta

más adelante. El término de la izquierda, $c_1 x^{-\alpha_1}$, representa una exponencial negativa, y el de la izquierda, $c_2(N+1-x)^{-\alpha_2}$, una exponencial positiva, ambas con mínimo en 0. La suma de ambos términos da como resultado una curva en forma de «U» asimétrica lo cual se ajusta al gráfico de frecuencias génicas.

Para determinar los parámetros libres, los autores integran la ecuación 1.6, obteniendo la curva acumulativa de la misma (figura 1.4a, abajo):

$$F(x) = k + \frac{c_1}{1-\alpha_1} x^{1-\alpha_1} - \frac{c_2}{1-\alpha_2}(N+1-x)^{1-\alpha_2} \tag{1.7}$$

y definen las categorías relativas al punto de inflección, x^* (o lo que es lo mismo, el mínimo en la ecuación 1.6), donde los genes únicos son aquellos presentes en menos que $0{,}1x^*$ cepas, los genes *core* aquellos presentes en más de $0{,}9N + 0{,}1x^*$, y los genes accesorios todos los que queden en el medio, ver figura 1.4b.

1.3.2. Clasificación «*twilight*»

Otra aproximación para clasificar a las familias génicas se basa en tener en cuenta la estructura poblacional y el sesgo de muestreo del grupo a estudiar, como proponen Horesh y colaboradores (2021)[57]. El planteo se justifica al considerar que en una colección de genomas, por ejemplo, un solo linaje puede estar muy sobrerrepresentado respecto al resto si por alguna razón es de particular interés para la comunidad. Si esta cepa llegara a contener cierto gen que está ausente en el resto de los genomas, esta familia génica, bajo la clasificación tradicional, sería clasificada como «intermedia» o «*shell*», cuando en realidad se trata de un gen «*singleton*» cepa específico.

En el trabajo, estas investigadoras e investigadores proponen clasificar a las familias génicas en 13 categorías basadas en la frecuencias génicas dentro y fuera de linajes determinados por algún método de agrupamiento poblacional (figura 1.5), como por ejemplo BAPS[58], PopPUNK[59], secuenciotipo, o serotipo, entre otros.

1.3.3. Esencialidad genética y pangenómica experimental

Con la secuenciación de los primeros genomas bacterianos en los años 90[3], como se comentó en la sección 1, nace también la genómica comparada. Al mismo tiempo, dadas las posibilidades de las nuevas técnicas, renace el interés por los genomas mínimos y la esencialidad genética[15,60]. En un trabajo pionero, Hutchison, Peterson, y colaboradores (1999)[61] utilizan mutagénesis global mediante transposones en *Mycoplasma genitalium* para identificar genes no esenciales (en condiciones de laboratorio), llegando a la conclusión que entre 265 y 350 genes codificantes para proteína, de un total de 480, eran esenciales. Caben destacar también los trabajos de Akerley y col. (1998)[62], y de Reich y col. (1999)[63] describiendo abordajes

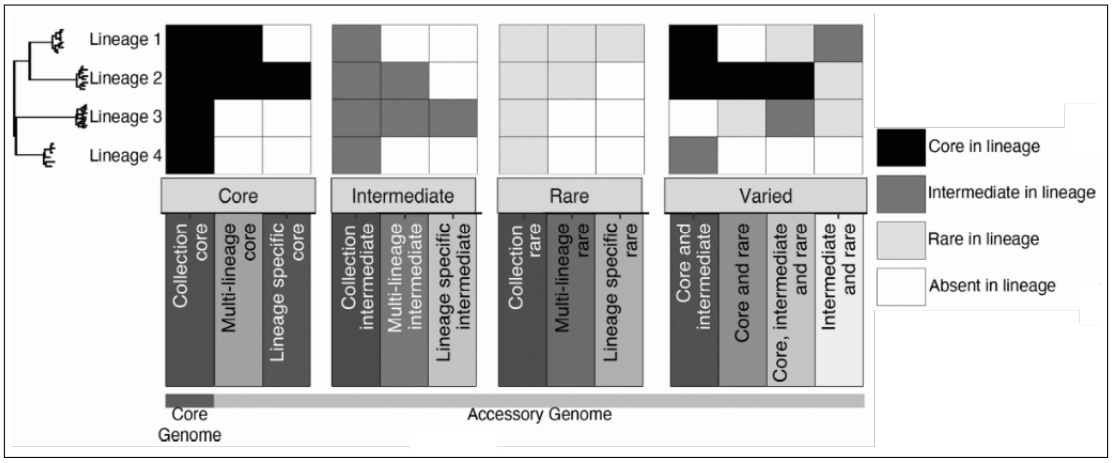


Figura 1.5: Iluistración esquemática de las categorías génicas definidas por Horesh y colaboradores. Tomado y adaptado de Horesh, et al. (2021) «Different evolutionary trends form the twilight zone of the bacterial pan-genome»[57], Licencia: Creative Commons CC BY 4.0.

similares y aplicándolas en *Haemophilus influenzae* y *Streptococcus pneumoniae*, aunque en estos dos casos las técnicas no se basaban en mutagénesis global sino en mutagénesis *in vitro* de productos de PCR de segmentos del genoma, y en la posterior transformación de cepas naturales con estos productos mediante recombinación homóloga. Si bien ya se habían utilizado técnicas similares para estudiar auxotrofías y función génica, estos tres trabajos fueron los primeros en aplicar un abordaje de mutagénesis masiva mediante transposones en la era postgenómica (es decir, con los genomas de los organismos estudiados disponibles), lo cual permitía mapear estos genes en el genoma. A estos trabajos siguieron varios en los años subsiguientes tomando a otros organismos como modelo: en *Mycobacterium bovis* (2001)[64], y en *Saccharomyces cerevisiae* (2002)[65], por nombrar algunos ejemplos. En 2009 cuatro trabajos independientes[66-69], con variaciones entre ellos, presentan lo que podemos englobar bajo el nombre de métodos de Secuenciación por Inserción de Transposones (SIT), los cuales combinan mutagénesis a gran escala por transposones con secuenciación de nueva generación, democratizando en buena medida los análisis de esencialidad génica.

Otro conjunto de técnicas que permiten hacer este tipo de estudios a gran escala en bacterias son las basadas en CRISPR/Cas9 interferente, desarrolladas más recientemente[70,71], las cuales permiten silenciar genes mediante interferencia de Ácido ribonucleico (ARN) sobre los mensajeros. La ventaja de esta técnica sobre la disrupción genética de las SITs es que es posible silenciar a demanda mediante un promotor de la activación del sistema inducible. Esto permite estudiar a los genes esenciales de otra forma que no sea *por la negativa* (es decir, identificando

a los genes esenciales como aquellos que no posean inserciones de transposones), y al mismo tiempo estudiar el grado de escencialidad ya que el nivel de inducción del sistema es variable, y no simplemente de encendido o apagado.

Con el desarrollo de la pangenómica, los conceptos de genes *core* y genes esenciales comienzan a converger en un mismo campo de estudio, que aunque si bien son conceptos diferentes, comparten una intersección *a priori* evidente. Las técnicas antes mencionadas permiten hoy en día estudiar no sólo la esencialidad genética en una cepa, sino a nivel poblacional e incluso de toda la especie. En 2019, Poulsen y col. definen el genoma esencial *core* mediante una combiación de técnicas computacionales para identificar los genes *core*, y posterior aplicación de SITs en *Pseudomonas aeruginosa*[72]. Una de las preguntas que buscaron responder es: ¿cuántas cepas deben ser examinadas para converger a un conjunto de genes que sean potencialmente esenciales en condiciones de infección (y por lo tanto, que sean buenos candidatos para el desarrollo de drogas)?

Más recientemente, también combinando las técnicas de SITs con pangenómica computacional, Rosconi y colaboradores (2022) sugieren que la esencialidad genética es contexto dependiente, y a su vez clasifican al «esencialoma» (genes esenciales en al menos una cepa) en 3 clases: genes esenciales universales, genes *core* no esenciales (es decir, presentes en todas las cepas pero sólo esenciales en algunas), y genes accesorios esenciales (genes accesorios que son esenciales cuando están presentes), clasificación que guarda relación con la propuesta por Horesh y colaboradores (2021)[57], discutida en la sección 1.3.2.

Los recientes desarrollos en estas áreas no son sólo interesantes desde el punto de vista evolutivo, sino que a nivel clínico pueden utilizarse para identificar genes de resistencia a antibióticos, o blancos para terapias farmacológicas.

1.4. Estudios de asociación de genotipos a fenotipos en pangenomas bacterianos

Los Estudios de Asociación de Genoma Completo, comunmente llamados *GWAS* por sus siglas en inglés (*Genome-Wide Association Studies*), han sido aplicados con éxito en genomas eucariotas para identificar relaciones causales entre rasgos fenotípicos y Polimorfismos Nucleotídicos Simples (*SNPs*, por sus siglas en inglés) desde hace al menos dos décadas desde la primera aplicación[73].

Para el caso de organismos procariotas, estas técnicas no son fácilmente trasladables en su forma original dados los procesos evolutivos que eventualmente configuran a los pangenomas, es decir: los procesos de recombinación, ganancia y pérdida de genes, y la transferencia horizontal de genes intra e inter especie. Existen, sin embargo, metodologías basadas en k-meros y en identificación de variantes

(SNPs) del genoma núcleo que abordan este problema con eficacia[74-76], englobadas dentro de lo que se conocen como bac-GWAS («bac» por *bacterial*), o mGWAS («m» por *microbial*).

A su vez es posible, y muy frecuentemente es el caso, que en bacterias los fenotipos se relacionen a la presencia o ausencia de genes, más que a SNPs. Debido a eso es que se acuña el término «pan-GWAS» para referirse a la aplicación de técnicas de asociación estadística de genotipos a fenotipos en el contexto de pangenomas bacterianos, para diferenciarlo del GWAS tradicional[77]. Este tipo de estudios usa la *pan-matrix* (matriz binaria que relaciona la presencia o ausencia de las familias génicas en cada genoma) como *input* frente a una matriz de características fenotípicas las cuales se busca asociar estadísticamente al estado (1 o 0) de cada relación genoma/familia génica.

1.5. Otros desarrollos recientes

En los últimos años el concepto de pangenoma se ha mezclado con otras disciplinas y ha encontrado otros ámbitos de aplicación más allá de los grupos bacterianos. A continuación una breve mención de estos avances.

1.5.1. Meta-pangenomas (y pan-metagenomas)

La metagenómica es un conjunto de técnicas y abordajes que buscan determinar la composición taxonómica y funcional de los organismos presentes en muestras ambientales utilizando técnicas de secuenciación masiva independientes de cultivo, y desde su conceptualización a la fecha, han revolucionado a la microbiología[78].

La conjunción de la metagenómica y la pangenómica resulta, de alguna forma, intuitiva. En un trabajo pionero, Kashtan y colaboradores (2014) describen un método para analizar la variación genómica y contenido genético de subpoblacions coexistentes de *Prochlorococcus* mediante genómica de célula única (*single-cell genomics*)[79] a partir de muestras ambientales, lo cual puede considerarse una forma de conjunción de ambos conceptos, si bien los autores no mencionan ninguna de las palabras que aparecen en el título de esta subsección.

Delmont y Eren (2018) acuñan como «metapangenoma» a la unión de ambos conceptos, basandose en la aplicación de técnicas basadas en el mapeo de *reads* de metagenomas contra genes del pangenoma de *Prochlorococcus*, este último determinado a partir de genomas de aislamientos disponibles[80].

Al siguiente año, Ma y colaboradores (2019-2020) también desarrollan el concepto de «meta-pangenoma», aunque para referirse al uso de genomas ensamblados a partir de metagenomas para reconstruir el pangenoma de cierta especie, mencionando las ventajas y desafíos que ello conlleva. Al mismo tiempo definen «sub-

especies metagenómicas» refiriéndose a grupos de poblaciones bacterianas, obtenidas mediante secuenciación masiva, que comparten un conjunto similar de genes, y «pan-metagenomas» para describir a todos los pangenomas presentes en cierto hábitat[81,82]. Este último concepto no debe confundirse al de «pan-metagenomas» y «core-metagenomas» que Shafquat y colaboradores (2014) usan para clasificar a los integrantes del microbioma humano (o de cualquier ambiente en particular), según se encuentren de forma consistente (*core*) en dicho hábitat, o no[83]. Estos autores utilizan los prefijos *pan* y *core* más como analogía que para describir la unión de ambas disciplinas.

Como sucede muchas veces y en este caso en particular, al tratarse de una sub disciplina relativamente jóven, muchos de los conceptos han sido acuñados más de una vez simulatáneamente y redefinidos, por lo que no están debidamente estandarizados en la literatura.

1.5.2. Pangenómica en organismos eucariotas

Como se describió en el comienzo de este capítulo, la idea del pangenoma surge para representar la enorme variabilidad en contenido genético que poseen las especies procariotas. Más recientemente, sin embargo, el concepto se extendió a hongos, plantas, e incluso animales, aunque con grandes matices respecto a la definición clásica dada la diametralmente distinta estructura genómica de los organismos eucariotas, en los que hay diploidía y poseen una mucho mayor proporción de regiones extragénicas. Estas diferencias han llevado a que se aborden los pangenomas desde dos perspectivas distintas: una centrada en los genes, y otra centrada en las secuencias[84]. En el primer caso se define al pangenoma como la unión de todos los genes o grupos de ortólogos, como tradicionalmente se ha enfrentado el problema en procariotas, mientras que en el segundo caso se define como el conjunto completo de secuencias no redundantes, incluyendo regiones intergénicas. Potencialmente, un pangenoma eucariota puede estar «cerrado» desde el punto de vista del contenido génico, y «abierto» teniendo en cuenta la totalidad de las secuencias.

1.6. Homología, ortología, paralogía.

Algunos conceptos fundamentales que preceden a la idea misma de pangenoma son los de homología, ortología, y paralogía. En la sección 1.1 se mencionaba que quienes acuñaron el término, explícitamente descartaron considerar la microdiversidad, de modo de considerar a los distintos alelos de un mismo gen como una única cosa. Y al hablar de «un mismo gen» debemos evitar utilizar un criterio bioquímico o funcional para agrupar a dos entidades genéticas que pueden ser

análogas, es decir, haber convergido independientemente, y restringirnos a criterios filogenéticos.

Definición 1. ***Analogía:*** *Dos secuencias son análogas si no descienden de un ancestro común pero comparten función o similitud de secuencia por convergencia evolutiva*[85]*.*

Definición 2. ***Homología:*** *Dos secuencias son homólogas si comparten un origen monofilético y evolucionaron divergentemente*[85]*.*

También es necesario precisar que nos referimos a homología entre organismos, no a homología entre genes de un mismo organismo. Dicho de otro modo, se trata de considerar a genes estrictamente ortólogos como pertenecientes a una misma familia génica.

Definición 3. ***Ortología:*** *Dos secuencias son ortólogas si descienden de un ancestro común y están separadas por un evento de especiación, de modo que la historia del gen refleja la historia de las especies*[85]*.*

La definición clásica anterior puede relajarse para considerar, por ejemplo, genes de una misma especie pero de distintas cepas o individuos, si existiera microdiversidad, por lo que no es necesario estrictamente el evento de especiación sino que la historia de los genes y de los individuos coincidan. La definición de ortología deja por fuera a aquella relación de homología intra organismo, a la cual nos referimos por paralogía.

Definición 4. ***Paralogía:*** *Dos secuencias son parálogas si su homología es el resultado de una duplicación génica, por lo que ambas copias descendieron en paralelo durante la historia del organismo*[85]*.*

Estos conceptos fueron acuñados por Fitch en 1970[85], y refinados por Sonnhammer y Koonin en 2002[86] para describir algunas sub relaciones entre parálogos, que eventualmente fueron relevantes para el estudio de pangenomas. Como fue notado por primera vez por Ohno en 1970, las duplicaciones génicas cambian las presiones selectivas sobre las funciones creando un entorno evolutivo distinto que tiende a hacer que al menos una de las copias diverja en función rápidamente[87].

Definición 5. ***Paralogía interna:*** *Dos secuencias son parálogas internas si son resultado de una duplicación génica luego de la radiación (especiación) que separa a dos linajes considerados*[86]*.*

Definición 6. ***Paralogía externa:*** *Dos secuencias son parálogas externas si la duplicación se dio antes de la radiación entre ambos linajes considerados*[86]*.*

Si bien las definiciones anteriores son estrictamente filogenéticas, desde el punto de vista práctico es a veces difícil determinar dichas relaciones siguiendo metodologías con filogenias, por lo que se utiliza información de similitud, sintenia, y contexto génico para ayudar a delimitarlas, como se verá más adelante.

Definición 7. ***Sintenia:*** *Presencia de dos o más textitloci en el mismo cromosoma.*

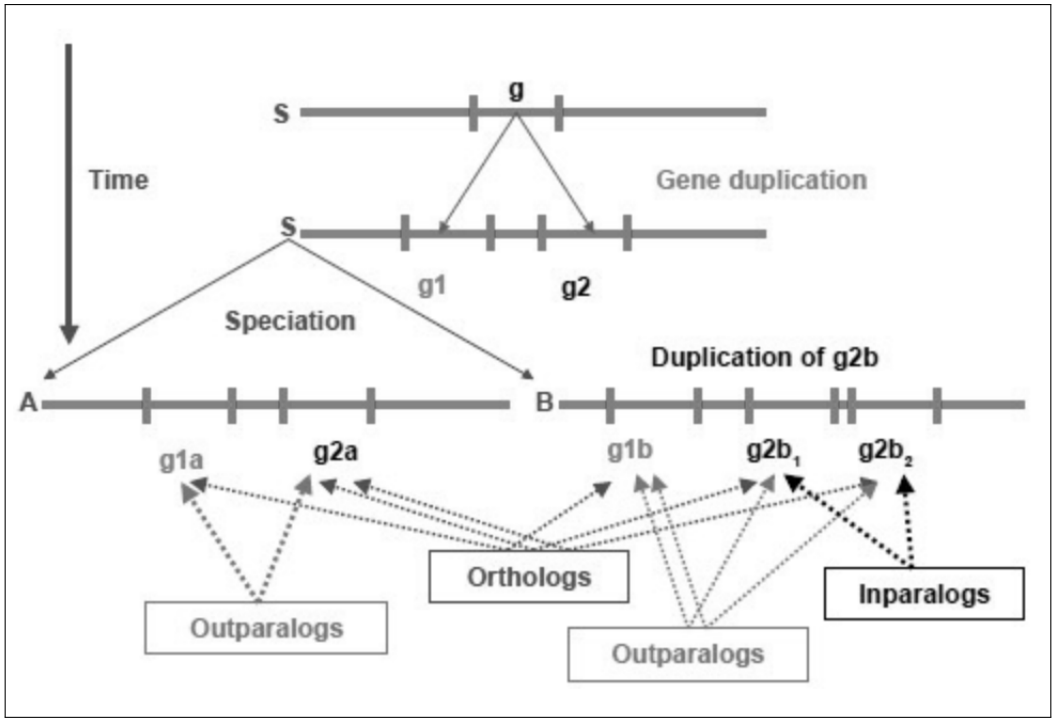


Figura 1.6: Ilustración esquemática de los conceptos de genes ortólogos, parálogos internos, y parálogos externos. Tomado y adaptado de Tekaia (2016) «Inferring Orthologs: Open Questions and Perspectives»[88], Licencia: Creative Commons CC BY-NC 3.0.

1.6.1. Búsqueda de homología

Las técnicas computacionales para la identificación de homología entre genes en general se han basado en utilizar similitud de secuencia como base, para luego aplicar alguna técnica de agrupamiento (*clustering*). Los primeros abordajes automáticos, por ejemplo, fueron basados en el motor de búsqueda BLASTp[89] para identificar *hits* entre dos organismos[90-93]. Otros posteriores se han basado en motores de búsqueda más modernos, como DIAMOND[94], pero que escencialmente siguen la misma estrategia[95].

Como se mencionó, también es necesario un segundo paso que filtre los *hits* para retener sólo aquellos significativos desde el punto de vista biológico. Un primer abordaje es filtrar los alineamientos basándonos en un valor de corte fijo para un *e-value*, un porcentaje de similitud, cobertura, o una combinación de las anteriores; el problema radica en la poca flexibilidad del método y en que no hay suficientes asunciones evolutivas para obtener un resultado considerable. La estrategia de mejor *hit* recíproco[96] ha sido útil cuando se trata de organismos genéticamente cercanos, pero se sabe que no es una metodología óptima para identificar ortólogos[97]. Un método popular para agrupar las secuencias en base a su similitud es MCL[98] (*Markov Clustering*), el cual se trata de un algoritmo de agrupamiento no supervisado para grafos, lo que permite no depender de umbrales fijos. Como *input* requiere una tabla con un valor de similitud para cada par de secuencias considerado. El grafo sobre el cual trabaja el MCL se forma con las secuencias conformando los nodos, y cada par de nodos unidos por un vértice con un peso que representa el valor de similitud entre ambas.

Cabe hacer la salvedad de que el uso de la similitud de secuencia para buscar homología es un *proxy*, ya que la homología en sí no puede medirse por tratarse de un concepto cualitativo. Lo que se busca es obtener buenas hipótesis de homología[99] a través de medidas de similitud de secuencia.

1.6.2. Heurísticas y redundancia

La estrategia de comparar todas las secuencias entre sí (lo cual llamaremos estrategia *all vs all*) mediante un motor de búsqueda como BLASTp o DIAMOND rápidamente se volvió inviable dado el crecimiento en el número de genomas disponibles para análisis. Se trata de un problema de explosión exponencial de complejidad con cada nuevo genoma considerado, por lo cual se comenzaron a aplicar pasos de reducción de redundancia previo a realizar comparaciones con métodos más sofisticados, y de esta forma evitar realizar alineamientos entre pares de secuencias que son suficientemente similares. La idea es utilizar métodos ultra rápidos, pero menos sensibles, para pre agruprar secuencias que son iguales o *muy parecidas*, seleccionar una sola secuencia de cada uno de estos *preclusters*, comparar sólo este *subset* de secuencias representativas entre sí mediante los motores más complejos y computacionalmente más caros para posteriormente aplicar el método de agrupamiento, y asignar las secuencias no seleccionadas de los *preclusters* al grupo al fue asignada su representativa. Con este tipo de estrategias se logran reducir los tiempos de ejecución varios órdenes de magnitud, casi sin pérdida en la calidad de los agrupamientos finales, permitiendo reconstruir pangenomas de decenas o incluso cientos de organismos en tiempos razonables.

Uno de los primeros programas que utilizaron esta estrategia de reducción de redundancia, y tal vez el programa más citado para reconstruir pangenomas, es

Roary, desarrollado por Page y colaboradores[56] (2015). Para dicho paso utiliza CD-HIT, desarrollado por Li y Godzik (2006; 2012)[100,101], que implementa una estrategia *greedy* incremental basada en conteo de k-meros para estimar similitudes. Otros *softwares* populares utilizados para reducir la redundancia son Linclust[102] y MMSeqs2[103].

1.6.3. Distinción entre parálogos y ortólogos

Como ya se mencionó, el uso de medidas de similitud de secuencia por sí sólo no es óptima para identificar secuencias ortólogas[97]. Algunas implementaciones han utilizado el contexo genómico, basándose en la conservación del vecindario genético, para distinguir ortólogos de otros tipos de homología[56,104,105], estrategia que suele dar buenos resultados cuando se trata de organismos filogenéticamente cercanos (pertenecientes a una misma especie), pero que probablemente haga aguas en situaciones donde el tiempo evolutivo entre los genomas considerados es tal que buena parte de los contextos genómicos se pierdan debido a rearreglos cromosómicos, inserción de secuencias, pseudogeneización, etc.

Al tratarse de conceptos filogenéticos, hay autores que enfatizan en el uso de algoritmos filogenéticos, como por ejemplo la reconciliación de árboles, donde se compara al árbol del linaje con el árbol de cierta famila génica para determinar las relaciones de ortología y paralogía[88,106]. Para este tipo de análisis, sin embargo, es necesario contar con el árbol del linaje estudiado, lo cual no siempre se tiene.

La transferencia horizontal de genes representa un desafío para cualquier método dado que la historia evolutiva de estos genes no se corresponde con la del linaje estudiado, agregando complejidad al problema.

1.7. Herramientas bioinformáticas

A continuación se describen brevemente algunas de las herramientas computacionales más utilizadas para la reconstrucción y análisis de pangenomas bacterianos. El órden elegido es según fecha de publicación. No se incluyen las que se presentan en ésta tesis, las cuales tienen capítulos dedicados.

1.7.1. Reconstrucción de pangenomas

- **OrthoMCL** (2003)[90]: El primer *software* bioinformáticos capaz de identificar grupos de ortólogos de más de dos organismos. Requiere computar *blastp*[89] de todos los pares posibles de proteínas, y realiza el agrupamiento mediante el algoritmo MCL[98].

- **PGAP** (2012)[107]: Para identificar ortólogos y parálogos hace uso de alguno de dos métodos: MultiParanoid[108], u otro basado en blastp[89] y MCL[98]. En ambos abordajes se utilizan umbrales de corte de identidad y cobertura para filtrar *hits*.

- **micropan** (2015)[93]: Paquete de R[109] que permite agrupar las secuencias mediante una de dos metodologías: un abordaje *all vs all* con blastp[89] como motor, y posterior agrupamiento jerárquico, o agrupamiento por arquitectura de dominios proteicos utilizando HMMER[110,111] y una versión local de Pfam[112,113].

- **Roary** (2015)[56]: Probablemente sea el *software* más usado para reconstruir pangenomas bacterianos. Aplica una heurística para pre agrupar los genes similares de forma rápida y eficiente con CD-HIT[100,101], y luego toma una secuencia representativa de estos pre agrupamientos y las compara entre sí usando un método más sensible, como blastp[89], disminuyendo órdenes de magnitud los tiempos de ejecución respecto a los otros programas disponibles hasta el momento de la publicación, casi sin pérdida de sensibilidad.

- **BPGA** (2016)[114]: Reconstruye relaciones mediante uno de tres motores de búsqueda: USEARCH[115] (default), CD-HIT[100,101], u OrthoMCL[90], estableciendo umbrales de corte fijos de acuerdo a la identidad de secuencia.

- **PanX** (2018)[95]: Utiliza DIAMOND[94] para calcular *score* de alineamientos mediante una estrategia *all vs all*, luego aplica MCL[98], y por último separa ortólogos de parálogos alineando cada pre grupo, computando un árbol con FastTree[116], y utilizando un método analítico para identificar ambas categorías.

- **PIRATE** (2019)[117]: Establece las relaciones de ortología y paralogía usando un método iterativo sobre múltiples umbrales de identidad de secuencia determinadas mediante CD-HIT[100,101], blastp[89] o DIAMOND[94], y usando MCL[98] como método de agrupamiento.

- **PPanGGoLiN** (2020)[105]: Establece las relaciones de ortología mediante un abordaje de grafos, definiendo cada presunta familia génica como un nodo y conectando los nodos si comparten vecindad genómica. Posteriormente aplica un método de maximización de expectativas para devolver el grafo más probable.

- **Panaroo** (2020)[104]: utiliza una combinación de métodos basados en identidad de secuencia, y grafos representando vecindad genómica, para identificar las relaciones de ortología.

1.7.2. Post-análisis: descripción estadística, visualización, y manipulación de datos

- **micropan** (2015)[93]: Mencionado en la sección anterior, cabe destacar que al ser un paquete de R permite no sólo reconstruir sino que analizar los resultados de forma interactiva dentro del ecosistema de R[109], y es particularmente robusto en proporcionar métodos de descripción estadística de los pangenomas.

- **PanViz** (2017)[118]: Aplicación escrita en Javascript usando la biblioteca D3[119], acompañada de un paquete de R que permite generar los archivos de *input* a la misma. Permite visualizar genomas individuales o en contexto con el pangenoma, y realizar consultas de forma gráfica e intuitiva.

- **PanX** (2018)[95]: Ya mencionado en la sección anterior, además de un método para reconstruir pangenomas, proporciona una interfaz web de tipo *dashboard* para explorarlos de forma gráfica.

- **PanACEA** (2018)[120]: Escrito en lenguaje Perl, devuelve archivis HTML, JSON y Javascript que pueden ser levantados con cualquier navegador en forma de página web interactiva. Es capaz de procesar datos pangenómicos generados mediante otras herramientas de reconstruicción de pangenomas.

- **Anvi'o** (2020)[121]: Se trata de los visualizadores de datos multi-ómicos más completos y usados por la comunidad científica, y no sólo es posible visualizar sino que es capaz de procesar datos crudos. No se limita sólo a pangenomas, tiene módulos de procesamiento de genomas individuales y metagenomas, realiza análisis de genética de poblaciones y filogenómica. Es capaz de devolver figuras de alta calidad adecuadas para publicaciones.

- **PanExplorer** (2022)[122]: Aplicación web que permite ingresar datos crudos y reconstruir el pangenoma mediante herramientas disponibles de terceros (por ejemplo PGAP[107] o Roary[56]) y generar visualizaciones interactivas a partir del resultado. Permite reevaluar los resultados de *subsets* de genomas elegidos por el usuario.

1.8. Ésta tesis

Desde un inicio, el concepto de pangenoma fue construido y aplicado a genomas pertenecientes a una misma especie. La extensión de este concepto a otros niveles taxonómicos, tales como género o familia, resulta natural desde el punto de vista evolutivo e incluso ya se ha propuesto a nivel teórico[123]. Las herramientas

desarrolladas hasta el momento, sin embargo, se enfocan sólo a nivel de especie y más recientemente los abordajes de grafos utilizan información del contexto génico e identifican rearreglos dentro de una población relativamente homogénea o cepa.

Debido a lo anterior, nos propusimos abordar este nicho metodológico relativamente poco explorado dentro del área de conocimiento, es decir, desarrollar un *software* capaz de identificar familias génicas remotas entre genomas que potencialmente puedan pertenecer a especies distintas dentro de un mismo taxón. Visualizamos una estrategia de reconstrucción de pangenomas divergentes basada en reducción de redundancia, identificación de homología remota mediante búsqueda de dominios proteicos y construcción de arquitecturas de dominios, y diferenciación de parálogos y ortólogos utilizando algoritmos filogenéticos independiente del contexto génico.

Para validar esta estrategia y para proveer una interfaz de post análisis de las reconstrucciones, se desarrollaron otras dos herramientas independientes las cuales fueron publicadas en revistas arbitradas. Si bien `pewit` comenzó a desarrollarse antes, más precisamente durante mi maestría con una versión prematura del mismo, en ésta tesis se presentarán primero estos dos *software* de apoyo en los capítulos 2 y 3 ya que permiten entender el resultado final (capítulo 4), además de que fueron publicadas antes en el tiempo.

En suma, en ésta tesis se presentará el trabajo realizado para generar una herramienta de reconstrucción de pangenomas con foco en los niveles taxonómicos superiores al de especie, y dos ramificaciones que fueron necesarias para validar la estrategia y facilitar el post análisis. Se abordarán áreas de conocimiento diversas tales como genómica evolutiva y filogenética, genómica comparada, microbiología, teoría de la información, estructuras de datos, y algorítmica.

1.8.1. Objetivo general

Desarrollar y evaluar estrategias que permitan reconstruir pangenomas procariotas evolutivamente distantes respecto a herramientas bioinformáticas ya existentes.

Los objetivos específicos se detallan en cada capítulo.

I must not fear. Fear is the mind-killer.
Fear is the little-death that brings total obliteration.
I will face my fear.
I will permit it to pass over me and through me.
And when it has gone past I will turn the inner eye to see its path.
Where the fear has gone
there will be nothing.
Only I will remain.

— ***Litany Against Fear,***
Frank Herbert's «Dune»

Capítulo 2

Implementación de un simulador de pangenomas

Para evaluar la calidad de cualquier clasificador, es necesario contar con un estándar contra cual comparar. La mejor estrategia para generar un estándar es simulando datos bajo cierto modelo, lo cual permite observar cómo se comporta el clasificador bajo distintas selecciones de parámetros. En este capítulo se describe la implementación de un simulador de pangenomas bacterianos, `simurg`, basado en el Modelo de Genes Infinitos[24] que se describió en la sección 1.2.1, el cual luego se usará para evaluar los agrupamientos generados por el algoritmo para reconstrucción de pangenomas que se describe en el capítulo 4. El nombre del paquete, `simurg`, hace referencia a un ave del folclore persa muy parecida al fénix de los griegos. En realidad del paseriforme mitológico simplemente reciclamos el «sim» para indicar que se trata de un **sim**ulador, no hay que darle mucha más vuelta.

2.1. Objetivo específico

Me propuse desarrollar un simulador de pangenomas bacterianos que permita producir pangenomas bajo distintos supuestos (parámetros) tales como tiempo evolutivo, tasa de mutación, y tasas de ganancia y pérdida de genes.

2.2. Métodos e implementación

Se desarrolló un paquete de R[109] que toma como *input* un archivo fasta con secuencias nucleotídicas, y número de *hojas* (secuencias en el árbol), generaciones, tasa de mutación, y tasas de ganancia y pérdida de genes como parámetros, para simular un pangenoma bacteriano bajo los supuestos del Modelo de Genes Infinitos[24], y evolución neutral a tasa constante sobre un número finito de sitios[124] para los cambios nucleotídicos.

Rápidamente, el algoritmo comienza simulando un árbol coalescente al azar con el número de *tips* establecido. Basado en los largos de rama y en las tasas de ganancia y pérdida de genes, se determinan las distribuciones de las frecuencias génicas finales del pangenoma, es decir, la *panmatrix*. Una vez obtenidas las distribuciones génicas del pangenoma, se muestrean secuencias del archivo fasta, y se simula un proceso de mutación aleatorio desde la raíz del árbol hasta el final, respetando las dicotomías del coalescente.

Para simular tanto los procesos de ganancia y pérdida de genes, como el proceso de mutación, se computan el número de eventos esperados en cada segmento del árbol según los modelos IMG y de evolución neutral, y se obtienen valores desviados al azar con una distribución de Poisson[125] a partir de estos. Si bien el proceso de mutación es neutral, se puede modificar una matriz de sustitución para establecer pesos probabilísticos a determinadas sustituciones, que por *default* es al azar excepto que evita generar codones *stop* en el medio de las secuencias.

El programa devuelve un objeto de clase «pangenomeSimulation» del cual se puede obtener la *panmatrix*, una lista de genes por cada grupo de ortólogos, otra lista con las sustituciones para cada gen, en cada rama del árbol, y una matriz de distancias genéticas entre cada par de genes dentro de un grupo de ortólogos. Finalmente devuelve archivos en formato fasta con las secuencias simuladas.

2.3. Resultados

El paquete se encuentra disponible en `https://github.com/iferres/simurg`, y un tutorial explicativo puede verse en `https://github.com/iferres/simurg/wiki`.

2.3.1. Publicación

El trabajo fue publicado en *Bioinformatics*: «simurg: simulate bacterial pangenomes in R», Ferrés I., Fresia P., e Iraola G., en 2020[126].

Bioinformatics, 36(4), 2020, 1273–1274
doi: 10.1093/bioinformatics/btz735
Advance Access Publication Date: 4 October 2019
Applications Note

Genome analysis

simurg: simulate bacterial pangenomes in R

Ignacio Ferrés[1,*], Pablo Fresia[1,2] and Gregorio Iraola[1,3,4,*]

[1]Microbial Genomics Laboratory, Institut Pasteur Montevideo, [2]Unidad Mixta UMPI, Institut Pasteur Montevideo + INIA, Montevideo 11400, Uruguay, [3]Center for Integrative Biology, Universidad Mayor, Santiago de Chile 7510041, Chile and [4]Wellcome Sanger Institute, Hinxton CB10 1SA, UK

*To whom correspondence should be addressed.

Associate Editor: Alfonso Valencia

Received on March 19, 2019; revised on August 6, 2019; editorial decision on September 19, 2019; accepted on September 25, 2019

Abstract

Motivation: The pangenome concept describes genetic variability as the union of genes shared in a set of genomes and constitutes the current paradigm for comparative analysis of bacterial populations. However, there is a lack of tools to simulate pangenome variability and structure using defined evolutionary models.

Results: We developed simurg, an R package that allows to simulate bacterial pangenomes using different combinations of evolutionary constraints such as gene gain, gene loss and mutation rates. Our tool allows the straightforward and reproducible simulation of bacterial pangenomes using real sequence data, providing a valuable tool for benchmarking of pangenome software or comparing evolutionary hypotheses.

Availability and implementation: The simurg package is released under the GPL-3 license, and is freely available for download from GitHub (https://github.com/iferres/simurg).

Contact: iferres@pasteur.edu.uy or giraola@pasteur.edu.uy

Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

The bacterial pangenome concept was coined by Tettelin *et al.* (2005) to describe the union of genes shared by a certain set of genomes (Vernikos *et al.*, 2015). This notion was then formalized by the distributed genome hypothesis, which states that the gene pool in a bacterial population is more complex and larger than that found in the genome of any single strain. Today, the infinitely many genes (IMG) model (Baumdicker *et al.*, 2012; Collins and Higgs, 2012) provides a comprehensive theoretical framework for outlining the distributed genome dynamics. The IMG is quantitative and can describe gene gain (e.g. by horizontal gene transfer), gene loss (e.g. by pseudogenization or deletion) and point mutation, providing a flexible model for bacterial pangenomes. The extensive comparative analysis of bacterial pangenomes is today a standard approach to gain insight on the evolutionary forces shaping virulence, host-adaptation, transmission or the acquisition of antibiotic resistance. Concomitantly, this has demanded the constant development and improvement of software tools to reconstruct pangenomes (Ding *et al.*, 2017; Page *et al.*, 2015; Snipen and Liland, 2015; Thibeaux *et al.*, 2018). However, the conclusions derived from results obtained with these tools are frequently raised without comparing against control cases with known evolutionary trajectories. This is partially caused by the unavailability of straightforward software tools that allow to simulate bacterial pangenomes under different evolutionary scenarios. Here, we developed simurg to fill this gap, providing a flexible tool for modeling pangenome dynamics.

2 Description

simurg is an R package that depends on ape (Paradis *et al.*, 2004) and imports functions from phangorn (Schliep, 2011) and seqinr (Charif and Lobry, 2007) for sequence and tree manipulation, reshape2 (Wickham, 2014) for data formatting and uses R base, stats and utils to execute probabilistic and stochastic processes. The simurg workflow consists in sampling a set of known gene sequences from a reference FASTA file and, based on a model of neutral evolution guided by a random coalescent tree, it evolves every sequence during a fixed number of generations and under defined rates of gene loss, gene gain and mutation (main model parameters are detailed in Supplementary Table S1). In addition, the model can be customized by modifying the substitution matrix, which is by default a matrix with equal probabilities for transitions between codons except by a constrain that prevents the introduction of stop codons and to jump between codons that differ in more than a single substitution.

Some other tools have been previously developed to simulate bacterial pangenomes. For example, SimBac (Brown *et al.*, 2016) and msPro (Akita *et al.*, 2018) are focused on simulation of both intra-genomic and inter-genomic recombinations along a coalescent tree, while simurg does not account for recombination in its current version. Alternatively, simurg allows to control absolute gene gain and loss rates independently of the underlying molecular mechanism, and is focused on the evolution of coding sequences following the IMG model.

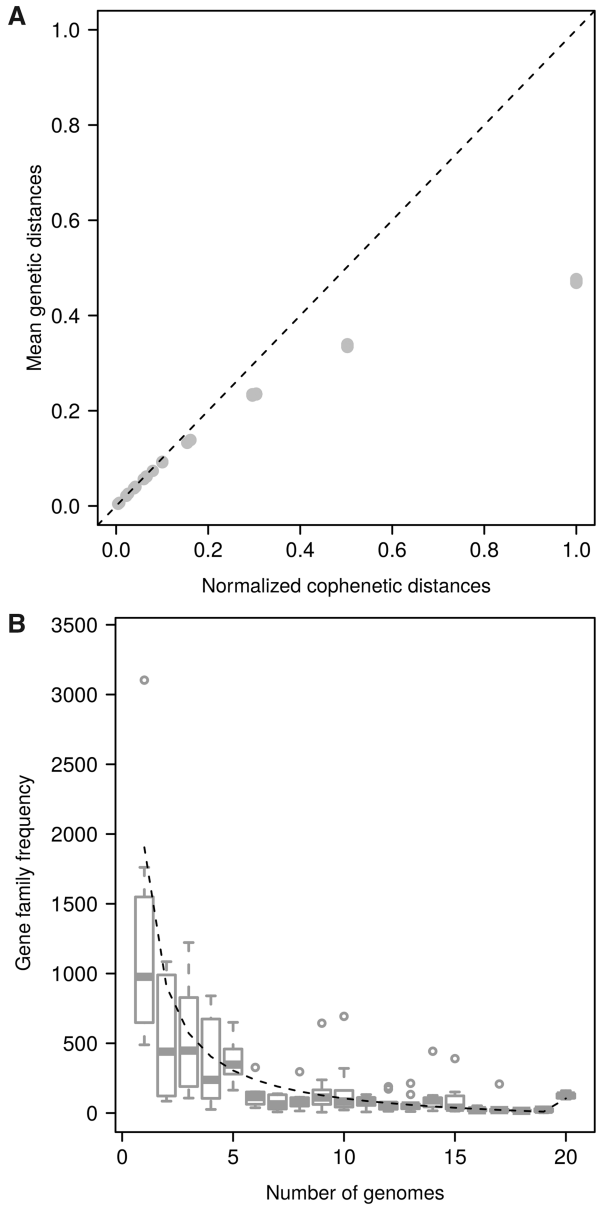


Fig. 1. Simulations performed with simurg. (**A**) Cophenetic and mean genetic distances between each pairs of genomes. Dashed line shows what would be expected under the infinitely many sites model, gray dots evidence saturation of substitutions on finite number of sites. (**B**) Gene family frequency spectrum box plot (gray) for 10 simulations. Black line shows the expected frequencies under the IMG model

3 Results

We used a previously published pangenome dataset (Iraola *et al.*, 2017) as input to simurg to perform simulations. Using default parameters, we simulated the pangenome of 20 organisms and measured the mean genetic distance and the normalized cophenetic distance (phylogenetic distance) between each pair of genomes, which is shown in Figure 1A. Saturation in the mean genetic distance is observed between pairs of genomes with high cophenetic distance, as expected under the Jukes–Cantor model and finite number of sites (Felsenstein, 2004). Also, the approximation to the IMG model was tested by simulating 10 pangenomes of 20 organisms each, with default parameters, and computing the gene family frequency spectrum. Figure 1B shows a box plot summarizing the observed gene family frequency spectra and the expected values under the IMG model using previously reported implementations (Collins and Higgs, 2012). Scripts used to perform these analyses are available at https://github.com/iferres/pangenome_simulation_figures.

4 Conclusion

We have developed simurg, an R package to simulate bacterial pangenomes using the IMG model of evolution. This constitutes the first package in R with this purpose, leveraging this widely used computation environment for benchmarking of pangenome reconstruction tools and testing evolutionary hypotheses on bacterial populations. Future developments will be focused on the incorporation of more complex evolutionary models, including recombination and non-neutral processes.

Acknowledgements

We thank Hugo Naya for providing lab space and computational resources at the Bioinformatics Unit (Institut Pasteur de Montevideo).

Funding

This work was partially funded by the Agencia Nacional de Investigación e Innovación (ANII) [POS_NAC_2018_1_151494 to I.F.].

Conflict of Interest: none declared.

References

Akita,T. *et al.* (2018) Coalescent framework for prokaryotes undergoing interspecific homologous recombination. *Heredity*, **120**, 474–484.
Baumdicker,F. *et al.* (2012) The infinitely many genes model for the distributed genome of bacteria. *Genome Biol. Evol.*, **4**, 443–456.
Brown,T. *et al.* (2016) SimBac: simulation of whole bacterial genomes with homologous recombination. *Microb. Genomics*, **2**, e000044.
Charif,D. and Lobry,J.R. (2007) SeqinR 1.0-2: a contributed package to the R project for statistical computing devoted to biological sequences retrieval and analysis. In: Bastolla,U. *et al.* (eds.) *Structural Approaches to Sequence Evolution*. Springer, Berlin, Germany, pp. 207–232.
Collins,R.E. and Higgs,P.G. (2012) Testing the infinitely many genes model for the evolution of the bacterial core genome and pangenome. *Mol. Biol. Evol.*, **29**, 3413–3425.
Ding,W. *et al.* (2017) panX: pan-genome analysis and exploration. *Nucleic Acids Res.*, **46**, e5.
Felsenstein,J. (2004) Inferring phylogenies. Chapter 11: distance matrix methods. In: Felsenstein,J. (ed.) *The Jukes–Cantor Model—An Example*, p.156. Sinauer Associates, Sunderland, MA.
Iraola,G. *et al.* (2017) Distinct *Campylobacter fetus* lineages adapted as livestock pathogens and human pathobionts in the intestinal microbiota. *Nat. Commun.*, **8**, 1367.
Page,A.J. *et al.* (2015) Roary: rapid large-scale prokaryote pan genome analysis. *Bioinformatics*, **31**, 3691–3693.
Paradis,E. *et al.* (2004) Ape: analyses of phylogenetics and evolution in R language. *Bioinformatics*, **20**, 289–290.
Schliep,K.P. (2011) phangorn: phylogenetic analysis in R. *Bioinformatics*, **27**, 592–593.
Snipen,L. and Liland,K.H. (2015) Micropan: an R-package for microbial pan-genomics. *BMC Bioinform.*, **16**, 79.
Tettelin,H. *et al.* (2005) Genome analysis of multiple pathogenic isolates of *Streptococcus agalactiae*: implications for the microbial "pan-genome". *Proc. Natl. Acad. Sci. USA*, **102**, 13950–13955.
Thibeaux,R. *et al.* (2018) Deciphering the unexplored *Leptospira* diversity from soils uncovers genomic evolution to virulence. *Microb. Genomics*, **4**, e000144.
Vernikos,G. *et al.* (2015) Ten years of pan-genome analyses. *Curr. Opin. Microbiol.*, **23**, 148–154.
Wickham,H. (2014) Tidy data. *J. Stat. Softw.*, **59**, 1–23.

Is this the real life?
Is this just fantasy?

— ***Freddie Mercury***

Capítulo 3

Generación de una interfaz para analizar datos de pangenomas

Si bien el objetivo central de esta tesis es el desarrollo de un software para la reconstrucción de pangenomas, igualmente importante es proveer a los usuarios de una interfaz para analizar y consultar esta reconstrucción. Para este paso, que llamamos post-análisis, decidí implementar una utilidad que simplifica la manipulación y el análisis de los datos pangenómicos. Si bien en un principio formó parte del *software* que presento en el capítulo 4, realicé algunas generalizaciones que permitieron que el proyecto tenga «vida propia», por lo que decidimos trabajarlo y publicarlo aparte.

Básicamente pensé a los pangenomas como una estructura relacional de datos formado por tres «tablas»: en los pangenomas hay genes, que pertenecen a organismos, y que son asignados a *clusters* de genes. Bajo esta idea desarrollé `pagoo`, cuya implementación consiste en encapsular los datos y los métodos para analizar esos datos dentro de un mismo objeto, y que es capaz de cargar pangenomas reconstruidos mediante otros *softwares* en la misma estructura estándar.

El nombre es una combinación de las palabras ***pa****n****g****enome* y ***O****bject* ***O****riented* ya que usé un paradigma de orientación a objetos encapsulados relativamente nuevo en el ecosistema R, llamado R6. Encontramos que existe un libro de cuentos infantiles estadounidense sobre un cangrejo ermitaño llamdo *Pagoo*[127], escrito por Holling C. Holling en 1961, y pensamos que la naturaleza encapsulada de los pangenomas se parece al cangrejo que se «encapsula» en conchas de moluscos.

3.1. Objetivo específico

Simplificar el post-análisis de pangenomas bacterianos mediante la implementación de una estructura de datos estándar con métodos asociados, que sea fácilmente consultable, y que pueda usarse de forma interactiva en concierto con otros paquetes del ecosistema de R.

3.2. Métodos e implementación

El paquete de R `pagoo` utiliza el motor R6, que sigue un paradigma de orientación a objetos encapsulados, a diferencia de la mayor parte del ecosistema R donde se sigue un paradigma de orientación a objetos funcional. En el primer caso, los datos y métodos pertenecen a los objetos, mientras que en el segundo caso los métodos están separados de los datos, y pertenecen a clases. Para el caso de una estructura de datos compleja, como un pangenoma bacteriano, una clase orientada a objetos encapsulada presenta algunas ventajas respecto a la otra ya que permite mutabilidad, autorreferencias, y simplicidad en las consultas.

Se implementaron tres clases anidadas tal que las clases más complejas heredan las propiedades de las clases más simples. Esto permite extender fácilmente las clases si en un futuro alguien quisiera crear clases más complejas a partir de las básicas implementadas en `pagoo`. Un ejemplo de esto último se verá en el capítulo 4.

La estructura de datos elegida para representar un pangenoma consiste en una estructura relacional donde los genes individuales poseen una columna clave que identifica al organismo de donde proceden, y otra que identifica al *cluster* al cual fueron asignadas por el método de reconstrucción de pangenomas. Cada tabla puede contener otras tantas columnas como metadatos se obtengan para cada observación. Adicionalmente se puede agregar otra «tabla» con las secuencias nucleotídicas de cada gen, de modo que la consulta y manipulación de las secuencias en el contexto de un pangenoma se vuelve relativamente simple.

Dado que el objeto es mutable, se pueden cambiar valores que modifican el comportamiento del objeto. Por ejemplo, de forma dinámica se puede cambiar el umbral de inclusión para que un *cluster* se considere *core* o no, y ello tiene un efecto sobre todo el objeto de modo que los métodos devolverán resultados acordes al nuevo umbral. Otro ejemplo de esto es la capacidad de esconder o volver a mostrar algunos organismos del *dataset*: al ocultar uno o más organismos, los genes de dicho organismo dejan de estar disponibles y los métodos modifican su resultado de acuerdo al estado del objeto.

Las clases implementadas en `pagoo` permiten fácilmente generar visualizaciones y calcular estadísticos que describen diversas propiedades de los pangenomas. Algo también destacable es la posibilidad de desplegar una interfaz *shiny* de usuario donde se pueden ver gráficos básicos y permite la interacción con el objeto de forma no programática, para los usuarios menos experimentados.

Por último cabe mencionar que el ecosistema de R contiene múltiples bibliotecas que pueden interaccionar con los métodos de `pagoo` para generar análisis más complejos.

3.3. Resultados

`pagoo` se encuentra disponible en `https://github.com/iferres/pagoo`, y en CRAN `https://CRAN.R-project.org/package=pagoo`.

Igualmente importante para la valoración de este trabajo son los tutoriales web disponibles en `https://iferres.github.io/pagoo/`. En el menú «*Articles*» se pueden seguir tutoriales para:

1. Crear objetos de las clases implementadas: `https://iferres.github.io/pagoo/articles/Input.html`
2. Consultar campos del objeto: `https://iferres.github.io/pagoo/articles/Querying_Data.html`
3. Hacer *subsets* del objeto: `https://iferres.github.io/pagoo/articles/Subseting.html`
4. Hacer gráficos y calcular estadísticos: `https://iferres.github.io/pagoo/articles/Methods_Plots.html`
5. Utilizar otros paquetes de R en concierto con `pagoo`, lo cual llamamos «recetas»: `https://iferres.github.io/pagoo/articles/Recipes.html`

3.3.1. Publicación

El trabajo fue publicado en *Cell Reports Methods*: «An object-oriented framework for evolutionary pangenome analysis», Ferrés I., e Iraola G., en 2021[128].

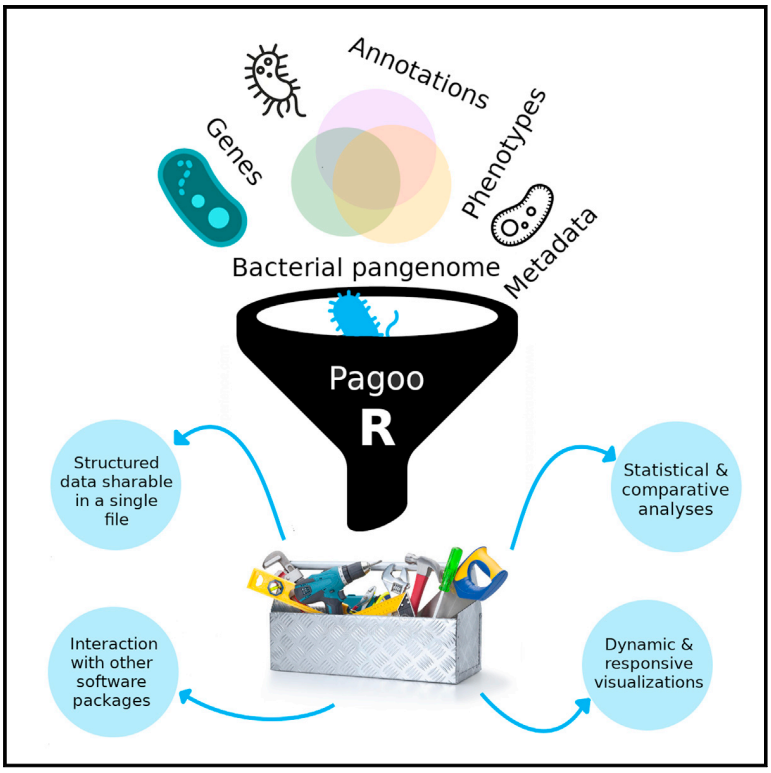
Un protocolo complementario para analizar pangenomas bacterianos utilizando `pagoo` fue publicado en *STAR Protocols*: «Protocol for post-processing of bacterial pangenome data using Pagoo pipeline», Ferrés I., e Iraola G., en 2021[129], ver anexo A.

Cell Reports
Methods

Report

An object-oriented framework for evolutionary pangenome analysis

Graphical abstract



Authors

Ignacio Ferrés, Gregorio Iraola

Correspondence

iferres@pasteur.edu.uy (I.F.),
giraola@pasteur.edu.uy (G.I.)

In brief

Ferrés and Iraola develop a software package in R to integrate and analyze bacterial pangenome data. This allows creating a single programming object that stores all the data, as well as methods to produce standard analyses and visualizations for comparative genomics.

Highlights

- Pagoo is a software framework to analyze bacterial pangenomes
- It can be used with the output of any pangenome reconstruction software
- Genetic, phenotypic, and other metadata are integrated in a single object or file
- Pagoo is extensible and interacts easily with other microbial genomics packages

Ferrés & Iraola, 2021, Cell Reports Methods *1*, 100085
September 27, 2021
https://doi.org/10.1016/j.crmeth.2021.100085

CellPress

Cell Reports Methods

CellPress
OPEN ACCESS

Report

An object-oriented framework for evolutionary pangenome analysis

Ignacio Ferrés[1,2,*] and Gregorio Iraola[1,2,3,4,5,*]
[1]Microbial Genomics Laboratory, Institut Pasteur Montevideo, Montevideo, Uruguay
[2]Center for Innovation in Epidemiological Surveillance, Institut Pasteur Montevideo, Montevideo, Uruguay
[3]Wellcome Sanger Institute, Hinxton, UK
[4]Center for Integrative Biology, Universidad Mayor, Santiago de Chile, Chile
[5]Lead contact
*Correspondence: iferres@pasteur.edu.uy (I.F.), giraola@pasteur.edu.uy (G.I.)
https://doi.org/10.1016/j.crmeth.2021.100085

MOTIVATION Despite the extensive availability of software for bacterial pangenome reconstruction, current tools for pangenome data analysis do not provide end-to-end solutions because, once visualizations are generated, users cannot use the same framework to return to the data and perform further comparisons and more complex analyses using standardized methods. To fill this gap, we developed Pagoo, a flexible and extensible but standardized framework that allows performing integrative pangenome analysis in a simple and reproducible way.

SUMMARY

Pangenome analysis is fundamental to explore molecular evolution occurring in bacterial populations. Here, we introduce Pagoo, an R framework that enables straightforward handling of pangenome data. The encapsulated nature of Pagoo allows the storage of complex molecular and phenotypic information using an object-oriented approach. This facilitates to go back and forward to the data using a single programming environment and saving any stage of analysis (including the raw data) in a single file, making it sharable and reproducible. Pagoo provides tools to query, subset, compare, visualize, and perform statistical analyses, in concert with other microbial genomics packages available in the R ecosystem. As working examples, we used 1,000 *Escherichia coli* genomes to show that Pagoo is scalable, and a global dataset of *Campylobacter fetus* genomes to identify evolutionary patterns and genomic markers of host-adaptation in this pathogen.

INTRODUCTION

The exponentially growing number of diverse bacterial genomes has prompted pangenome reconstruction as a gold standard to uncover molecular evolution of bacterial populations (Tettelin et al., 2005; Vernikos et al., 2015). This is because of the high intraspecific diversity observed in bacterial genomes, which are affected by horizontal gene transfer, variations in effective population size, and constant colonization of new niches. As these forces are influential in determining pangenome size and structure (McInerney et al., 2017), pangenome comparisons can reveal genome evolutionary dynamics associated with important biological processes, such as speciation, host adaptation, pathogenicity, or the acquisition of antimicrobial resistance. Pangenome reconstruction is typically performed from genes annotated in a set of whole-genome sequences. In general, coding sequences of different strains are grouped in orthologous clusters based on different similarity criteria. Then, pangenome data inform about the belonging of each gene encoded in each genome to a certain orthologous cluster. In recent years, several software tools have been developed to reconstruct bacterial pangenomes, such as Roary (Page et al., 2015), panX (Ding et al., 2018), PanOCT (Fouts et al., 2012), PIRATE (Bayliss et al., 2019), PEPPAN (Zhou et al., 2020), or Panaroo (Tonkin-Hill et al., 2020). These tools focus on automation of steps, improvement of clustering algorithms, and optimization of computational costs to process thousands of sequences of increasingly large genomic datasets. Many of these software include data visualization modules along with other specific software that have been developed with this purpose, including Phandango (Hadfield et al., 2018) or PanViz (Pedersen et al., 2017). However, current tools do not provide end-to-end solutions for customized pangenome data analysis, since, once visualizations are generated, users cannot use the same framework to return to the data and perform further comparisons and more complex analyses using standardized methods.

Here, we introduce Pagoo, a pangenome post-processing tool that can take the output produced by pangenome reconstruction software tools providing a standardized framework for its analysis. Pagoo is based on an object-oriented design built on a class system in R (R Core team, 2017), which implements: (1) an integrative data structure for standardized storage of pangenome information, such as orthologous clusters, sequences, annotations, and metadata, in a single object (shareable through a single file); (2) a set of straightforward methods for responsive querying, handling, and subsetting of this data structure; and (3) a set of standard statistics and active visualizations leveraging flexible downstream comparative analyses. Along with extensive documentation, we show how Pagoo interacts with other widely used microbial genomics tools and the R ecosystem for improved analysis of molecular evolution in bacterial populations.

New approaches

Description of the software

A pangenome can be represented as individual genes that belong to organisms (genomes), which are then assigned to a cluster of orthologous genes. Pagoo stores this as a three-column matrix, with one column identifying an individual gene, the next one identifying the organism that this gene belongs to, and the last one identifying the orthologous cluster that the gene was assigned by the pangenome reconstruction method. Optionally, this matrix can contain additional columns as gene-specific metadata, such as annotations, functional assignments, or genomic coordinates. Orthologous clusters and organisms can also take metadata represented as two different matrices, with the condition that each one must contain a column that correctly maps each observation (cluster or organism) into the former matrix. Gene sequences can also be added to this structure, with the condition that their names must also map to rows in the first matrix (Figure 1A). This relational structure optimizes data storage avoiding duplication, enables flexibility to work with different types of metadata, and facilitates complex querying and analysis.

A salient and unique feature of Pagoo is that these data structures are stored and managed in an encapsulated, object-oriented fashion using the R6 package as backend. In contrast with traditional R programming, the R6 paradigm considers that methods belong to objects rather than to generic functions, so an object contains both the data and embedded methods to analyze it. In this context, the Pagoo object is built on three R6 classes. PgR6 is the most basic class that contains methods and functions for data handling and subsetting. Then, PgR6M inherits all the methods and fields from PgR6 and incorporates statistical methods and visualization tools based on the ggplot2 package (Wickham, 2011). PgR6MS inherits all capabilities from the others and adds methods for manipulation of biological sequences using the Biostrings package (Figure 1B). These classes support the main data types that typically represent a pangenome, providing a synergistic framework to manage both the raw data and methods to perform operations and explore results with customized visualizations. Moreover, any of these classes could be further inherited and easily extended by third-party applications.

Another remarkable feature of Pagoo is that raw data stored in the pangenome object are kept unaltered in the background, while users can query, mutate, or subset the object using active bindings. This allows changing the state of the object without altering the original data. For example, users can temporarily hide certain organisms from the dataset, actively set thresholds that change the definition of core genes, or extract specific information from organisms, genes, clusters, or sequences. Class-specific methods for generic subset operators are also implemented, enabling extraction of relevant fields straight from the object by using standard R subset notation.

Pagoo provides specific methods for generating the pangenome object from scratch by formatting output files from any pangenome reconstruction software. In addition, Pagoo provides specific functions to automatically generate the pangenome object from output files produced by Roary and Panaroo. Other popular pangenome reconstruction software, such as PIRATE and PEPPAN, have functions to transform their outputs into Roary's output format, making them compatible with Pagoo. Then, pangenome can be saved with any changes to the object along with the unaltered original data as a single file. Pagoo has been built and tested in all three major operative systems (Linux, Windows, and Mac). A detailed explanation of each method and operator for data input, saving, and loading, and for specific comparative analyses, is provided in the online user manual (GitHub: https://iferres.github.io/pagoo/). Together, this implementation represents a conceptual advance for pangenome data handling, facilitating reproducibility, and enabling multiple and flexible analyses.

Data analysis and visualization

Pagoo includes statistical and visualization methods. Customized plots and statistical analyses can be generated directly from the pangenome object using active bindings on the console, or by deploying a built-in R-Shiny application. This interactive application is divided into two main components: (1) a general dashboard that interactively displays summary statistics, including number of organisms, orthologous clusters and genes, core and accessory genome sizes, gene frequency barplots, pangenome curves, and scrollable information about core genome clusters and genes (i.e., annotation or any other metadata); and (2) a specific dashboard showing clustering of genomes according to accessory gene distances and principal-component analysis (PCA), genome-specific accessory genome sizes, visualization of gene presence/absence matrix with associated metadata, and information about accessory gene clusters (Figure S1). This interactive application allows responsive exploration of evolutionary trends in bacterial populations, guiding downstream analyses on the console that can be performed over the same pangenome object using methods provided by Pagoo or leveraging the interaction with other R tools.

Creating recipes for more complex analyses

Remarkably, more complex comparative pangenome analyses can be performed by applying concise code recipes. We define recipes as relatively short snippets that pipe pangenome information extracted from the object as input to other R tools. We have developed example recipes (available in the online user manual at GitHub: https://iferres.github.io/pagoo/articles/6-Recipes.html) to build core genome phylogenies, identify population structure, explore genome-wide selective pressures acting over the core genes, and compare individual gene

CellPress
OPEN ACCESS

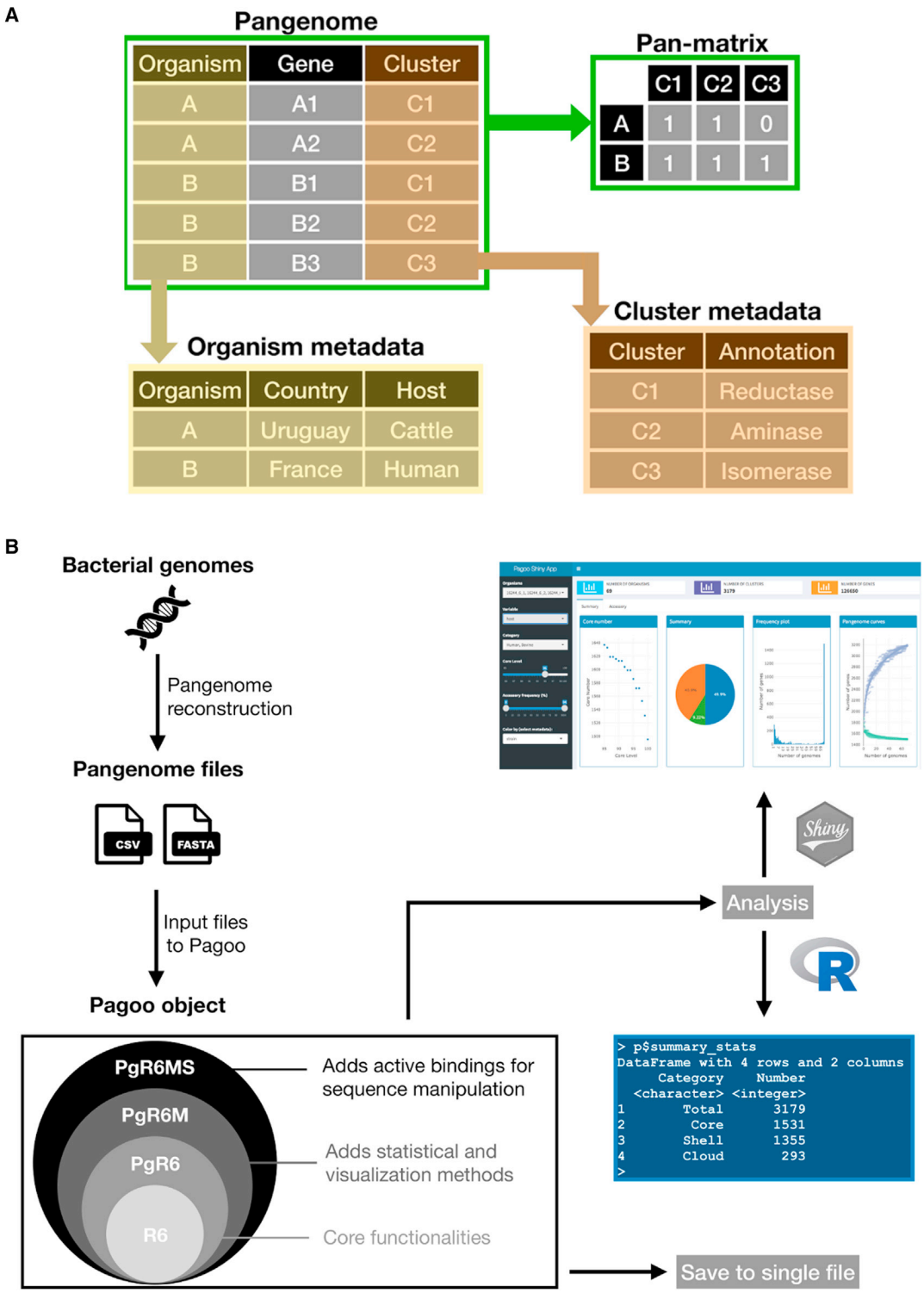


Figure 1. Framework and overall design of Pagoo

(A) Example of the relational structure implemented to store, link, and operate over different pangenome data types.

(B) General description of the workflow from assembled genomes to Pagoo analysis. Once pangenome files are created with any available pangenome reconstruction software, these files can be loaded to create the Pagoo object. The specific R6 classes store and manage different data types that can store all the information in a single file or perform comparative analyses using the R console interface or the Pagoo Shiny application.

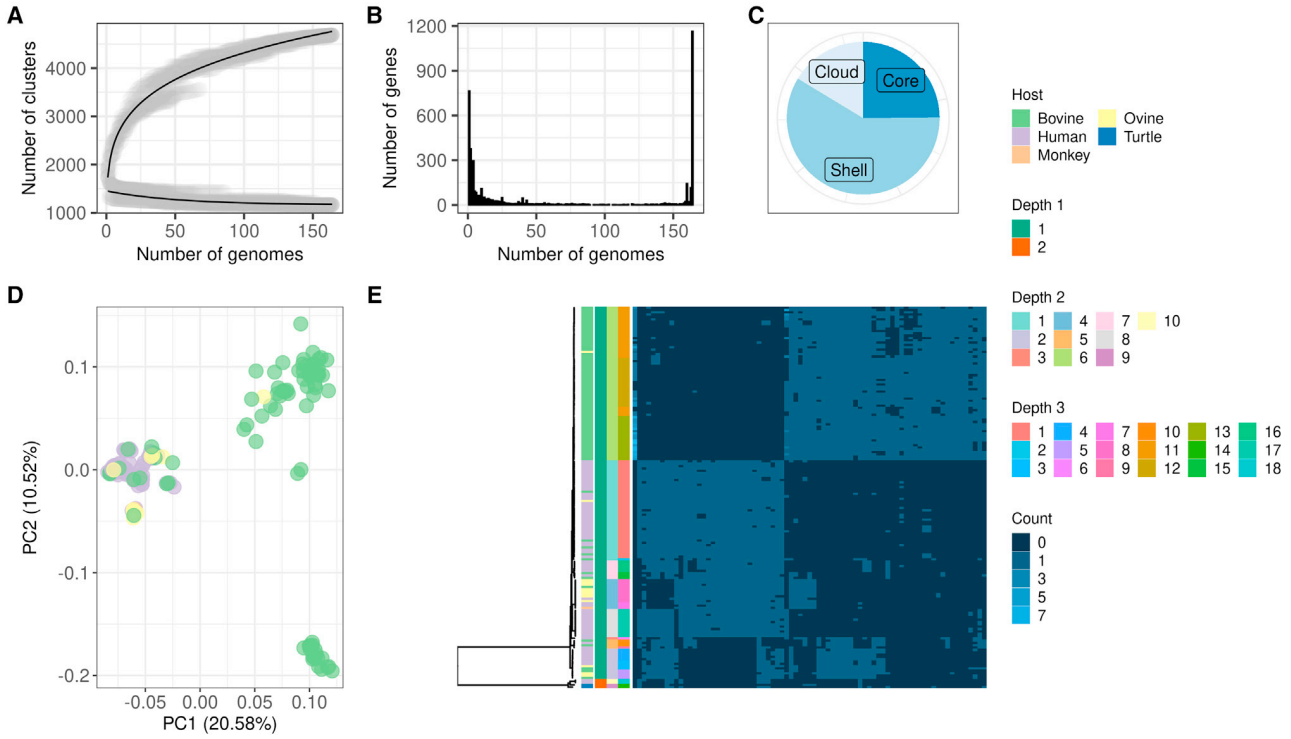


Figure 2. Results extracted from the pangenome object

Exploration of the *C. fetus* pangenome using information directly extracted from the pangenome object and customized esthetics.

(A) Pangenome and core genome curves with gray circles representing different sub-samples at increasing numbers of genomes; the black lines show the fitting to the power law and exponential decay functions, respectively.

(B) The distribution of genes in different subsets of genomes.

(C) The distribution of the pangenome in core genes and accessory genes (shell and cloud genes).

(D) A principal components analysis generated from the gene presence/absence matrix whose first principal component (PC1) clearly distinguishes between two groups of genomes, one of mostly bovine-derived strains (right), and the other with various hosts (left).

(E) A heatmap (right) with gene count (paralogs) abundance (in columns) and per organisms in (rows, associated to a phylogenetic tree [left]) inferred from a concatenated alignment of core genes. Between the tree and the heatmap, the color bars indicate, from left to right, the host associated to each isolate, and three different depth levels of subpopulations inferred by the Rhierbaps package.

sequences against specific databases. Importantly, the development and implementation of recipes enable full reproducibility of publication-quality figures generated directly from the pangenome object. Despite the R ecosystem currently including dozens of packages that allow performing diverse and complex comparative analyses, it is possible that some specific tasks could be difficult to implement in R. In this sense, Pagoo recipes can be designed to interact with external tools to generate results outside the R session and integrate them into the pangenome object using standard R data input functions.

RESULTS

As a working example, we used a previously published study on the genomic evolution of *Campylobacter fetus* (Iraola et al., 2017), a zoonotic pathogen that presents a strong population structure with different lineages adapted to livestock or humans. In brief, we used Pagoo to analyze a pangenome reconstructed from 164 *C. fetus* genomes (Figure 2). Using simple and readable R code we were able to recover the main diversity trends reported for this species, such as a marked difference in accessory gene patterns between livestock- and human-adapted lineages. Specifically, we identified a set of 78 highly discriminant accessory genes that can differentiate between these lineages and could be used to develop molecular typing tools (Table S1). In addition, automatic extraction of core genes from the Pagoo object enabled robust phylogenomic analyses in interaction with other R packages, such as DECIPHER (Wright, 2015), for multiple sequence alignment, phangorn (Schliep, 2011) for phylogenetic reconstruction, and Rhierbaps (Tonkin-Hill et al., 2018) for population structure inference. This analysis revealed the same population structure composed by eight main *C. fetus* lineages as reported in the original study (Iraola et al., 2017). Then, we used Pagoo to compare phylogenies built from every single core gene against the core genome phylogeny. This allowed us to rank the core genes based on their goodness to recover the population structure, *mcp4* being the one with the closest distance (Figure S2A). This enables the future development of high-resolution *C. fetus* typing methods using amplicon sequencing of single core genes. Also, the PCA based on the accessory genes allowed to recover previously

observed patterns of variation separating bovine-adapted *C. fetus* from other hosts, including humans (discrimination given by PC1 in observed in Figure 2D). Using this approach we also detected a separation between bovine-adapted genomes given by PC2 (Figure 2D). Further exploration of this demonstrated a specific group of genomes from Spain with particular accessory gene patterns, suggesting a previously unnoticed geographic structuring of this pathogen.

An additional dataset consisting of hundreds of multi-drug resistant *E. coli* genomes (Decano and Downing, 2019) was used to test scalability by measuring the time it takes Pagoo to perform certain operations. First, Pagoo was able to upload the output from Roary, including gene sequences automatically extracted from GFF3 annotation files, and automatically build its relational structure for 500 genomes in ~20 min. This time does not scale linearly because it depends on the number and size of gene clusters, but was completed in reasonable time (<2 h) for 1,000 genomes. Second, once the Pagoo object is created, information can be queried on the fly in matter of seconds or minutes. For example, one single operation can extract all core genes from 1,000 genomes in ~2 min. Also, visualizations such as gene distribution plots can be rendered in seconds just using single operations (Figure S2B).

DISCUSSION

The advent of high-throughput sequencing technologies more than 15 years ago pushed microbiology toward the field of comparative genomics, which rapidly transitioned from studies including few to thousands of genomes (Vernikos et al., 2015). This substantially increased the complexity of datasets, requiring new approaches to systematically handle and track different components of interrelated pangenomic data. Pagoo introduces a framework underpinned in a concept that leverages the simplicity of storing all the information in a standardized and reproducible manner in a single, shareable object, as well as providing specific methods to query and analyze it. The implemented classes can be easily inherited and extended, so users can eventually incorporate methods able to cope with any kind of metadata added to the pangenomes, such as genomic coordinates or structural information. Pagoo not only allows to perform basic exploratory analysis through the Shiny application, but also facilitates expert bioinformatic analysis to deliver more complex results through the R console ecosystem working in concert with other population genomic packages. The advantage of using an interpreted language for post-processing of pangenomic data, like R, in contrast to compiled tools that run from the terminal interface, is that users can go back and forward to the data, producing both general and fine-grained analyses in a single environment. Pagoo's encapsulated and object-oriented nature brings simplicity and intuition to the experience of working with these kinds of complex relational data, in combination with the active-binding feature of R6 classes, which opens the possibility of changing the state of the object as a whole and its behavior. On the contrary, the classic functional object-oriented R programming works well for simpler data structures, but falls short and becomes inconvenient when dealing with complex and potentially mutable structures, which pangenomes are. Overall, Pagoo's design aims to improve and facilitate current practices on the analysis of molecular evolution in bacterial populations.

Limitations of the study

Although the R ecosystem includes dozens of packages to perform diverse comparative genomic analyses, it is possible that some specific tasks are difficult to implement in R or are better covered by other pangenome analysis tools. Also, Pagoo has been tested with hundreds to thousands of genomes, but a potential limitation is the time it can take to generate Pagoo objects from extremely big datasets. In particular, Pagoo's Shiny application for dynamic visualization has been designed to work with small to medium size datasets.

STAR★METHODS

Detailed methods are provided in the online version of this paper and include the following:

- KEY RESOURCES TABLE
- RESOURCE AVAILABILITY
  - Lead contact
  - Materials availability
  - Data and code availability
- METHOD DETAILS
  - Re-analysis of *C. fetus* dataset
  - Scalability assessment using *E. coli* genomes
- QUANTIFICATION AND STATISTICAL ANALYSIS

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.crmeth.2021.100085.

ACKNOWLEDGMENTS

We thank Pablo Fresia, Daniela Costa, and Andrés Parada for insightful comments and suggestions during testing of Pagoo. I.F. is funded by grant ANII-POS_NAC_2018_1_151494 from the Agencia Nacional de Investigación e Innovación (ANII), Uruguay. This work has been partially funded by the G4 Research Groups Program from Institut Pasteur Montevideo and Banco de Seguros del Estado (BSE) of Uruguay.

AUTHOR CONTRIBUTIONS

G.I. and I.F. conceived the idea. I.F. developed the software and performed experiments. G.I. and I.F. wrote the manuscript.

DECLARATION OF INTERESTS

The authors declare no competing interests.

Received: March 9, 2021
Revised: June 4, 2021
Accepted: August 25, 2021
Published: September 27, 2021

REFERENCES

Bayliss, S.C., Thorpe, H.A., Coyle, N.M., Sheppard, S.K., and Feil, E.J. (2019). PIRATE: a fast and scalable pangenomics toolbox for clustering diverged orthologues in bacteria. GigaScience *8*, 1–9.

Decano, A.G., and Downing, T. (2019). An *Escherichia coli* ST131 pangenome atlas reveals population structure and evolution across 4,071 isolates. Sci. Rep. *9*, 17394.

Ding, W., Baumdicker, F., and Neher, R.A. (2018). panX: pan-genome analysis and exploration. Nucleic Acids Res. *46*, e5.

Fouts, D.E., Brinkac, L., Beck, E., Inman, J., and Sutton, G. (2012). PanOCT: automated clustering of orthologs using conserved gene neighborhood for pan-genomic analysis of bacterial strains and closely related species. Nucleic Acids Res. *40*, e172.

Hadfield, J., Croucher, N.J., Goater, R.J., Abudahab, K., Aanensen, D.M., and Harris, S.R. (2018). Phandango: an interactive viewer for bacterial population genomics. Bioinformatics *34*, 292–293.

Iraola, G., Forster, S.C., Kumar, N., Lehours, P., Bekal, S., García-Peña, F.J., Paolicchi, F., Morsella, C., Hotzel, H., Hsueh, P.-R., et al. (2017). Distinct *Campylobacter fetus* lineages adapted as livestock pathogens and human pathobionts in the intestinal microbiota. Nat. Commun. *8*, 1367.

Kurtzer, G.M., Sochat, V., and Bauer, M.W. (2017). Singularity: scientific containers for mobility of compute. PLoS One *12*, e0177459.

McInerney, J.O., McNally, A., and O'Connell, M.J. (2017). Why prokaryotes have pangenomes. Nat. Microbiol. *2*, 1–5.

Page, A.J., Cummins, C.A., Hunt, M., Wong, V.K., Reuter, S., Holden, M.T.G., Fookes, M., Falush, D., Keane, J.A., and Parkhill, J. (2015). Roary: rapid large-scale prokaryote pan genome analysis. Bioinformatics *31*, 3691–3693.

Paradis, E., and Schliep, K. (2019). Ape 5.0: an environment for modern phylogenetics and evolutionary analyses in R. Bioinformatics *35*, 526–528.

Pedersen, T.L., Nookaew, I., Wayne Ussery, D., and Månsson, M. (2017). PanViz: interactive visualization of the structure of functionally annotated pangenomes. Bioinformatics *33*, 1081–1082.

R Core team (2017). R: A Languague and Environment for Statistical Computing.

Schliep, K.P. (2011). phangorn: phylogenetic analysis in R. Bioinformatics *27*, 592–593.

Seemann, T. (2014). Prokka: rapid prokaryotic genome annotation. Bioinformatics *30*, 2068–2069.

Sochat, V.V., Prybol, C.J., and Kurtzer, G.M. (2017). Enhancing reproducibility in scientific computing: metrics and registry for singularity containers. PLoS One *12*, e0188511.

Tettelin, H., Masignani, V., Cieslewicz, M.J., Donati, C., Medini, D., Ward, N.L., Angiuoli, S.V., Crabtree, J., Jones, A.L., Durkin, A.S., et al. (2005). Genome analysis of multiple pathogenic isolates of *Streptococcus agalactiae*: implications for the microbial "pan-genome". Proc. Natl. Acad. Sci. U.S.A. *102*, 13950–13955.

Tonkin-Hill, G., Lees, J.A., Bentley, S.D., Frost, S.D.W., and Corander, J. (2018). RhierBAPS: an R implementation of the population clustering algorithm hierBAPS. Wellcome Open Res. *3*, 93. https://doi.org/10.12688/wellcomeopenres.14694.1.

Tonkin-Hill, G., MacAlasdair, N., Ruis, C., Weimann, A., Horesh, G., Lees, J.A., Gladstone, R.A., Lo, S., Beaudoin, C., Floto, R.A., et al. (2020). Producing polished prokaryotic pangenomes with the Panaroo pipeline. Genome Biol. *21*, 180.

Vernikos, G., Medini, D., Riley, D.R., and Tettelin, H. (2015). Ten years of pan-genome analyses. Curr. Opin. Microbiol. *23*, 148–154.

Wickham, H. (2011). ggplot2. WIREs Comput. Stat. *3*, 180–185.

Wright, E.S. (2015). DECIPHER: harnessing local sequence context to improve protein multiple sequence alignment. BMC Bioinformatics *16*, 322.

Zhou, Z., Charlesworth, J., and Achtman, M. (2020). Accurate reconstruction of bacterial pan- and core genomes with PEPPAN. Genome Res. *30*, 1667–1679.

CellPress
OPEN ACCESS

STAR★METHODS

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
Campylobacter fetus dataset	Iraola et al. (2017)	https://doi.org/10.6084/m9.figshare.13622354.v1
Escherichia coli ST131 dataset	Decano and Downing (2019)	https://doi.org/10.5281/zenodo.3341535
Software and algorithms		
Pagoo	This paper	https://cran.r-project.org/package=pagoo and https://github.com/iferres/pagoo and https://doi.org/10.6084/m9.figshare.15074652.v1
Prokka	Seemann, 2014	https://github.com/tseemann/prokka
ggplot2	Wickham (2016)	cran.r-project.org/package=ggplot2
DECIPHER	Wright (2015)	https://www.bioconductor.org/packages/release/bioc/html/DECIPHER.html
Phangorn	Schliep (2010)	cran.r-project.org/package=phangorn
Rhierbaps	Tonkin-Hill et al., 2018	cran.r-project.org/package=rhierbaps
Ape	Paradis and Schliep (2019)	cran.r-project.org/package=ape
Roary	Page et al. (2015)	https://sanger-pathogens.github.io/Roary/
Scripts to run all the analysis	This paper	github.com/iferres/pagoo_publication_scripts and https://doi.org/10.6084/m9.figshare.15074673.v1
Singularity	Kurtzer et al. (2017)	https://sylabs.io/singularity/
Singularity container hosted at singularity-hub	Sochat et al. (2017)	https://singularity-hub.org/collections/5123

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources should be directed to and will be fulfilled by the lead contact, Gregorio Iraola (giraola@pasteur.edu.uy).

Materials availability

This study did not generate new unique reagents.

Data and code availability

- Pagoo code has been deposited and is available at the Comprehensive R Archive Network (CRAN). Pagoo's source code is also available at GitHub. We provide a Singularity (Kurtzer et al., 2017) image with all dependencies needed to fully reproduce analyses using the two working dataset (*C. fetus* and *E. coli*), from downloading the data to generating the plots. The Singularity definition file along with the scripts are publicly available at GitHub. The prebuilt Singularity image is hosted at Singularity-hub (Sochat et al., 2017).
- Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

METHOD DETAILS

Re-analysis of *C. fetus* dataset

Genome assemblies were obtained from a previously published study by (Iraola et al., 2017). Genome re-annotation was performed with Prokka (Seemann, 2014) and the pangenome was reconstructed using Roary (Page et al., 2015) with default parameters. Assessment of genome quality revealed that 4 assemblies contained an abnormal number of genes (Figure S2C), so they were hidden from the dataset using Pagoo's 'drop()' function. This function is recommended only with exploratory purposes or when the number

of hidden organisms is small. Otherwise, it is recommended to reconstruct the pangenome by removing undesired genomes from the beginning. Plots describing summary statistics were generated using Pagoo's built-in methods and the ggplot2 package (Wickham, 2011).

Core genes were extracted from the pangenome object using Pagoo's method 'core_seqs_4_phylo', were aligned with DECIPHER (Wright, 2015) and a reference phylogeny was reconstructed from concatenated core gene sequences using the phangorn package (Schliep, 2011). The Rhierbaps package (Tonkin-Hill et al., 2018) was used to infer population structure. Each individual core gene was aligned to reconstruct gene-specific phylogenies as described above. The topological distance between the reference phylogeny and gene-specific phylogenies was calculated with the 'dist.topo' function from the ape package (Paradis and Schliep, 2019). A tanglegram was plotted to compare the reference phylogeny with the closest topology obtained with single genes.

Scalability assessment using *E. coli* genomes

To evaluate scalability, annotated GFF3 files from 1,000 *E. coli* genomes were obtained from (Decano and Downing, 2019). We used subsets of 10, 100, 500 and 1,000 genomes to build pangenomes using Roary (Page et al., 2015) with default parameters. Then, we assessed the time Pagoo takes to complete five different operations using these pangenome datasets (Figure S2B). The evaluated operations were grouped in two categories: (1) loading pangenome data, and (2) querying already loaded pangenome data. In the first category, we evaluated the time it takes to: (1.1) create a Pagoo object using the built-in function 'roary_2_pagoo()' providing only the gene presence/absence matrix file, (1.2) create a Pagoo object using the 'roary_2_pagoo()' function but also providing the GFF3 files to include sequences into the Pagoo object, and (1.3) create a Pagoo object from already loaded information and sequences into the R session using the 'pagoo()' function. In the second category we measured the time it takes to (2.1) retrieve core genome sequences, and (2.2) generate a pangenome frequency plot. Each operation was repeated 10 times.

QUANTIFICATION AND STATISTICAL ANALYSIS

A Principal Component Analysis (PCA) based on accessory gene presence/absence patterns was generated with Pagoo's method 'pan_pca()'. Then, those accessory genes which most contributed to discriminate between livestock-associated and human-associated lineages were identified based on the eigenvector of the first component. Genes with a loading value lower than −0.05 and greater than 0.05 were selected (Figure S2D; Table S1).

The question "What is Life" is... a linguistic trap.
To answer according to the rules of grammar,
we must supply a noun, a thing.
But life on Earth is more like a verb.
It repairs, maintains, re-creates, and outdoes itself.

— Lynn Margulis

Capítulo 4

Desarrollo de una herramienta para reconstruir pangenomas de taxones superiores

En este capítulo describo `pewit`, el *software* que diseñé e implementé para reconstruir pangenomas de niveles taxonómicos superiores al de especie, y que llevó a tener que desarrollar `simurg` y `pagoo` para evaluarlo y darle una interfaz para el post-análisis, por lo que es el último resultado de esta tesis.

La idea se gestó hace varios años, incluso una versión muy prematura formó parte de uno de los capítulos de mi tesis de maestría, aunque en el proceso sufrió alguna tardanza por mejoras sustanciales, por la necesidad de implementar los otros dos trabajos previos para darle validez a éste, por otros proyectos que corrieron en paralelo, y por la pandemia de COVID-19 que postergó la publicación de `pewit` ya que se tuvieron que reorientar algunos esfuerzos por la emergencia sanitaria[130-135]. El problema de reconstruir pangenomas diversos surge a partir de una línea de investigación en *Leptospira sp.* donde se observa que existen especies relativamente distantes dentro del género, que contienen cepas que comparten cierto fenotipo serológico. Lo que se nos ocurrió en ese entonces consistía en reconstruir el pangenoma del género y ver qué tenían en común estas cepas, y que las diferenciaban del resto, para que desplieguen dicho fenotipo. En seguida nos encontramos con una barrera ya que los algoritmos existentes estaban diseñados para trabajar a nivel de especie, por lo que las reconstrucciones a nivel de todo el género no nos eran informativas. Una cosa que observábamos, por ejemplo, es que no obteníamos genes *core*, lo cual nos resultaba extraño dado que todo el género comparte fenotipos complejos como forma, movilidad, tamaño, una etiología similar en los caso de especies patógenas, etc, además de que es esperable encontrar vías metabólicas y genes *housekeeping* conservados.

El nombre del paquete es un acrónimo de ***P**angenome **E**stimation **W**alks **I**nside the **T**axonomy*, pero *pewit* se le dice a la versión del tero del hemisferio norte (*Vallenus vallenus*, también llamado coloquialmente «*northern lapwing*»), por lo que nos pareció divertido homenajear a tan noble animal.

4.1. Objetivo específico

Desarrollar un *software* capaz de reconstruir pangenomas bacterianos a niveles taxonómicos superiores al de especie y evaluarlo junto con otros programas disponibles contra pangenomas simulados.

4.2. Métodos e implementación

Para evitar el problema de identificación de homología remota a través de métodos basados en similitud de secuencia, decidimos basarnos en la observación de que dos proteínas ortólogas que divergieron en el tiempo tienden a perder identidad de secuencia pero mantienen la arquitectura de dominios proteicos que las componen[99,136-138], entendiendo por «arquitectura de dominios» a la secuencia contigua de dominios que componen a una proteína. Para identificar dominios utilizamos la base de datos Pfam[113], y HMMER[110,111] como motor de búsqueda ya que los modelos ocultos de Markov son más sensibles que los métodos basados en identidad de secuencia.

El paso anterior, sin embargo, sólo identifica relaciones de homología, por lo que es necesario aplicar un método subsiguiente para separar ortólogos de parálogos. Dado que la idea era reconstruir pangenomas de organismos evolutivamente remotos, para este paso decidimos no basarnos en los contextos genómicos dado que esta propiedad tiende a perderse con el tiempo. Utilizamos las definiciones de «ortólogos» y «parálogos» para separar ambas relaciones de homología, como ha sido sugerido anteriormente[88,99,106]. Para ello se implementó un algoritmo de *tree-prunning* que logra separar ortólogos de parálogos externos.

Otro punto que merece la pena mencionar es en lo referente a la implementación de heurísticas para manejar eficientemente la redundancia de secuencias. Los algoritmos mencionados son costosos computacionalmente, por lo que en buena parte del proceso sólo se aplican a un *subset* representativo de secuencias, disminuyendo en órdenes de magnitud la complejidad y el tiempo de ejecución. Para identificar genes redundantes se optó por la aplicación de MinHash[139] y *Locality-Sensitive Hashing*[140] (LSH), algoritmos basados en k-meros tomados de teorías de la información que originalmente se usaron para identificar duplicaciones o «casi duplicaciones» de páginas web en motores de búsqueda.

El programa devuelve un objeto que consiste en una clase modificada de las implementadas en `pagoo` (capítulo 3), de modo que todos los métodos y capacidades de dicho paquete se encuentran disponibles para el post-análisis, más algún otro que aprovecha los datos de las coordenadas genómicas presentes en los archivos de entrada para permitir graficar mapas de contexto genómico centrados en un *cluster* a elección.

Se evaluó la calidad de las agrupaciones clasificadas por `pewit` contra pangenomas simulados con `simurg` (capítulo 2) a distintos tiempos evolutivos, y el tiempo de ejecución contra un número creciente de genomas considerados, a la par con otros *softwares* populares en el área.

A continuación se describen los métodos de forma detallada.

4.2.1. Implementación

`Pewit` está escrito en R[109] y depende de HMMER[110,111] y MCL[98] como software externo.

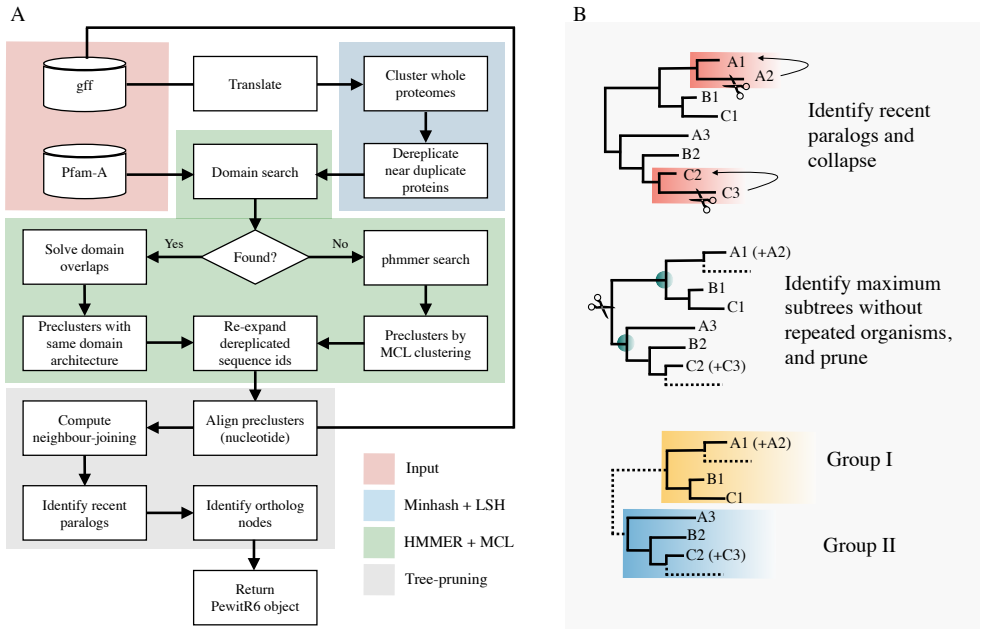


Figura 4.1: Flujo de trabajo de Pewit. A) Esquema general del flujo de trabajo. Los colores identifican los pasos clave. En rojo, la entrada que consiste en los archivos gff anotados y, opcionalmente, una base de datos local de modelos ocultos de Markov Pfam-A. En azul, los pasos donde los algoritmos de bosquejo rápido (fast sketching) reducen la complejidad del conjunto de datos al identificar secuencias casi duplicadas. En verde, se utilizan HMMER y MCL para agrupar las proteínas en clústeres gruesos. En gris, el algoritmo de poda de árboles divide los clústeres gruesos anteriores en grupos de ortólogos. B) Ilustración esquemática del algoritmo de poda de árboles. Las letras mayúsculas en las hojas representan el organismo; el número es un nombre de gen arbitrario. Las ramas discontinuas representan hojas y nodos podados. Comienza identificando eventos de duplicación recientes (in-parálogos o parálogos recientes), fusionándolos en una sola hoja, luego elimina los nodos con organismos repetidos y finalmente devuelve los grupos de ortólogos verdaderos.

Entrada

El algoritmo comienza leyendo los archivos gff y extrayendo las filas etiquetadas como «`CDS`», no considerándose otro tipo de entradas. Las secuencias de nucleótidos se extraen y se traducen. Toda la manipulación de secuencias se realiza utilizando el paquete Biostrings[141].

Opcionalmente, los usuarios pueden proporcionar una base de datos Pfam para construir arquitecturas de dominios de cada secuencia de aminoácidos.

Estimación rápida de la distancia de organismos usando MinHash sobre proteomas completos

Para cada organismo, se extraen todos los k-meros de aminoácidos, con k = 4, y se calculan 512 funciones hash que devuelven números enteros (hashes) sobre cada k-mero. Se retiene el entero más pequeño (minhashes) en cada ronda, lo que da como resultado 512 enteros por organismo. Se calcula la distancia de Jaccard entre el vector de minhashes de cada par de organismos. Con esta distancia, se calcula un agrupamiento jerárquico utilizando la función `hclust()` de R con el método `ward.D2`. Utilizando la función `cuttree()` con el parámetro de altura `h = 0.3`, se identifican clústeres de organismos que comparten aproximadamente el 70 % del contenido de k-meros. Este paso es lo suficientemente económico desde el punto de vista computacional como para identificar dónde aplicar eficazmente las heurísticas posteriores y evitar comparaciones inútiles. El paso de extracción de k-meros está implementado de forma nativa en C++, y los minhashes se calculan utilizando el paquete `textreuse`[142].

Agrupamiento rápido de proteínas usando MinHash y Locality-Sensitive Hashing

Dentro de cada grupo de organismos identificado en el paso anterior (ver sección 4.2.1), se calculan 16 minhashes ($k = 4$) para cada secuencia de aminoácidos. Se espera que los vectores de minhashes que difieren como máximo en un minhash correspondan a secuencias que comparten aproximadamente $15/16 = 0{,}9375$ o más de similitud de secuencia, lo cual es lo suficientemente alto a nivel de aminoácidos como para considerarlas como la "misma" proteína. En lugar de calcular las distancias de Jaccard entre cada par posible de vectores de 16 minhashes para detectar aquellos que difieren como máximo en un minhash, cada vector se divide en mitades y cada vector de 8 minhashes se convierte en un único valor hash (tokenized). Luego, solo los pares de mitades convertidas que comparten al menos uno de los dos tokens se marcan como candidatos para ser comparados con los 16 hashes completos (los duplicados exactos en el espacio de minhash compartirán ambos pares de tokens, y los casi duplicados que comparten 15 de 16 minhashes

diferirán en, exactamente, una mitad), reduciendo drásticamente el número de comparaciones necesarias para identificar esos casi duplicados.

Una vez identificados los pares candidatos y calculadas sus distancias de Jaccard, los grupos de casi duplicados se agrupan y solo se selecciona una secuencia correspondiente de cada grupo para pasar a las comparaciones con `HMMER` en el siguiente paso.

Búsqueda de dominios Pfam mediante `hmmsearch`, limpieza de solapamientos de dominios y agrupamiento por arquitectura de dominios

Una secuencia representativa de cada grupo de casi duplicados del paso anterior se compara contra una base de datos Pfam local utilizando el comando `hmmsearch`, del conjunto de herramientas `HMMER`. To identificar coincidencias se utilizan los Gathering Bit Scores curados de cada modelo configurando la opción `--cut_ga` (ver el manual de `HMMER` como referencia). Dado que algunos dominios Pfam se agrupan en clanes, una única región de una secuencia de aminoácidos puede tener múltiples aciertos de diferentes dominios del mismo clan.

Para eliminar estos aciertos superpuestos, se calcula una matriz de solapamiento determinando qué coordenadas de dominio comparten posiciones en la secuencia de aminoácidos. Considere el siguiente ejemplo: una proteína que tiene tres dominios Pfam A, B y C, en la cual los dos primeros tienen coordenadas superpuestas. La matriz de dominios superpuestos sería:

Cuadro 4.1: Ejemplo de una matriz de dominios superpuestos para el caso (A[ ) B] (C)

	A	B	C
A	TRUE	TRUE	FALSE
B	TRUE	TRUE	FALSE
C	FALSE	FALSE	TRUE

Sobre esta estructura de datos, un algoritmo simple de programación dinámica recupera una lista de solapamientos de dominios:

Algorithm 1 Algoritmo de programación dinámica para determinar solapamientos de dominios.

Require: m es una matriz de dominios superpuestos de tamaño $N \times N$

```
1: i ← 1                                                          // Índice de fila
2: j ← 1                                                          // Índice de columna
3: List L ← {}  // Crear Lista para guardar resultados intermedios de dominios
   superpuestos locales.
4: List Overlaps ← {}        // Crear Lista de dominios superpuestos a devolver.
5: domainName ← m_{i; j}.COLUMNNAME()    // Obtener el nombre de columna de
   la primera celda.
6: L.APPEND(domainName)          // Inicializar la lista intermedia L con el
   primer nombre de columna.
7: while j ≠ N do                    // Mientras j no alcance el borde derecho:
8:     if m_{i; j+1} = TRUE then    // Si la siguiente celda (derecha) es TRUE...
9:         j ← j + 1                // ...entonces moverse a la siguiente celda.
10:        domainName ← m_{i; j}.COLUMNNAME()  // Obtener nombre de columna.
11:        L.APPEND(domainName)    // Añadir nombre del dominio (columna) a
   la lista intermedia.
12:    else if m_{i+1; j} = TRUE then  // Si la celda de la derecha es FALSE pero
   la de abajo es TRUE...
13:        i ← i + 1                // ...entonces moverse a la celda de abajo.
14:    else     // Si tanto la celda de la derecha como la de abajo son FALSE...
15:        Overlaps.APPEND(L)     // ...anidar la lista resultante en la lista
   final.
16:        List L ← {}                          // Registrar nueva lista local...
17:        i ← i + 1 // ...luego moverse a la celda inferior derecha (diagonal).
18:        j ← j + 1
19:        domainName ← m_{i, j}.COLUMNNAME()        // Obtener el nombre de
   columna de esta celda.
20:        L.APPEND(domainName)    // Inicializar la nueva lista intermedia
   con este dominio.
21:    end if
22: end while
23: Return Overlaps                                  // Devolver solapamientos.
```

El algoritmo anterior devolverá la siguiente lista anidada en el ejemplo de la Tabla 4.1:

(1) A, B

(2) C

Luego se retiene el dominio con el e-value mínimo en cada lista de solapamiento local, lo que conduce a una arquitectura de dominios (es decir, la secuencia lineal consecutiva de dominios no superpuestos) para cada secuencia de aminoácidos con aciertos en Pfam.

Una vez aplicado este procedimiento sobre cada secuencia de aminoácidos, la arquitectura de dominios se transfiere primero a sus secuencias casi idénticas (si las hay) identificadas en el paso Minhash+LSH (ver sección 4.2.1), y luego todas las secuencias con la misma arquitectura de dominios se agrupan juntas. Estos preclústeres se dividirán posteriormente en familias de ortólogos verdaderos en los pasos siguientes mediante un algoritmo de poda de árboles.

Agrupamiento de proteínas sin dominios Pfam

Las secuencias de aminoácidos sin ningún dominio Pfam encontrado en el paso anterior (ver sección 4.2.1) se comparan entre sí mediante `phmmer` y se agrupan utilizando `MCL`. Las secuencias casi idénticas del paso MinHash+LSH (ver 4.2.1) se asignan al preclúster de su secuencia representativa y se pasan al algoritmo de poda de árboles que sigue. Los usuarios pueden confiar únicamente en este paso de preagrupamiento si no proporcionan la base de datos Pfam como argumento.

Identificación de grupos de ortólogos mediante un algoritmo de poda de árboles

Las secuencias de nucleótidos de los preclústeres identificados en los dos pasos anteriores (hasta este paso, todas las operaciones se realizaron con las secuencias de aminoácidos) se alinean utilizando el paquete de R `DECIPHER`[143]. Se calcula un árbol por *neighbor-joining*[144] utilizando el paquete `ape`[145], se enraíza en el punto medio (midpoint rooted) y se implementa el siguiente algoritmo para identificar los nodos que contienen parálogos recientes (in-parálogos):

Algorithm 2 Algoritmo de recorrido de árboles para identificar parálogos recientes.

Require: *tree* es un árbol inferido mediante *neighbor-joining*.

List $nodesWithParalogs \leftarrow \{\}$ // Inicializar lista vacía.
for tip **in** tree.TIPS( ) **do** // Iterar sobre cada hoja
    Var $myNodeWithParalogs \leftarrow \{\}$ // Inicializar variable de nodo.
    $ancestorNode \leftarrow tip$.ANCESTORNODE() // Calcular el nodo ancestro de la hoja
    $descendantTips \leftarrow ancestorNode$.DESCENDANTTIPS() // Calcular todas las hojas que descienden de ese nodo.
    while all $descendantTips$.ORGANISMS() $=$ tip.ORGANISM() **do** // Mientras todas las hojas descendientes pertenezcan al mismo organismo, hacer:
        $myNodeWithParalogs \leftarrow ancestorNode$ // Guardar nodo actual.
        $ancestorNode \leftarrow ancestorNode$.ANCESTORNODE() // Saltar al nodo anterior.
        $descendantTips \leftarrow ancestorNode$.DESCENDANTTIPS() // Recuperar todas las hojas de ese nodo anterior.
    end while
    $nodesWithParalogs$.APPEND($myNodeWithParalogs$) // Añadir el nodo a la lista.
end for
$nodesWithParalogs \leftarrow nodesWithParalogs$.UNIQUE() // Eliminar nodos repetidos.
Return: $nodesWithParalogs$ // Devolver lista con nodos que contienen in-parálogos

Una vez identificados los nodos con parálogos recientes, se eliminan del árbol todas las hojas descendientes de ese nodo excepto una. Este árbol defoliado se pasa luego al algoritmo de poda para identificar los grupos finales de ortólogos. Este algoritmo consiste en identificar el subárbol máximo que no contenga hojas pertenecientes al mismo organismo:

Algorithm 3 Algoritmo de poda de árboles para identificar nodos definitorios de ortólogos.

Require: *tree* es un árbol inferido mediante *neighbor-joining*.

List $nodeWithoutDuplicates \leftarrow \{\}$ // Inicializar lista para registrar nodos sin organismos duplicados.
$nodes \leftarrow tree$.NODES() // Obtener lista de nodos
for node **in** nodes **do** // Iterar sobre cada nodo
  $descendantTips \leftarrow node$.DESCENDANTTIPS() // Obtener las hojas descendientes del nodo.
  $organisms \leftarrow descendantTips$.ORGANISMS() // Obtener el organismo al que pertenece cada hoja.
  if $organisms$.NODUPLICATED() **then** // Si no hay ningún organismo duplicado...
    $nodeWithoutDuplicates$.APPEND($node$) // ...guardar el nodo.
  end if
end for
List $orthologGroupNodes \leftarrow \{\}$ // Inicializar lista para guardar nodos que contienen un único grupo de ortólogos.
for node_i **in** nodeWithoutDuplicates **do** // Iterar sobre cada nodo.
  List $isContainedIn \leftarrow \{\}$ // Inicializar lista temporal para guardar los nodos en los que se encuentra contenido cada uno.
  for node_j **in** nodeWithoutDuplicates **do**
    $tips_\text{i} \leftarrow node_\text{i}$.DESCENDANTTIPS()
    $tips_\text{j} \leftarrow node_\text{j}$.DESCENDANTTIPS()
    if all tips_j **in** tips_i **then** // Si todas las hojas del nodo j están contenidas en el nodo i...
      $isContainedIn$.APPEND($node_\text{j}$) // ...guardar este nodo.
    end if
  end for
  if $isContainedIn$.LENGTH() $= 1$ **then** // Si esta lista se contiene solo a sí misma...
    $orthologGroupNodes$.APPEND($node_\text{i}$) // ...este es un nodo contenedor de grupo de ortólogos.
  end if
end for
Return: $orthologGroupNodes$

Una vez identificados los nodos que contienen cada grupo de ortólogos, se recuperan las hojas descendientes de cada uno de ellos y se agrupan juntas. Las hojas marcadas como parálogos recientes y eliminadas del árbol en el algoritmo

2 se asignan luego al grupo en el que se dejó su hoja representativa en este paso final.

La mayor parte de las manipulaciones de árboles dependen del paquete `ape`[145] y del paquete `phangorn`[146]. Ambos algoritmos anteriores están muy simplificados para ilustrar la idea general; los casos marginales son manejados eficazmente por el paquete, pero no se muestran aquí.

Devolución de un objeto de clase `PewitR6`

El software devuelve un objeto de clase `PewitR6`, que es una versión ligeramente extendida de la clase `PgR6MS` del paquete `pagoo` (ver 3). El objeto contiene los datos estructurados de forma relacional, donde cada gen está vinculado a un organismo y asignado a un clúster. Cada una de estas tres categorías (genes, clústeres y organismos) se puede complementar con metadatos adicionales. También contiene métodos para consultar y analizar los datos de una manera intuitiva.

Además de la estructura de datos y los métodos proporcionados por el paquete `pagoo`, en la clase `PewitR6` se proporciona un método para consultar y graficar el contexto genético alrededor de un conjunto de genes, dado que las coordenadas de los genes se capturan a partir de los archivos gff de entrada.

4.2.2. Evaluación comparativa de herramientas de reconstrucción de pangenomas

Versiones de software

`Pewit` se comparó con las siguientes herramientas: `roary`, `PanX`, `micropan` y `panaroo`. En el caso de `pewit` y `micropan`, se utilizó la instalación nativa de `R`, con las siguientes dependencias externas: el conjunto de herramientas `blast+`, `HMMER` y `MCL`. Esas y otras herramientas de soporte junto con sus versiones se enumeran a continuación:

Cuadro 4.2: Versiones de software de las herramientas instaladas de forma nativa y paquetes de R.

Software	Versión	Cita
R	4.0.5	[109]
pewit	1.3.2	Este trabajo.
pagoo	0.3.13	[128]
simurg	0.2.1	[126]
ggplot2	3.3.5	[147]
micropan	2.1	[93]
conjunto blast+	2.12.0	[89]
HMMER	3.3.2	[110,111]
MCL	14-137	[98]

Para ejecutar `pewit`, se utilizó la versión 35.0 de Pfam[113]. `Roary`, `panx` y `panaroo` se ejecutaron a través de imágenes de Singularity[148] (versión 3.8.6-1) recuperadas de la colección BioContainers[149]. Las aplicaciones en contenedores utilizadas se enumeran a continuación:

Cuadro 4.3: Fuentes y etiquetas de los contenedores Singularity.

Transporte	Registro	Colección	Nombre	Etiqueta	Cita
docker	quay.io	biocontainers	roary	3.13.0--pl526h516909a_0	[56]
docker	quay.io	biocontainers	panx	1.6.0--py27_0	[95]
docker	quay.io	biocontainers	panaroo	1.2.10--pyhdfd78af_0	[104]
docker	quay.io	biocontainers	gff3toembl	1.1.4--pyh864c0ab_2	[150]

Evaluación del tiempo de ejecución

Para evaluar los tiempos de ejecución, cada algoritmo se ejecutó contra 10, 20, 30, 50, 70, 90 y 120 genomas de *C. fetus* muestreados de un conjunto de datos publicado[151]. Para cada número de genomas, se realizaron 5 muestreos independientes (réplicas).

`Pewit` se evaluó tanto con como sin Pfam, y en todos los casos utilizando solo 1 procesador (CPU).

En el caso de `roary`, se evaluaron los parámetros `-i 95` y `-i 70` (del manual: `-i porcentaje de identidad mínimo para blastp`), y en ambos casos se utilizó `-p 1` (número de procesadores).

Los tiempos de ejecución de `micropan` se evaluaron en dos etapas. Primero se registró el tiempo de todas las comparaciones de `blastp` (etapa de «*todos contra*

todos≫), y luego el tiempo para calcular cada uno de los tres algoritmos de agrupamiento (simple, promedio y completo) se sumó a la etapa de ≪*todos contra todos*≫. Los parámetros principales utilizados fueron `e.value = 1e-10` y `threads = 1L` en la función `blastpAllAll()`, y `threshold = 0.74` en la función `bClust()`. Para este algoritmo, los tiempos de ejecución se evaluaron hasta 50 genomas.

`PanX` se evaluó con los siguientes parámetros: `-sl Cfetus -cg 0.5 -st 1 2 3 4 5 6 -t 1 -ct`.

`Panaroo` se evaluó en sus tres modos para refinar genes potencialmente contaminantes: estricto, moderado y sensible, y con `--threads` configurado en 1.

Evaluación del agrupamiento

Para evaluar la calidad del agrupamiento de cada algoritmo, se utilizó el paquete `simurg` (ver 2) para simular pangenomas con un número creciente de generaciones a partir del ancestro común más reciente.

Los parámetros de `simurg` utilizados para simular el conjunto de datos fueron los siguientes:

Cuadro 4.4: Parámetros de `simurg` utilizados para simular los pangenomas.

Parámetro	Descripción	Valor
ref	Un archivo multi-fasta de donde muestrear los genes.	"Ecoli_pan_genome_reference.fa"
norg	El número de organismos muestreados al tiempo final.	10
Ne	Número de generaciones desde el MRCA.	2e+09, 1e+10, 5e+10, 25e+10, 125e+10
C	El tamaño del genoma central (coregenome).	100
υ	Probabilidad de ganancia de genes por generación.	1e-8
ν	Probabilidad de pérdida de genes por generación.	1e-11
μ	La tasa de sustitución por sitio por generación.	5e-12

Se probaron cinco números de generaciones (ver Tabla 4.4, parámetro N_e), con

cinco réplicas independientes cada uno. Todos los softwares se probaron con los mismos parámetros descritos en la sección 4.2.2.

El resultado de un par de genes que se agruparon juntos se definió como un verdadero positivo (TP) si aparecían efectivamente juntos en la simulación, o como un falso positivo (FP) si no lo hacían. El resultado de un par de genes que no se agruparon juntos se definió como un verdadero negativo (TN) si tampoco pertenecían al mismo clúster en la simulación, o como un falso negativo (FN) si lo hacían. Se contabilizó la clasificación para cada par de genes, se calculó una matriz de confusión y se derivaron las estadísticas de agrupamiento a partir de ella.

4.2.3. Casos de estudio

Para el análisis de Serratia, se descargaron 664 genomas anotados desde el enlace proporcionado en el estudio (`https://doi.org/10.6084/m9.figshare.18051824`). Después de la reconstrucción del pangenoma con `pewit`, los genes core se alinearon, se concatenaron y se utilizó IQTree[152] para la inferencia filogenética con los mismos parámetros de modelo que en el artículo original[153]. Se utilizó RHierBAPS para inferir la estructura, y se empleó el script `classify_genes.R`[57] del repositorio de Horesh para calcular las categorías *twilight*. Se utilizaron los paquetes `ggplot2`[147] y `ggtree`[154] para graficar el árbol y el mapa binario.

En el caso de la familia *Enterobacteriaceae*, se descargaron todos los ensamblajes de genomas de referencia (211) disponibles en el NCBI y se anotaron con Prokka[155]. Los pangenomas se reconstruyeron con `pewit` y panaroo[104] (con la opción `--clean-mode moderate`). Las frecuencias génicas se calcularon a partir de la pan-matriz de cada uno de los softwares de reconstrucción de pangenomas y se graficaron en escala log 10 para una mejor visualización.

4.2.4. Disponibilidad

`pewit` está disponible para su descarga en `https://github.com/iferres/pewit`. Todos los scripts para reproducir los análisis están disponibles en `https://github.com/iferres/pewit_publication_scripts`.

4.2.5. Hardware

Todas las evaluaciones se ejecutaron en una computadora de escritorio con CPU Intel® Core™ i7-7700 @ 3.60GHz x 8, 64Gb de RAM, con el sistema operativo Fedora 34 (Workstation Edition).

4.3. Resultados

4.3.1. Evaluación comparativa (Benchmark)

Evaluamos `pewit` frente a los siguientes softwares de reconstrucción de pangenomas de última generación: roary[56], que es probablemente el más citado de las herramientas de reconstrucción de pangenomas, con el parámetro `-i` configurado en 95 (por defecto) y 70 para capturar relaciones más distantes; el paquete de R `micropan`, que implementa una estrategia `blastp` de *todos contra todos* seguida de uno de los tres métodos de aglomeración implementados en la función `hclust()` de R (simple, promedio y completo); `panX`[95], que también compara todas las proteínas contra todas pero utiliza `DIAMOND`[156] en lugar de `blastp` como motor de búsqueda; y `panaroo`[104], que utiliza CD-HIT[100] y luego se apoya en el contexto génico para refinar el agrupamiento, y cuenta con tres modos diferentes para lidiar con anotaciones potencialmente erróneas (estricto, moderado y sensible).

Cabe destacar que `pewit`, `roary` y `panaroo` implementan un algoritmo de preagrupamiento muy rápido (Minhash + LSH en el caso de `pewit`, y `CD-HIT` en el caso tanto de `roary` como de `panaroo`) para eliminar la redundancia y disminuir la complejidad, aplicando luego el algoritmo central, que tiende a ser computacionalmente más intensivo, sobre el conjunto de datos reducido.

El preagrupamiento Minhash + LSH es un método eficiente para eliminar la redundancia y disminuir la complejidad

Para comparar los tiempos de ejecución, cada algoritmo se ejecutó contra un número creciente de genomas de *Campylobacter fetus* extraídos de un conjunto de datos publicado[151], desde 10 hasta 120 (ver métodos, sección 4.2.2). In cada caso, el número de hilos de CPU se fijó en 1 para evaluar el algoritmo base. Los resultados se muestran en la figura 4.2.

Los tiempos de ejecución se pudieron agrupar en dos categorías: aquellos de herramientas que implementan una estrategia de preagrupamiento muy rápida para eliminar la redundancia antes de la búsqueda más sensible, y aquellos de herramientas que no lo hacen y utilizan una estrategia de «*todos contra todos*», siendo esta última mucho más lenta que la primera (figura 4.2A).

En el caso de `pewit`, aunque es marginalmente más lento, sus tiempos de ejecución están en el mismo orden de magnitud que los de `roary` y `panaroo`, e incluso tienden a converger a medida que aumenta el tamaño del conjunto de datos, como se puede observar al graficar con una escala de tiempo Log_{10} (figura 4.2B). Esto es especialmente importante cuando se trabaja con conjuntos de datos de tamaño actual, que pueden comprender cientos o potencialmente miles de genomas.

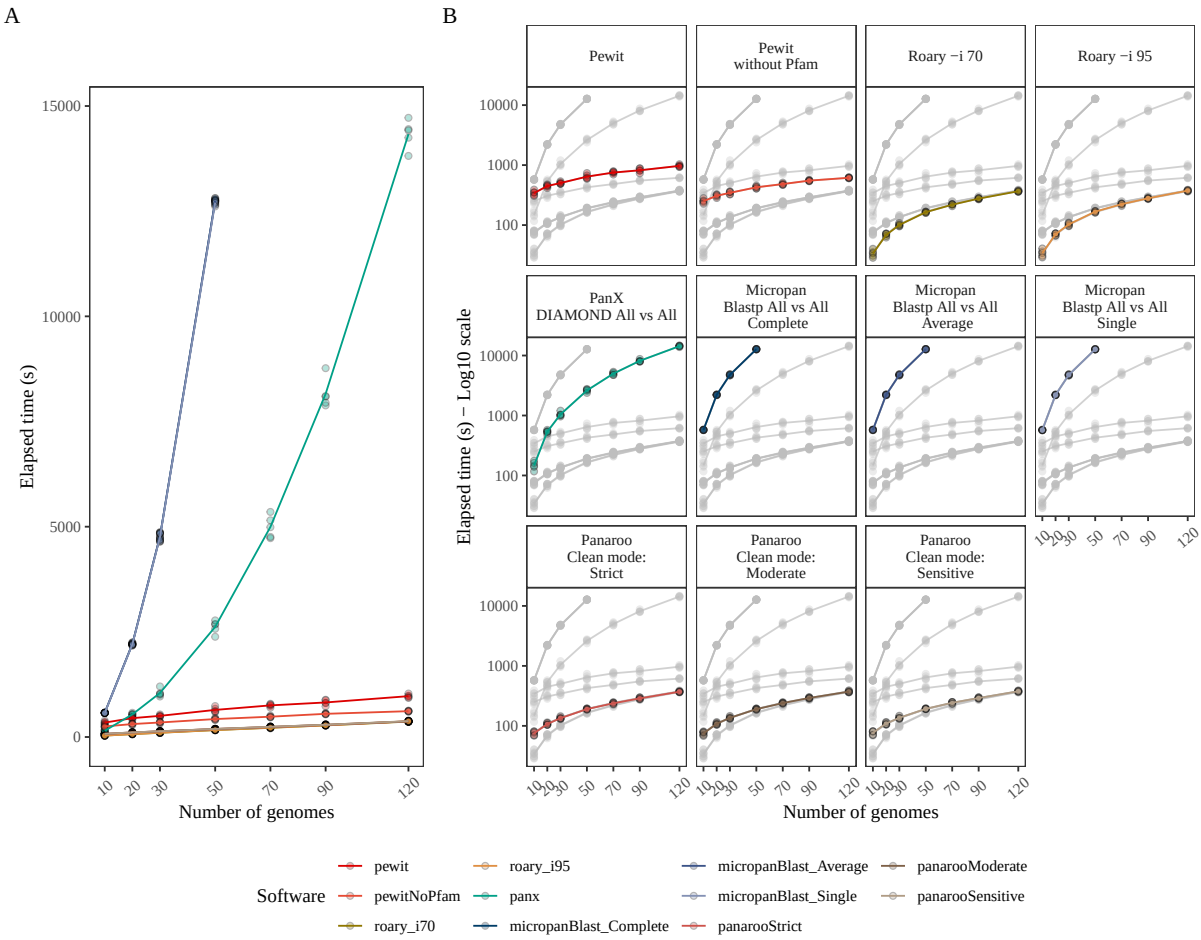


Figura 4.2: Evaluación de los tiempos de ejecución de las herramientas de pangenoma. A) Tiempos de ejecución en escala lineal (eje Y). B) Tiempos de ejecución en escala log10 (eje Y); cada panel resalta una condición, el software evaluado más algunos parámetros.

La búsqueda con HMMER combinada con algoritmos de poda de árboles captura ortólogos remotos mejor que las herramientas de búsqueda por alineamiento y los refinamientos por contexto genético

Para evaluar la calidad del agrupamiento, simulamos pangenomas con el paquete de R `simurg`[126] utilizando tasas de mutación y tasas de ganancia/pérdida de genes constantes, pero aumentando el número de generaciones. Luego intentamos reconstruir los pangenomas con cada una de las herramientas mencionadas y comparamos las agrupaciones con los datos de la simulación. El resultado de cada par de genes, estando o no agrupados juntos, se clasificó como verdadero positivo, verdadero negativo, falso positivo y falso negativo según su concordancia con los datos de la simulación (ver métodos, sección 4.2.2), se contabilizó y se derivaron las estadísticas de agrupamiento a partir de ello. Se calculó el Coeficiente de Correlación de Matthews[157] (MCC) para cada clasificación binaria y los resultados se muestran en la figura 4.3A.

`Pewit` se comportó de manera comparable a `roary` al reconstruir pangenomas con tiempos de evolución cortos, y de manera comparable o mejor que las estrategias de *todos contra todos* como `micropan` y `panX` cuando los pangenomas eran altamente divergentes. En todos los casos, `pewit` superó o fue similar al mejor de los otros softwares en cada tiempo de evolución. En el caso de `panaroo`, dado que el algoritmo se basa en gran medida en la contigüidad genética y las simulaciones no toman eso en cuenta, el rendimiento fue deficiente. Probablemente funcionaría mucho mejor en un escenario de caso real con cepas muy cercanas, pero a medida que aumentan los tiempos de evolución se espera que la mayor parte del contexto genético se pierda de todos modos, y su motor de búsqueda `CD-HIT` no fue diseñado para detectar homólogos distantemente relacionados por sí solo.

La precisión y el *recall* disminuyeron a medida que aumentaron los tiempos de evolución para todos los softwares probados (Figura 4.3B). En el caso de `pewit`, ambas estadísticas disminuyeron de manera uniforme, mientras que en la mayoría de los demás softwares disminuyeron de forma desigual, perdiendo una de las magnitudes más rápidamente que la otra. El perfil más comparable al de `pewit` en este aspecto fue el de «*todos contra todos*» con enlace completo de `micropan`, siendo este el método más costoso con la rutina de aglomeración más robusta.

4.3.2. Casos de estudio

Para evaluar pewit con conjuntos de datos reales, volvimos a analizar los genomas de *Serratia* del estudio de Williams et al. de 2022 sobre el género[153]. En este estudio, los autores utilizaron panaroo para reconstruir el pangenoma a nivel de género y clasificaron cada clúster de genes de acuerdo con la llamada clasificación *twilight*, tal como fue conceptualizada por Horesh y colaboradores[57]. Reproduji-

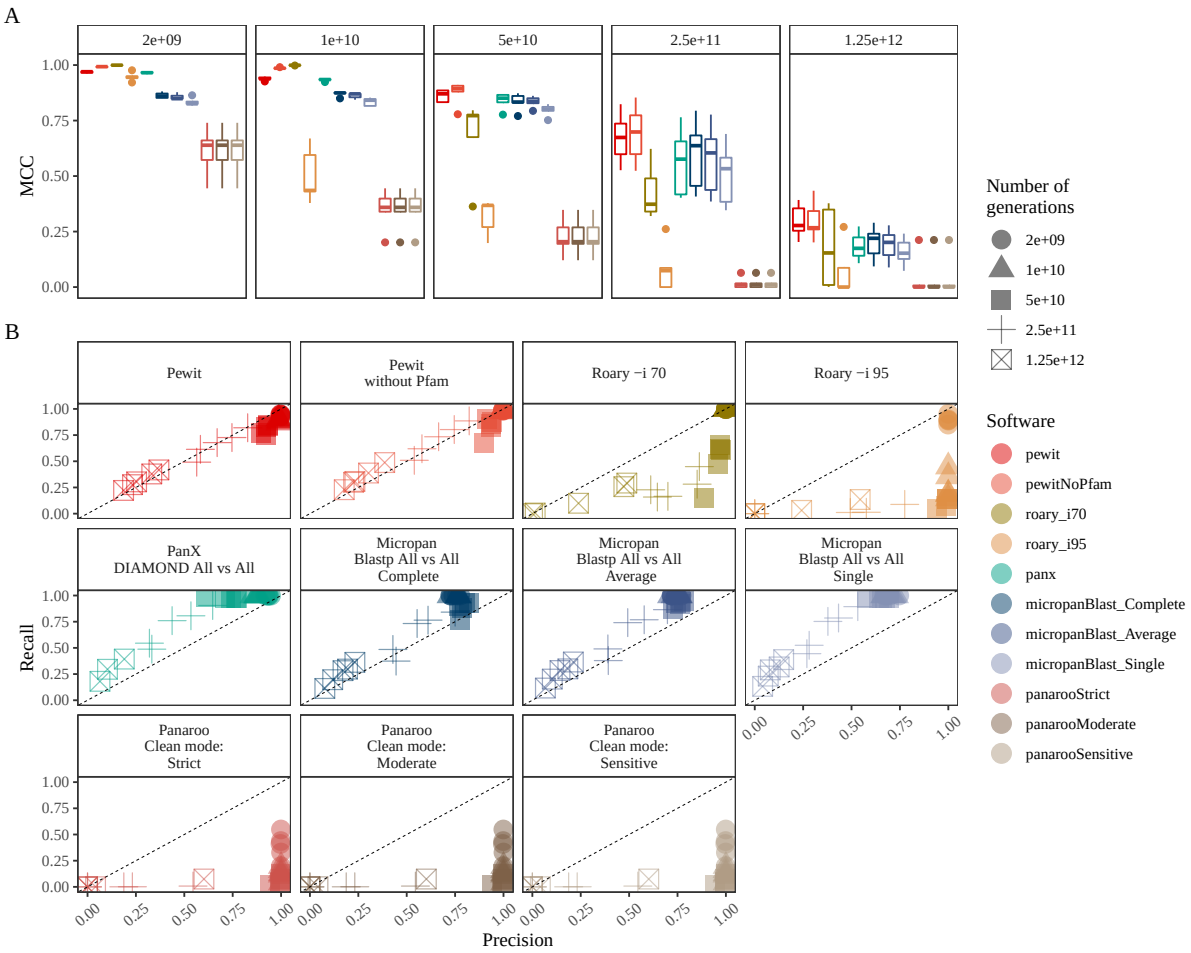


Figura 4.3: Evaluación de los algoritmos de agrupamiento de pangenomas frente a pangenomas simulados con un número creciente de generaciones. A) Coeficiente de Correlación de Matthews calculado para cada software. B) Exhaustividad (Recall) vs. Precisión; la línea discontinua muestra la diagonal.

mos el análisis y la figura 2A del estudio citado para demostrar que `pewit` puede llegar a las mismas conclusiones pero utilizando un enfoque completamente diferente para reconstruir el pangenoma (figura 4.4A). El alcanzar resultados similares demostró que panaroo se comporta mucho mejor con conjuntos de datos reales en comparación con lo que se mostró en las simulaciones (figura 4.3), lo cual puede explicarse considerando que panaroo se apoya fuertemente en el contexto genómico para identificar relaciones de ortología, aspecto que no fue considerado en las simulaciones.

Para presionar aún más ambos métodos, descargamos todos los ensamblajes de referencia de la familia *Enterobacteriaceae* del NCBI y procedimos a reconstruir el pangenoma a nivel de familia. Planteamos la hipótesis de que, dado que panaroo fue desarrollado para lidiar con pangenomas a nivel de especie (aunque funcionó muy bien con el género *Serratia*), podría estar sobre-representando el número de clústeres al dividir grupos de ortólogos que contienen secuencias divergentes. Por el contrario, pewit debería ser capaz de identificar esas relaciones de homología remota. El gráfico de frecuencias (figura 4.4B) muestra que panaroo probablemente está sobre-dividiendo grupos de ortólogos remotos, encontrando difícilmente algún gen central para todo el conjunto de datos, mientras que pewit es capaz de encontrar clústeres de genes presentes en todos los organismos.

4.4. Discusión de resultados de pewit

A medida que las bases de datos de genomas tanto de aislados como ensamblados a partir de metagenomas siguen creciendo, los enfoques pangenómicos para estudiar la evolución microbial no solo se vuelven viables, sino también necesarios para comprender las similitudes y diferencias fenotípicas. La creciente importancia de este concepto relativamente novedoso se evidencia fácilmente al buscar registros de «pangenoma» en el portal PubMed, el cual hasta la fecha parece mostrar un crecimiento exponencial. Desde las implementaciones más antiguas y directas, como los enfoques de «*todos contra todos*», hasta las heurísticas de búsqueda modernas y las estrategias de grafos, las herramientas de software se han centrado en la reconstrucción de pangenomas a nivel de especie o cepa, pero el concepto puede, en principio, extenderse a niveles taxonómicos superiores. El desafío práctico a este respecto sigue siendo la identificación de relaciones de homología remota de una manera computacionalmente eficiente.

Utilizando una combinación de técnicas de bosquejo ultra rápidas, detección de homología de secuencias mediante Modelos Ocultos de Markov y algoritmos filogenéticos de poda de árboles, demostramos que es posible reconstruir de manera efectiva y eficiente pangenomas que contienen genomas divergentes. Cada uno de los tres algoritmos mencionados aborda un desafío diferente en la reconstrucción

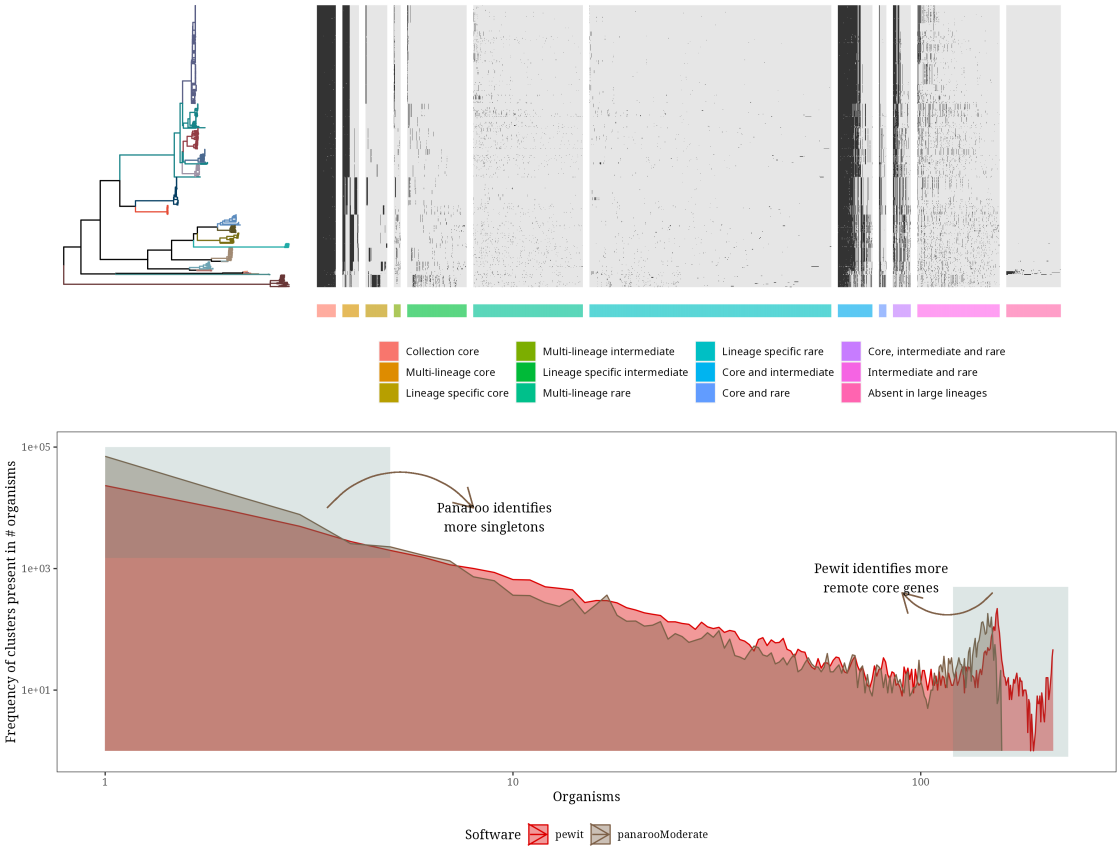


Figura 4.4: Casos de estudio. El panel superior muestra la replicación del análisis de Williams et al. 2022 del género *Serratia*, a partir de la reconstrucción del pangenoma realizada con pewit. La parte más a la izquierda representa un árbol filogenético inferido del alineamiento y concatenación de los genes centrales, con las determinaciones de cepas de RHierBAPS coloreadas por ramas. La parte más a la derecha es un mapa binario que representa la presencia (negro) o ausencia (gris) de un gen (columnas) en cada genoma (filas, en el orden del árbol filogenético). La barra de color muestra cada categoría de clúster, según lo definido por Horesh et al. 2021. El panel inferior muestra un gráfico de frecuencias para la reconstrucción del pangenoma tanto de panaroo (gris) como de pewit (rojo). El eje X está en escala log10.

de pangenomas: la fase de bosquejo rápido reduce drásticamente la complejidad y, por lo tanto, los tiempos de ejecución al eliminar la redundancia; el módulo de búsqueda HMM proporciona un método sensible para identificar relaciones de homología remota; y finalmente, los algoritmos de poda de árboles ayudan a distinguir los ortólogos de los parálogos aplicando conceptos filogenéticos.

Vale la pena señalar que esta combinación de técnicas superó a todos los demás softwares, o fue similar al mejor de ellos, en todos los escenarios de simulación con tiempos de evolución crecientes, sin la necesidad de ajustar ningún parámetro en absoluto. Asimismo, los tiempos de ejecución fueron competitivos frente a los algoritmos más rápidos, y los resultados mostraron una convergencia a medida que se incluían más genomas en los análisis, lo cual es clave en un escenario donde se espera que crezca el número de genomas considerados en los estudios pangenómicos.

Curiosamente, los mejores resultados tanto para la clasificación como para los tiempos de ejecución se lograron sin la base de datos Pfam, algo que al principio fue inesperado por nosotros. Cabe destacar, sin embargo, que la evaluación de la clasificación se realizó sobre conjuntos de datos simulados que pueden estar sesgados y no reflejar casos reales. Por ejemplo, las mutaciones se introducen aleatoriamente en las simulaciones y se espera que los dominios proteicos reales no cambien de esa manera, sino que los motivos y los tipos de aminoácidos (carga y polaridad) deberían ser más robustos frente a tiempos evolutivos largos.

El reanálisis de un conjunto de datos del género *Serratia* mostró que `pewit` se comportó de manera muy similar a `panaroo`, un software de reconstrucción de pangenomas bien establecido, en conjuntos de datos reales a este nivel taxonómico. Fue al nivel taxonómico de familia cuando la diferencia entre los dos softwares se hizo más evidente, siendo `pewit` capaz de recuperar genes centrales, mientras que panaroo no fue lo suficientemente sensible como para recuperar estos genes distantemente relacionados. El número de singletons (genes únicos) y genes de baja frecuencia, sin embargo, estuvo sobre-representado en la reconstrucción de panaroo, lo que indica que algunos genes que probablemente deberían pertenecer al mismo clúster se dividieron en varios debido a una menor capacidad para detectar homología remota.

La posibilidad de extender el concepto de pangenoma más allá del nivel taxonómico de especie ya ha sido discutida[123,158], pero las herramientas específicas para este propósito no han sido consideradas adecuadamente. Los resultados de este estudio muestran que el problema es principalmente metodológico y tiene que ver con la definición de homología, ortología y paralogía que utilizamos. El concepto se extiende naturalmente una vez que se toma en consideración un método adecuado con un buen conjunto de datos.

Lo que dejo por escrito
No está tallado en granito
Yo apenas suelto en el viento
Presentimientos
Pido lo que necesito
Tinta y tiempo, tinta y tiempo

— ***Jorge Drexler***
«Tinta y tiempo»

Capítulo 5

Discusión

Como se mencionó en la introducción, la pangenómica es un abordaje relativamente nuevo cuya aplicación viene creciendo de forma exponencial junto con el aumento de genomas disponibles en bases de datos públicas. Es esperable que esta tendencia continúe en los próximos años, por lo que innovar en métodos computacionales es fundamental para extraer la máxima información posible de los datos disponibles.

En esta tesis se desarrollaron tres paquetes escritos en el lenguaje R que van en esa dirección. Se abordó la simulación, la reconstrucción, y el post-análisis de pangenomas bacterianos, mediante la aplicación de múltiples áreas del conocimiento, tales como la estadística, la programación, la ciencia de datos, aplicaciones de las teorías de la información, genómica comparada, y evolución. Dos de estos paquetes fueron publicados en revistas arbitradas, y cuentan con tutoriales disponibles en la web, y el tercero se encuentra en etapas finales de análisis y escritura.

Cabe mencionar que tanto `pagoo` como versiones preliminares de `pewit` han sido aplicados en casos reales de estudio por parte de colegas del laboratorio, o colaboradores externos. Por ejemplo, con Thibeaux y colaboradores (2018)[159] también utilizamos `pewit`, en este caso para reconstruir el pangenoma del género *Leptospira*, y caracterizar nuevas especies. Con Costa y colaboradores (2021)[160] utilizamos `pewit` para reconstruir el pangenoma de dos subespecies de *Campylobacter hyointestinalis* con el fin de caracterizar las principales fuentes de diversidad molecular entre ambos. En el trabajo de Nzoyikorera y colaboradores (2021)[161], el Dr. Pablo Fresia (quien hace de tutor en este doctorado) aplica `pagoo` para manipular datos y aplicar estadísticos sobre el pangenoma de el serotipo 1 de *Streptococcus pneumoniae*, reconstruido con Roary[56]. En otros trabajos que se encuentran aún en maduración también se aplican estos dos métodos, por sí solos o en conjunto. Para el caso de `simurg`, su ámbito de aplicación es más restringido, pero podría usarse para testear algunas hipótesis evolutivas, o evaluar mejoras a los paquetes desarrollados.

Respecto al objetivo general, el conjunto de técnicas implementadas en *pewit* logró superar o igualar a todos los restantes *softwares* de reconstrucción de

pangenomas en cada tiempo evolutivo simulado, sin necesidad de cambiar ningún parámetro, de acuerdo a las métricas de clasificación binaria calculadas. En tiempos de ejecución también fue competitivo con los *softwares* más rápidos, que tienden a ser aquellos con peor calidad de clasificación a tiempos evolutivos grandes, por lo que `pewit` logra un excelente compromiso entre velocidad y calidad de los agrupamientos. Con una herramienta capaz de reconstruir pangenomas identificando relaciones de homología remotas, se plantea entonces el punto central de esta tesis que es la extensión del concepto a niveles taxonómicos mayores al de especie. Si bien esta discusión parte de un pequeño aporte metodológico, es una primera aproximación a este problema, por lo que se «abre la cancha» a desarrollos y mejoras posteriores en este sentido. Estos y nuevos métodos que aborden dicho problema habilitarán estudios evolutivos sobre la aparición o desaparición de fenotipos de interés a escalas temporales más amplias, permitiendo a su vez integrar información de distintos grupos taxonómicos que anteriormente, y desde la perspectiva pangenómica, debían analizarse de forma aislada.

En esta tesis se muestra como un concepto, el pangenoma, casi canónicamente aplicado cierto problema, la especie bacteriana, simplemente requiere un avance metodológico para que sea naturalmente extendido o generalizado. Muchas veces es la ciencia la que permite nuevas tecnologías o métodos, pero algunas otras veces es todo lo contrario: es el refinamiento del método lo que habilita a hacerse nuevas preguntas, lo cual luego decanta en nueva ciencia.

La revelación me dejó atónita. Acababa de comprender uno de los mecanismos más crueles de los adultos: asustar a los más débiles, aterrorizarlos. Y uno de los mecanismos más complejos de defensa: elegir aquello que se nos quiere imponer, como un acto de libertad, no como un castigo.

— ***Cristina Peri Rossi***
«*La insumisa*»

Capítulo 6

Perspectivas

Como perspectivas, en el caso de `simurg` podría incorporarse la posibilidad de agregar a la simulación la transferencia horizontal de genes, para lo cual ya existe un modelo matemático del mismo[28]. A su vez, se podría actualizar el objeto de salida para que devuelva un objeto derivado de las clases implementadas en `pagoo`, volviendo más armonioso al ecosistema de paquetes generado.

Para `pagoo`, se podrían sumar nuevos métodos de análisis de pangenomas desarrollados más recientemente y que se mencionan en la introducción (ver secciones 1.3.2, 1.3.1). Otra mejora que me gustaría desarrollar es la posibilidad de exportar y trabajar a través de conexiones a una base de datos local SQLite (ya que se trata de una estructura relacional) para evitar cargar todos los pangenomas en memoria lo cual da la posibilidad de trabajar con conjuntos de datos mucho más grandes. Incluso se podría pensar en trabajar contra bases de datos de pangenomas basadas en servicios (*server-based*), como por ejemplo del tipo PosgreSQL, o MySQL, si se encontrara la forma de crear una base de datos masiva estandarizada para pangenomas bacterianos.

En el caso de `pewit`, además de terminar los análisis propuestos para el manuscrito, una perspectiva sería reimplementar el algoritmo adaptándolo para tecnologías *cloud* para permitir un escalado masivo, con miras al minado de pangenomas a partir de metagenomas disponibles públicamente. Por otro lado, los nuevos métodos de plegado computacional de proteínas podrían ayudar a identificar homología a niveles más profundos en la taxonomía.

Las personas equilibradas son así,
acostumbran a simplificarlo todo,
y después, pero siempre demasiado tarde,
las vemos asombrándose
de la copiosa diversidad de la vida (...)

— ***José Saramago***
«El Hombre Duplicado»

Bibliografía

[1] Allen D. Roses. «The genome era begins...» En: *Nature Genetics* 33.S3 (mar. de 2003), págs. 217-217. DOI: 10.1038/ng1110.

[2] Alan E. Guttmacher y Francis S. Collins. «Welcome to the Genomic Era». En: *New England Journal of Medicine* 349.10 (sep. de 2003), págs. 996-998. DOI: 10.1056/NEJMe038132.

[3] Robert D. Fleischmann et al. «Whole-Genome Random Sequencing and Assembly of *Haemophilus influenzae* Rd». En: *Science* 269.5223 (jul. de 1995), págs. 496-512. DOI: 10.1126/science.7542800.

[4] Hervé Tettelin et al. «Complete Genome Sequence of *Neisseria meningitidis* Serogroup B Strain MC58». En: *Science* 287.5459 (mar. de 2000), págs. 1809-1815. DOI: 10.1126/science.287.5459.1809.

[5] Mariagrazia Pizza et al. «Identification of Vaccine Candidates Against Serogroup B Meningococcus by Whole-Genome Sequencing». En: *Science* 287.5459 (mar. de 2000), págs. 1816-1820. DOI: 10.1126/science.287.5459.1816.

[6] Hervé Tettelin et al. «Complete Genome Sequence of a Virulent Isolate of *Streptococcus pneumoniae*». En: *Science* 293.5529 (jul. de 2001), págs. 498-506. DOI: 10.1126/science.1061217.

[7] Philippe Glaser et al. «Genome sequence of Streptococcus agalactiae, a pathogen causing invasive neonatal disease: Genome sequence of Streptococcus agalactiae». En: *Molecular Microbiology* 45.6 (sep. de 2002), págs. 1499-1513. DOI: 10.1046/j.1365-2958.2002.03126.x.

[8] A. J. Martinez-Murcia, S. Benlloch y M. D. Collins. «Phylogenetic Interrelationships of Members of the Genera Aeromonas and Plesiomonas as Determined by 16S Ribosomal DNA Sequencing: Lack of Congruence with Results of DNA-DNA Hybridizations». En: *International Journal of Systematic Bacteriology* 42.3 (jul. de 1992), págs. 412-421. DOI: 10.1099/00207713-42-3-412.

[9] R. A. Welch et al. «Extensive mosaic structure revealed by the complete genome sequence of uropathogenic *Escherichia coli*». En: *Proceedings of the National Academy of Sciences* 99.26 (dic. de 2002), págs. 17020-17024. DOI: 10.1073/pnas.252529799.

[10] Hervé Tettelin et al. «Genome analysis of multiple pathogenic isolates of Streptococcus agalactiae: implications for the microbial "pan-genome"». En: *Proceedings of the National Academy of Sciences of the United States of America* 102.39 (sep. de 2005), págs. 13950-13955. DOI: 10.1073/pnas.0506758102.

[11] Carl R. Woese y George E. Fox. «The concept of cellular evolution». En: *Journal of Molecular Evolution* 10.1 (mar. de 1977), págs. 1-6. DOI: 10.1007/BF01796132.

[12] C R Woese. «Bacterial evolution». En: *Microbiological Reviews* 51.2 (jun. de 1987), págs. 221-271. DOI: 10.1128/mr.51.2.221-271.1987.

[13] O Kandler. «The early diversification of life.» En: *Early Life on Earth Nobel Symposium 84*. Columbia University Press, New York, 1993, págs. 152-160.

[14] OTTO Kandler. «The early diversification of life and the origin of the three domains: a proposal». En: *Thermophiles: the keys to molecular evolution and the origin of life* (1998), págs. 19-31.

[15] Jeffrey G Lawrence. «Gene transfer and minimal genome size». En: *Size limits of very small microorganisms, Proceedings of a Workshop. National Academy of Sciences*. 1999, págs. 32-38.

[16] H. S. Heaps. *Information retrieval, computational and theoretical aspects*. Academic Press, 1978.

[17] Hervé Tettelin et al. «Comparative genomics: the bacterial pan-genome». En: *Current Opinion in Microbiology* 11.5 (oct. de 2008), págs. 472-477.

[18] Justin S. Hogg et al. «Characterization and modeling of the Haemophilus influenzae core and supragenomes based on the complete genomic sequences of Rd and 12 clinical nontypeable strains». En: *Genome Biology* 8.6 (2007), R103. DOI: 10.1186/gb-2007-8-6-r103.

[19] Lars Snipen, Trygve Almøy y David W. Ussery. «Microbial comparative pan-genomics using binomial mixture models». En: *BMC genomics* 10 (ago. de 2009), pág. 385. DOI: 10.1186/1471-2164-10-385.

[20] Jason C. Hyun, Jonathan M. Monk y Bernhard O. Palsson. «Comparative pangenomics: analysis of 12 microbial pathogen pangenomes reveals conserved global structures of genetic and functional diversity». En: *BMC Genomics* 23.1 (dic. de 2022), pág. 7. DOI: 10.1186/s12864-021-08223-8.

[21] Andrey O Kislyuk et al. «Genomic fluidity: an integrative view of gene diversity within microbial populations». En: *BMC Genomics* 12 (ene. de 2011), pág. 32. DOI: 10.1186/1471-2164-12-32.

[22] Nadia Andrea Andreani, Elze Hesse y Michiel Vos. «Prokaryote genome fluidity is dependent on effective population size». En: *The ISME Journal* 11.7 (jul. de 2017), págs. 1719-1721. DOI: 10.1038/ismej.2017.36.

[23] Louis-Marie Bobay y Howard Ochman. «Factors driving effective population size and pan-genome evolution in bacteria». En: *BMC Evolutionary Biology* 18.1 (dic. de 2018), pág. 153. DOI: 10.1186/s12862-018-1272-4.

[24] Franz Baumdicker, Wolfgang R. Hess y Peter Pfaffelhuber. «The infinitely many genes model for the distributed genome of bacteria». En: *Genome Biology and Evolution* 4.4 (2012), págs. 443-456. DOI: 10.1093/gbe/evs016.

[25] Motoo Kimura y James F Crow. «The number of alleles that can be maintained in a finite population.» En: *Genetics* 49.4 (abr. de 1964), págs. 725-738. DOI: 10.1093/genetics/49.4.725.

[26] Motoo Kimura. «The number of heterozygous nucleotide sites maintained in a finite population due to steady flux of mutations.» En: *Genetics* 61.4 (abr. de 1969), págs. 893-903. DOI: 10.1093/genetics/61.4.893.

[27] R. Eric Collins y Paul G. Higgs. «Testing the Infinitely Many Genes Model for the Evolution of the Bacterial Core Genome and Pangenome». En: *Molecular Biology and Evolution* 29.11 (nov. de 2012), págs. 3413-3425. DOI: 10.1093/molbev/mss163.

[28] Franz Baumdicker y Peter Pfaffelhuber. «The infinitely many genes model with horizontal gene transfer». En: *Electronic Journal of Probability* 19.none (ene. de 2014). DOI: 10.1214/EJP.v19-2642.

[29] Leigh M. van Valen. «A new evolutionary law». En: 1973.

[30] Michiel Vos et al. «Rates of Lateral Gene Transfer in Prokaryotes: High but Why?» En: *Trends in Microbiology* 23.10 (oct. de 2015), págs. 598-605. DOI: 10.1016/j.tim.2015.07.006.

[31] Ming-Chun Lee y Christopher J. Marx. «Repeated, Selection-Driven Genome Reduction of Accessory Genes in Experimental Populations». En: *PLoS Genetics* 8.5 (mayo de 2012). Ed. por Nancy A. Moran, e1002651. DOI: 10.1371/journal.pgen.1002651.

[32] Alan McNally et al. «Combined Analysis of Variation in Core, Accessory and Regulatory Genome Regions Provides a Super-Resolution View into the Evolution of Bacterial Populations». En: *PLOS Genetics* 12.9 (sep. de 2016). Ed. por Diarmaid Hughes, e1006280. DOI: 10.1371/journal.pgen.1006280.

[33] Akshit Goyal. «Metabolic adaptations underlying genome flexibility in prokaryotes». En: *PLOS Genetics* 14.10 (oct. de 2018). Ed. por Diarmaid Hughes, e1007763. DOI: 10.1371/journal.pgen.1007763.

[34] Alexander E. Lobkovsky, Yuri I. Wolf y Eugene V. Koonin. «Gene Frequency Distributions Reject a Neutral Model of Genome Evolution». En: *Genome Biology and Evolution* 5.1 (ene. de 2013), págs. 233-242. DOI: 10.1093/gbe/evt002.

[35] Itamar Sela, Yuri I. Wolf y Eugene V. Koonin. «Theory of prokaryotic genome evolution». En: *Proceedings of the National Academy of Sciences* 113.41 (oct. de 2016), págs. 11399-11407. DOI: 10.1073/pnas.1614083113.

[36] James O. McInerney, Alan McNally y Mary J. O'Connell. «Why prokaryotes have pangenomes». En: *Nature Microbiology* 2.4 (abr. de 2017), pág. 17040. DOI: 10.1038/nmicrobiol.2017.40.

[37] Maria Rosa Domingo-Sananes y James O. McInerney. *Selection-based model of prokaryote pangenomes*. preprint. Microbiology, sep. de 2019. DOI: 10.1101/782573.

[38] Fiona J Whelan, Rebecca J Hall y James O McInerney. «Evidence for Selection in the Abundant Accessory Gene Content of a Prokaryote Pangenome». En: *Molecular Biology and Evolution* 38.9 (ago. de 2021). Ed. por Deepa Agashe, págs. 3697-3708. DOI: 10.1093/molbev/msab139.

[39] James O McInerney. «Prokaryotic Pangenomes Act as Evolving Ecosystems». En: *Molecular Biology and Evolution* (oct. de 2022). Ed. por Brandon Gaut, msac232. DOI: 10.1093/molbev/msac232.

[40] Elizabeth A Cummins et al. «Prokaryote pangenomes are dynamic entities». En: *Current Opinion in Microbiology* 66 (abr. de 2022), págs. 73-78. DOI: 10.1016/j.mib.2022.01.005.

[41] Eduardo P C Rocha. «Neutral Theory, Microbial Practice: Challenges in Bacterial Population Genetics». En: *Molecular Biology and Evolution* 35.6 (jun. de 2018). Ed. por Sudhir Kumar, págs. 1338-1347. DOI: 10.1093/molbev/msy078.

[42] Motoo Kimura. «Evolutionary Rate at the Molecular Level». En: *Nature* 217.5129 (feb. de 1968), págs. 624-626. DOI: 10.1038/217624a0.

[43] B. Jesse Shapiro. «The population genetics of pangenomes». En: *Nature Microbiology* 2.12 (dic. de 2017), págs. 1574-1574. DOI: 10.1038/s41564-017-0066-6.

[44] Gavin M Douglas y B Jesse Shapiro. «Genic Selection Within Prokaryotic Pangenomes». En: *Genome Biology and Evolution* 13.11 (nov. de 2021). Ed. por Adam Eyre-Walker, evab234. DOI: 10.1093/gbe/evab234.

[45] J. Jeffrey Morris, Richard E. Lenski y Erik R. Zinser. «The Black Queen Hypothesis: Evolution of Dependencies through Adaptive Gene Loss». En: *mBio* 3.2 (mayo de 2012), e00036-12. DOI: 10.1128/mBio.00036-12.

[46] J. Jeffrey Morris. «Black Queen evolution: the role of leakiness in structuring microbial communities». En: *Trends in Genetics* 31.8 (ago. de 2015), págs. 475-482. DOI: 10.1016/j.tig.2015.05.004.

[47] Matthew S. Fullmer, Shannon M. Soucy y Johann Peter Gogarten. «The pan-genome as a shared genomic resource: mutual cheating, cooperation and the black queen hypothesis». En: *Frontiers in Microbiology* 6 (jul. de 2015). DOI: 10.3389/fmicb.2015.00728.

[48] Carl R. Woese. «On the evolution of cells». En: *Proceedings of the National Academy of Sciences* 99.13 (jun. de 2002), págs. 8742-8747. DOI: 10.1073/pnas.132266999.

[49] James O McInerney et al. «The public goods hypothesis for the evolution of life on Earth». En: *Biology Direct* 6.1 (2011), pág. 41. DOI: 10.1186/1745-6150-6-41.

[50] James O. McInerney y Douglas H. Erwin. «The role of public goods in planetary evolution». En: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 375.2109 (dic. de 2017), pág. 20160359. DOI: 10.1098/rsta.2016.0359.

[51] James O. McInerney et al. «Pangenomes and Selection: The Public Goods Hypothesis». En: *The Pangenome*. Ed. por Hervé Tettelin y Duccio Medini. Cham: Springer International Publishing, 2020, págs. 151-167. DOI: 10.1007/978-3-030-38281-0_7.

[52] Paul A Samuelson. «The pure theory of public expenditure». En: *The review of economics and statistics* (1954), págs. 387-389.

[53] Richard A Musgrave. *The theory of public finance; a study in public economy*. Kogakusha Co., 1959.

[54] Vincent Ostrom y Elinor Ostrom. «A theory for institutional analysis of common pool problems». En: *Managing the commons* 157 (1977).

[55] James M. Buchanan. «An Economic Theory of Clubs». En: *Economica* 32.125 (feb. de 1965), pág. 1. DOI: 10.2307/2552442.

[56] Andrew J. Page et al. «Roary: rapid large-scale prokaryote pan genome analysis». En: *Bioinformatics* 31.22 (nov. de 2015), págs. 3691-3693. DOI: 10.1093/bioinformatics/btv421.

[57] Gal Horesh et al. «Different evolutionary trends form the twilight zone of the bacterial pan-genome». En: *Microbial Genomics* 7.9 (sep. de 2021). DOI: 10.1099/mgen.0.000670.

[58] Jukka Corander y Pekka Marttinen. «Bayesian identification of admixture events using multilocus molecular markers: BAYESIAN IDENTIFICATION OF ADMIXTURE EVENTS». En: *Molecular Ecology* 15.10 (jun. de 2006), págs. 2833-2843. DOI: 10.1111/j.1365-294X.2006.02994.x.

[59] John A. Lees et al. «Fast and flexible bacterial genomic epidemiology with PopPUNK». En: *Genome Research* 29.2 (feb. de 2019), págs. 304-316. DOI: 10.1101/gr.241455.118.

[60] A R Mushegian y E V Koonin. «A minimal gene set for cellular life derived by comparison of complete bacterial genomes.» En: *Proceedings of the National Academy of Sciences* 93.19 (sep. de 1996), págs. 10268-10273. DOI: 10.1073/pnas.93.19.10268.

[61] Clyde A. Hutchison et al. «Global Transposon Mutagenesis and a Minimal Mycoplasma Genome». En: *Science* 286.5447 (dic. de 1999), págs. 2165-2169. DOI: 10.1126/science.286.5447.2165.

[62] Brian J. Akerley et al. «Systematic identification of essential genes by *in vitro mariner* mutagenesis». En: *Proceedings of the National Academy of Sciences* 95.15 (jul. de 1998), págs. 8927-8932. DOI: 10.1073/pnas.95.15.8927.

[63] Karl A. Reich, Linda Chovan y Paul Hessler. «Genome Scanning in Haemophilus influenzae for Identification of Essential Genes». En: *Journal of Bacteriology* 181.16 (ago. de 1999), págs. 4961-4968. DOI: 10.1128/JB.181.16.4961-4968.1999.

[64] Christopher M. Sassetti, Dana H. Boyd y Eric J. Rubin. «Comprehensive identification of conditionally essential genes in mycobacteria». En: *Proceedings of the National Academy of Sciences* 98.22 (oct. de 2001), págs. 12712-12717. DOI: 10.1073/pnas.231275498.

[65] Guri Giaever et al. «Functional profiling of the Saccharomyces cerevisiae genome». En: *Nature* 418.6896 (jul. de 2002), págs. 387-391. DOI: 10.1038/nature00935.

[66] Tim van Opijnen, Kip L Bodi y Andrew Camilli. «Tn-seq: high-throughput parallel sequencing for fitness and genetic interaction studies in microorganisms». En: *Nature Methods* 6.10 (oct. de 2009), págs. 767-772. DOI: 10.1038/nmeth.1377.

[67] Gemma C. Langridge et al. «Simultaneous assay of every *Salmonella* Typhi gene using one million transposon mutants». En: *Genome Research* 19.12 (dic. de 2009), págs. 2308-2316. DOI: 10.1101/gr.097097.109.

[68] Andrew L. Goodman et al. «Identifying Genetic Determinants Needed to Establish a Human Gut Symbiont in Its Habitat». En: *Cell Host & Microbe* 6.3 (sep. de 2009), págs. 279-289. DOI: 10.1016/j.chom.2009.08.003.

[69] Jeffrey D. Gawronski et al. «Tracking insertion mutants within libraries by deep sequencing and a genome-wide screen for *Haemophilus* genes required in the lung». En: *Proceedings of the National Academy of Sciences* 106.38 (sep. de 2009), págs. 16422-16427. DOI: 10.1073/pnas.0906627106.

[70] Matthew H Larson et al. «CRISPR interference (CRISPRi) for sequence-specific control of gene expression». En: *Nature Protocols* 8.11 (nov. de 2013), págs. 2180-2196. DOI: 10.1038/nprot.2013.132.

[71] Jason M. Peters et al. «A Comprehensive, CRISPR-based Functional Analysis of Essential Genes in Bacteria». En: *Cell* 165.6 (jun. de 2016), págs. 1493-1506. DOI: 10.1016/j.cell.2016.05.003.

[72] Bradley E. Poulsen et al. «Defining the core essential genome of *Pseudomonas aeruginosa*». En: *Proceedings of the National Academy of Sciences* 116.20 (mayo de 2019), págs. 10072-10080. DOI: 10.1073/pnas.1900570116.

[73] Kouichi Ozaki et al. «Functional SNPs in the lymphotoxin-alpha gene that are associated with susceptibility to myocardial infarction». En: *Nature Genetics* 32.4 (dic. de 2002), págs. 650-654. DOI: 10.1038/ng1047.

[74] Sarah G. Earle et al. «Identifying lineage effects when controlling for population structure improves power in bacterial association studies». En: *Nature Microbiology* 1.5 (abr. de 2016), pág. 16041. DOI: 10.1038/nmicrobiol.2016.41.

[75] John A Lees et al. «pyseer: a comprehensive tool for microbial pangenome-wide association studies». En: *Bioinformatics* 34.24 (dic. de 2018). Ed. por Oliver Stegle, págs. 4310-4312. DOI: 10.1093/bioinformatics/bty539.

[76] Francesc Coll et al. «PowerBacGWAS: a computational pipeline to perform power calculations for bacterial genome-wide association studies». En: *Communications Biology* 5.1 (mar. de 2022), pág. 266. DOI: 10.1038/s42003-022-03194-2.

[77] Ola Brynildsrud et al. «Rapid scoring of genes in microbial pan-genome-wide association studies with Scoary». En: *Genome Biology* 17.1 (dic. de 2016), pág. 238. DOI: 10.1186/s13059-016-1108-8.

[78] Jo Handelsman et al. «Molecular biological access to the chemistry of unknown soil microbes: a new frontier for natural products». En: *Chemistry & Biology* 5.10 (oct. de 1998), R245-R249. DOI: 10.1016/S1074-5521(98)90108-9.

[79] Nadav Kashtan et al. «Single-Cell Genomics Reveals Hundreds of Coexisting Subpopulations in Wild *Prochlorococcus*». En: *Science* 344.6182 (abr. de 2014), págs. 416-420. DOI: 10.1126/science.1248575.

[80] Tom O. Delmont y A. Murat Eren. «Linking pangenomes and metagenomes: the *Prochlorococcus* metapangenome». En: *PeerJ* 6 (ene. de 2018), e4320. DOI: 10.7717/peerj.4320.

[81] Bing Ma et al. «A comprehensive non-redundant gene catalog reveals extensive within-community intraspecies diversity in the human vagina». En: *Nature Communications* 11.1 (feb. de 2020), pág. 940. DOI: 10.1038/s41467-020-14677-3.

[82] Bing Ma, Michael France y Jacques Ravel. «Meta-Pangenome: At the Crossroad of Pangenomics and Metagenomics». En: *The Pangenome*. Ed. por Hervé Tettelin y Duccio Medini. Cham: Springer International Publishing, 2020, págs. 205-218. DOI: 10.1007/978-3-030-38281-0_9.

[83] Afrah Shafquat et al. «Functional and phylogenetic assembly of microbial communities in the human microbiome». En: *Trends in Microbiology* 22.5 (mayo de 2014), págs. 261-266. DOI: 10.1016/j.tim.2014.01.011.

[84] Agnieszka A. Golicz et al. «Pangenomics Comes of Age: From Bacteria to Plant and Animal Applications». En: *Trends in Genetics* 36.2 (feb. de 2020), págs. 132-145. DOI: 10.1016/j.tig.2019.11.006.

[85] Walter M. Fitch. «Distinguishing Homologous from Analogous Proteins». En: *Systematic Biology* 19.2 (jun. de 1970), págs. 99-113. DOI: 10.2307/2412448.

[86] Erik L. L. Sonnhammer y Eugene V. Koonin. «Orthology, paralogy and proposed classification for paralog subtypes». En: *Trends in Genetics* 18.12 (dic. de 2002), págs. 619-620. DOI: 10.1016/S0168-9525(02)02793-2.

[87] Susumu Ohno. *Evolution by Gene Duplication*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1970. DOI: 10.1007/978-3-642-86659-3.

[88] Fredj Tekaia. «Inferring Orthologs: Open Questions and Perspectives». En: *Genomics Insights* 9 (2016), págs. 17-28. DOI: 10.4137/GEI.S37925.

[89] S. F. Altschul et al. «Basic local alignment search tool». En: *Journal of Molecular Biology* 215.3 (oct. de 1990), págs. 403-410. DOI: 10.1016/S0022-2836(05)80360-2.

[90] Li Li, Christian J. Stoeckert y David S. Roos. «OrthoMCL: Identification of Ortholog Groups for Eukaryotic Genomes». En: *Genome Research* 13.9 (sep. de 2003), págs. 2178-2189. DOI: 10.1101/gr.1224503.

[91] Jochen Blom et al. «EDGAR: A software framework for the comparative analysis of prokaryotic genomes». En: *BMC Bioinformatics* 10.1 (dic. de 2009), pág. 154. DOI: 10.1186/1471-2105-10-154.

[92] Bruno Contreras-Moreira y Pablo Vinuesa. «GET_HOMOLOGUES, a Versatile Software Package for Scalable and Robust Microbial Pangenome Analysis». En: *Applied and Environmental Microbiology* 79.24 (dic. de 2013), págs. 7696-7701. DOI: 10.1128/AEM.02411-13.

[93] Lars Snipen y Kristian Hovde Liland. «micropan: an R-package for microbial pan-genomics». En: *BMC bioinformatics* 16 (mar. de 2015), pág. 79. DOI: 10.1186/s12859-015-0517-0.

[94] Benjamin Buchfink, Chao Xie y Daniel H Huson. «Fast and sensitive protein alignment using DIAMOND». En: *Nature Methods* 12.1 (ene. de 2015), págs. 59-60. DOI: 10.1038/nmeth.3176.

[95] Wei Ding, Franz Baumdicker y Richard A Neher. «panX: pan-genome analysis and exploration». En: *Nucleic Acids Research* 46.1 (ene. de 2018), e5. DOI: 10.1093/nar/gkx977.

[96] Maria C. Rivera et al. «Genomic evidence for two functionally distinct gene classes». En: *Proceedings of the National Academy of Sciences of the United States of America* 95.11 (mayo de 1998), págs. 6239-6244.

[97] L. B. Koski y G. B. Golding. «The closest BLAST hit is often not the nearest neighbor». En: *Journal of Molecular Evolution* 52.6 (jun. de 2001), págs. 540-542. DOI: 10.1007/s002390010184.

[98] A. J. Enright, S. Van Dongen y C. A. Ouzounis. «An efficient algorithm for large-scale detection of protein families». En: *Nucleic Acids Research* 30.7 (abr. de 2002), págs. 1575-1584.

[99] Toni Gabaldón y Eugene V. Koonin. «Functional and evolutionary implications of gene orthology». En: *Nature Reviews. Genetics* 14.5 (mayo de 2013), págs. 360-366. DOI: 10.1038/nrg3456.

[100] Weizhong Li y Adam Godzik. «Cd-hit: a fast program for clustering and comparing large sets of protein or nucleotide sequences». En: *Bioinformatics (Oxford, England)* 22.13 (jul. de 2006), págs. 1658-1659. DOI: 10.1093/bioinformatics/btl158.

[101] Limin Fu et al. «CD-HIT: accelerated for clustering the next-generation sequencing data». En: *Bioinformatics* 28.23 (dic. de 2012), págs. 3150-3152. DOI: 10.1093/bioinformatics/bts565.

[102] Martin Steinegger y Johannes Söding. «Clustering huge protein sequence sets in linear time». En: *Nature Communications* 9.1 (jun. de 2018), pág. 2542. DOI: 10.1038/s41467-018-04964-5.

[103] Martin Steinegger y Johannes Söding. «MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets». En: *Nature Biotechnology* 35.11 (nov. de 2017), págs. 1026-1028. DOI: 10.1038/nbt.3988.

[104] Gerry Tonkin-Hill et al. «Producing polished prokaryotic pangenomes with the Panaroo pipeline». En: *Genome Biology* 21.1 (jul. de 2020), pág. 180. DOI: 10.1186/s13059-020-02090-4.

[105] Guillaume Gautreau et al. «PPanGGOLiN: Depicting microbial diversity via a partitioned pangenome graph». En: *PLOS Computational Biology* 16.3 (mar. de 2020). Ed. por Christos A. Ouzounis, e1007732. DOI: 10.1371/journal.pcbi.1007732.

[106] Toni Gabaldón. «Large-scale assignment of orthology: back to phylogenetics?» En: *Genome Biology* 9.10 (oct. de 2008), pág. 235. DOI: 10.1186/gb-2008-9-10-235.

[107] Yongbing Zhao et al. «PGAP: pan-genomes analysis pipeline». En: *Bioinformatics* 28.3 (feb. de 2012), págs. 416-418. DOI: 10.1093/bioinformatics/btr655.

[108] Andrey Alexeyenko et al. «Automatic clustering of orthologs and inparalogs shared by multiple proteomes». En: *Bioinformatics* 22.14 (jul. de 2006), e9-e15. DOI: 10.1093/bioinformatics/btl213.

[109] R Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2016.

[110] S. R. Eddy. «Multiple alignment using hidden Markov models». En: *Proceedings. International Conference on Intelligent Systems for Molecular Biology* 3 (1995), págs. 114-120.

[111] Sean R. Eddy. «Accelerated Profile HMM Searches». En: *PLoS Computational Biology* 7.10 (oct. de 2011). DOI: 10.1371/journal.pcbi.1002195.

[112] E. L. Sonnhammer, S. R. Eddy y R. Durbin. «Pfam: a comprehensive database of protein domain families based on seed alignments». En: *Proteins* 28.3 (jul. de 1997), págs. 405-420.

[113] Alex Bateman et al. «The Pfam Protein Families Database». En: *Nucleic Acids Research* 30.1 (ene. de 2002), págs. 276-280. DOI: 10.1093/nar/30.1.276.

[114] Narendrakumar M. Chaudhari, Vinod Kumar Gupta y Chitra Dutta. «BPGA-an ultra-fast pan-genome analysis pipeline». En: *Scientific Reports* 6.1 (abr. de 2016), pág. 24373. DOI: 10.1038/srep24373.

[115] Robert C. Edgar. «Search and clustering orders of magnitude faster than BLAST». En: *Bioinformatics* 26.19 (oct. de 2010), págs. 2460-2461. DOI: 10.1093/bioinformatics/btq461.

[116] M. N. Price, P. S. Dehal y A. P. Arkin. «FastTree: Computing Large Minimum Evolution Trees with Profiles instead of a Distance Matrix». En: *Molecular Biology and Evolution* 26.7 (jul. de 2009), págs. 1641-1650. DOI: 10.1093/molbev/msp077.

[117] Sion C. Bayliss et al. «PIRATE: A fast and scalable pangenomics toolbox for clustering diverged orthologues in bacteria». En: *GigaScience* 8.10 (oct. de 2019), giz119. DOI: 10.1093/gigascience/giz119.

[118] Thomas Lin Pedersen et al. «PanViz: interactive visualization of the structure of functionally annotated pangenomes». En: *Bioinformatics* 33.7 (abr. de 2017). Ed. por John Hancock, págs. 1081-1082. DOI: 10.1093/bioinformatics/btw761.

[119] M. Bostock, V. Ogievetsky y J. Heer. «D^3 Data-Driven Documents». En: *IEEE Transactions on Visualization and Computer Graphics* 17.12 (dic. de 2011), págs. 2301-2309. DOI: 10.1109/TVCG.2011.185.

[120] Thomas H. Clarke et al. «PanACEA: a bioinformatics tool for the exploration and visualization of bacterial pan-chromosomes». En: *BMC Bioinformatics* 19.1 (jun. de 2018), pág. 246. DOI: 10.1186/s12859-018-2250-y.

[121] A. Murat Eren et al. «Community-led, integrated, reproducible multi-omics with anvi'o». En: *Nature Microbiology* 6.1 (dic. de 2020), págs. 3-6. DOI: 10.1038/s41564-020-00834-3.

[122] Alexis Dereeper, Marilyne Summo y Damien F Meyer. «PanExplorer: a web-based tool for exploratory analysis and visualization of bacterial pan-genomes». En: *Bioinformatics* 38.18 (sep. de 2022). Ed. por Tobias Marschall, págs. 4412-4414. DOI: 10.1093/bioinformatics/btac504.

[123] Michael A. Brockhurst et al. «The Ecology and Evolution of Pangenomes». En: *Current Biology* 29.20 (oct. de 2019), R1094-R1103. DOI: 10.1016/j.cub.2019.08.012.

[124] Joseph Felsenstein. *Inferring phylogenies*. Sunderland, Mass: Sinauer Associates, 2004.

[125] Siméon-Denis Poisson. *Recherches sur la probabilité des jugements en matière criminelle et en matière civile: précédées des règles générales du calcul des probabilités*. Bachelier, 1837.

[126] Ignacio Ferrés, Pablo Fresia y Gregorio Iraola. «simurg: simulate bacterial pangenomes in R». En: *Bioinformatics* 36.4 (feb. de 2020), págs. 1273-1274. DOI: 10.1093/bioinformatics/btz735.

[127] Holling Clancy Holling y Lucille Webster Holling. *Pagoo*. Boston: Houghton Mifflin Co, 1985.

[128] Ignacio Ferrés y Gregorio Iraola. «An object-oriented framework for evolutionary pangenome analysis». En: *Cell Reports Methods* 1.5 (sep. de 2021), pág. 100085. DOI: 10.1016/j.crmeth.2021.100085.

[129] Ignacio Ferrés y Gregorio Iraola. «Protocol for post-processing of bacterial pangenome data using Pagoo pipeline». En: *STAR Protocols* 2.4 (dic. de 2021), pág. 100802. DOI: 10.1016/j.xpro.2021.100802.

[130] Daiana Mir et al. «Recurrent Dissemination of SARS-CoV-2 Through the Uruguayan–Brazilian Border». En: *Frontiers in Microbiology* 12 (mayo de 2021), pág. 653986. DOI: 10.3389/fmicb.2021.653986.

[131] Pilar Moreno et al. *An effective COVID-19 response in South America: the Uruguayan Conundrum*. preprint. Infectious Diseases (except HIV/AIDS), jul. de 2020. DOI: 10.1101/2020.07.24.20161802.

[132] Cecilia Salazar et al. *Multiple introductions, regional spread and local differentiation during the first week of COVID-19 epidemic in Montevideo, Uruguay*. preprint. Microbiology, mayo de 2020. DOI: 10.1101/2020.05.09.086223.

[133] Natalia Rego et al. «Real-Time Genomic Surveillance for SARS-CoV-2 Variants of Concern, Uruguay». En: *Emerging Infectious Diseases* 27.11 (nov. de 2021), págs. 2957-2960. DOI: 10.3201/eid2711.211198.

[134] Natalia Rego et al. «Emergence and Spread of a B.1.1.28-Derived P.6 Lineage with Q675H and Q677H Spike Mutations in Uruguay». En: *Viruses* 13.9 (sep. de 2021), pág. 1801. DOI: 10.3390/v13091801.

[135] Cecilia Salazar et al. «Case Report: Early Transcontinental Import of SARS-CoV-2 Variant of Concern 202012/01 (B.1.1.7) From Europe to Uruguay». En: *Frontiers in Virology* 1 (mayo de 2021), pág. 685618. DOI: 10.3389/fviro.2021.685618.

[136] Jessica H. Fong et al. «Modeling the Evolution of Protein Domain Architectures Using Maximum Parsimony». En: *Journal of Molecular Biology* 366.1 (feb. de 2007), págs. 307-315. DOI: 10.1016/j.jmb.2006.11.017.

[137] Byungwook Lee y Doheon Lee. «Protein comparison at the domain architecture level». En: *BMC bioinformatics* 10 Suppl 15 (dic. de 2009), S5. DOI: 10.1186/1471-2105-10-S15-S5.

[138] Kristoffer Forslund, Isabella Pekkari y Erik LL Sonnhammer. «Domain architecture conservation in orthologs». En: *BMC Bioinformatics* 12 (ago. de 2011), pág. 326. DOI: 10.1186/1471-2105-12-326.

[139] Piotr Indyk y Rajeev Motwani. «Approximate nearest neighbors: towards removing the curse of dimensionality». En: *Proceedings of the thirtieth annual ACM symposium on Theory of computing*. STOC '98. New York, NY, USA: Association for Computing Machinery, mayo de 1998, págs. 604-613. DOI: 10.1145/276698.276876.

[140] Aristides Gionis, Piotr Indyk y Rajeev Motwani. «Similarity search in high dimensions via hashing». En: *Vldb*. Vol. 99. 1999, págs. 518-529.

[141] H. Pagès et al. *Biostrings: Efficient manipulation of biological strings*. Bioconductor version: Release (3.15). 2022. DOI: 10.18129/B9.bioc.Biostrings.

[142] Lincoln Mullen. *textreuse: Detect Text Reuse and Document Similarity*. Mayo de 2020.

[143] Erik S. Wright. «DECIPHER: harnessing local sequence context to improve protein multiple sequence alignment». En: *BMC Bioinformatics* 16.1 (dic. de 2015), pág. 322. DOI: 10.1186/s12859-015-0749-z.

[144] N. Saitou y M. Nei. «The neighbor-joining method: a new method for reconstructing phylogenetic trees.» En: *Molecular Biology and Evolution* 4.4 (jul. de 1987), págs. 406-425. DOI: 10.1093/oxfordjournals.molbev.a040454.

[145] Emmanuel Paradis, Julien Claude y Korbinian Strimmer. «APE: Analyses of Phylogenetics and Evolution in R language». En: *Bioinformatics (Oxford, England)* 20.2 (ene. de 2004), págs. 289-290.

[146] Klaus Peter Schliep. «phangorn: phylogenetic analysis in R». En: *Bioinformatics (Oxford, England)* 27.4 (feb. de 2011), págs. 592-593. DOI: 10.1093/bioinformatics/btq706.

[147] Hadley Wickham. *ggplot2*. Use R! Cham: Springer International Publishing, 2016. DOI: 10.1007/978-3-319-24277-4.

[148] Gregory M. Kurtzer, Vanessa Sochat y Michael W. Bauer. «Singularity: Scientific containers for mobility of compute». En: *PLOS ONE* 12.5 (mayo de 2017), e0177459. DOI: 10.1371/journal.pone.0177459.

[149] Felipe da Veiga Leprevost et al. «BioContainers: an open-source and community-driven framework for software standardization». En: *Bioinformatics* 33.16 (ago. de 2017), págs. 2580-2582. DOI: 10.1093/bioinformatics/btx192.

[150] Andrew J. Page et al. «GFF3toEMBL: Preparing annotated assemblies for submission to EMBL». En: *Journal of Open Source Software* 1.6 (oct. de 2016), pág. 80. DOI: 10.21105/joss.00080.

[151] Gregorio Iraola et al. «Distinct Campylobacter fetus lineages adapted as livestock pathogens and human pathobionts in the intestinal microbiota». En: *Nature Communications* 8.1 (nov. de 2017), pág. 1367. DOI: 10.1038/s41467-017-01449-9.

[152] Lam-Tung Nguyen et al. «IQ-TREE: A Fast and Effective Stochastic Algorithm for Estimating Maximum-Likelihood Phylogenies». En: *Molecular Biology and Evolution* 32.1 (ene. de 2015), págs. 268-274. DOI: 10.1093/molbev/msu300.

[153] David J. Williams et al. «The genus Serratia revisited by genomics». En: *Nature Communications* 13.1 (sep. de 2022), pág. 5195. DOI: 10.1038/s41467-022-32929-2.

[154] Guangchuang Yu et al. «GGTREE : an R package for visualization and annotation of phylogenetic trees with their covariates and other associated data». En: *Methods in Ecology and Evolution* 8.1 (ene. de 2017). Ed. por Greg McInerny, págs. 28-36. DOI: 10.1111/2041-210X.12628.

[155] Torsten Seemann. «Prokka: rapid prokaryotic genome annotation». En: *Bioinformatics (Oxford, England)* 30.14 (jul. de 2014), págs. 2068-2069. DOI: 10.1093/bioinformatics/btu153.

[156] Benjamin Buchfink, Klaus Reuter y Hajk-Georg Drost. «Sensitive protein alignments at tree-of-life scale using DIAMOND». En: *Nature Methods* 18.4 (abr. de 2021), págs. 366-368. DOI: 10.1038/s41592-021-01101-x.

[157] B. W. Matthews. «Comparison of the predicted and observed secondary structure of T4 phage lysozyme». En: *Biochimica et Biophysica Acta (BBA) - Protein Structure* 405.2 (oct. de 1975), págs. 442-451. DOI: 10.1016/0005-2795(75)90109-9.

[158] Maria Rosa Domingo-Sananes y James O. McInerney. «Mechanisms That Shape Microbial Pangenomes». En: *Trends in Microbiology* 29.6 (jun. de 2021), págs. 493-503. DOI: 10.1016/j.tim.2020.12.004.

[159] Roman Thibeaux et al. «Deciphering the unexplored Leptospira diversity from soils uncovers genomic evolution to virulence». En: *Microbial Genomics* 4.1 (2018). DOI: 10.1099/mgen.0.000144.

[160] Daniela Costa et al. «Pangenome analysis reveals genetic isolation in Campylobacter hyointestinalis subspecies adapted to different mammalian hosts». En: *Scientific Reports* 11.1 (feb. de 2021), pág. 3431. DOI: 10.1038/s41598-021-82993-9.

[161] Néhémie Nzoyikorera et al. «Whole genomic comparative analysis of Streptococcus pneumoniae serotype 1 isolates causing invasive and non-invasive infections among children under 5 years in Casablanca, Morocco». En: *BMC Genomics* 22.1 (dic. de 2021), pág. 39. DOI: 10.1186/s12864-020-07316-0.

Apéndice A

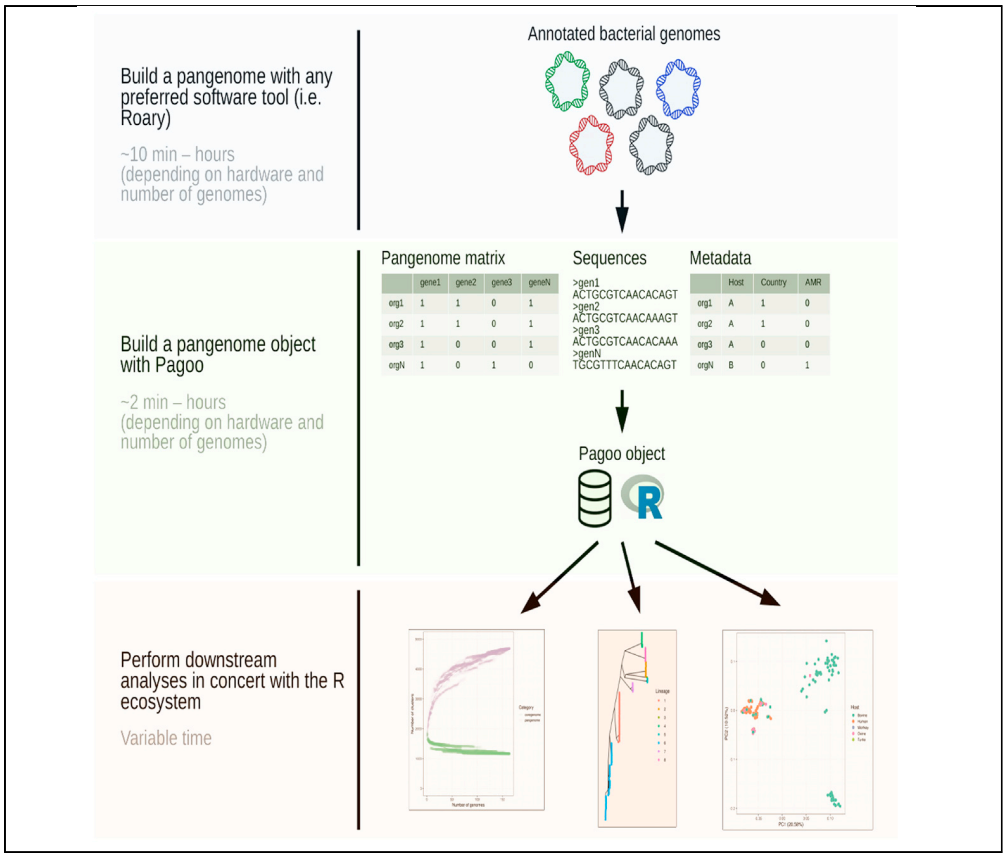
Protocolo para el análisis de pangenomas bacterianos

STAR Protocols

CellPress
OPEN ACCESS

Protocol

Protocol for post-processing of bacterial pangenome data using Pagoo pipeline



Multiple downstream analyses are necessary to interpret the output of bacterial pangenome reconstruction software. This requires integrating diverse kinds of genetic and phenotypic data, which to date are left to each user's criterion. To fill this gap, we created Pagoo, a pangenome post-processing tool that leverages a standardized but flexible and extensible framework for data integration, analysis, and storage. Here, we provide the protocol for running Pagoo and performing from simple to more complex comparative analyses on bacterial pangenome data.

Ignacio Ferrés, Gregorio Iraola

iferres@pasteur.edu.uy (I.F.)
giraola@pasteur.edu.uy (G.I.)

Highlights

Bacterial pangenome analysis needs the integration of genetic and phenotypic data

This protocol shows how to achieve this using the Pagoo software package

Pagoo works in concert with other microbial genomics packages in the R ecosystem

Ferrés & Iraola, STAR Protocols 2, 100802 December 17, 2021
https://doi.org/10.1016/j.xpro.2021.100802

STAR Protocols

CellPress
OPEN ACCESS

Protocol

Protocol for post-processing of bacterial pangenome data using Pagoo pipeline

Ignacio Ferrés[1,2,5,*] and Gregorio Iraola[1,2,3,4,6,*]

[1]Microbial Genomics Laboratory, Institut Pasteur Montevideo, Montevideo, Montevideo 11400, Uruguay

[2]Center for Innovation in Epidemiological Surveillance, Institut Pasteur Montevideo, Montevideo, Montevideo 11400, Uruguay

[3]Wellcome Sanger Institute, Hinxton, Cambridgeshire CB10 SA1, UK

[4]Center for Integrative Biology, Universidad Mayor, Providencia, Santiago de Chile, Chile

[5]Technical contact

[6]Lead contact

*Correspondence: iferres@pasteur.edu.uy (I.F.), giraola@pasteur.edu.uy (G.I.)
https://doi.org/10.1016/j.xpro.2021.100802

SUMMARY

Multiple downstream analyses are necessary to interpret the output of bacterial pangenome reconstruction software. This requires integrating diverse kinds of genetic and phenotypic data, which to date are left to each user's criterion. To fill this gap, we created Pagoo, a pangenome post-processing tool that leverages a standardized but flexible and extensible framework for data integration, analysis, and storage. Here, we provide the protocol for running Pagoo and performing from simple to more complex comparative analyses on bacterial pangenome data.

For complete details on the use and execution of this protocol, please refer to Ferrés and Iraola (2021).

BEFORE YOU BEGIN

Pagoo is designed to take the output generated by pangenome reconstruction software like Roary (Page et al., 2015), Panaroo (Tonkin-Hill et al., 2020), PEPPAN (Zhou et al., 2020) or panX (Ding et al., 2018). This protocol explains how to use built-in functions from Pagoo to automatically load and analyze output files produced by these pangenome reconstruction tools as they are widely used by the community. The output of any of these software tools can be loaded by Pagoo as a set of tables and genetic sequences. Hence, for starting using Pagoo, pangenome reconstruction needs to be performed on a set of genomes using any preferred tool.

Pangenome reconstruction

Timing: 30 min

Here, we provide a full-reproducible example on how to build a pangenome based on 168 previously published *Campylobacter fetus* genomes (Iraola et al., 2017). These genomes have been previously annotated using Prokka (Seemann, 2014) using default parameters (follow the steps described here to perform genome annotation). In this case, we will use Roary (Page et al., 2015) with default parameters to reconstruct the pangenome. It is assumed that Roary is installed and available in the user's path.

1. Download and decompress genomes from inside the R session (Box 1):

Box 1

```
tar_gz <- "cfetus_pangenome.tar.gz"

if ( !file.exists(tar_gz) ) {

      download.file(url = "https://ndownloader.figshare.com/files/26144075",

      destfile = tar_gz)

}

untar(tarfile = tar_gz, exdir = "C_fetus")
```

2. Run Roary to reconstruct the pangenome:
   a. List GFF annotation files and run Roary from inside the R session (Box 2):

Box 2

```
gffs <- list.files(path = "C_fetus",

                   pattern = "[.]gff$",

                   recursive = TRUE,

                   full.names = TRUE)

roary <- paste("roary -f ./C_fetus/roary_output",

               paste(gffs, collapse = " "))

system(roary)
```

Note: Timing of this step depends on the software selected to perform pangenome reconstruction and the number of genomes to be analyzed.

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
GFF3 input files for pangenome reconstruction	Figshare	https://doi.org/10.6084/m9.figshare.13622354.v1
Software and algorithms		
Pagoo	Ferrés and Iraola, 2021	https://github.com/iferres/pagoo
Roary	Page et al., 2015	https://sanger-pathogens.github.io/Roary

MATERIALS AND EQUIPMENT

- Data (output files produced by pangenome reconstruction software - see pangenome reconstruction in before you begin). As a working example, this protocol uses the dataset listed in the key resources table.
- Pagoo works in Linux, Mac and Windows systems in which R is installed. This protocol was conducted on Linux (Fedora Generic release 26) and R v4.0.3 using a desktop computer with 64 GB RAM and an Intel Core i7-7700 (3.60GHz) processor.

STEP-BY-STEP METHOD DETAILS

Step 1: Installing Pagoo

Timing: 4 min

Full installation of Pagoo includes downloading the Pagoo package and resolving all its dependencies from CRAN. Alternatively, the last development version of Pagoo can be installed from GitHub.

1. Install Pagoo from CRAN by running the following code:

```
install.packages("pagoo")
```

2. Alternatively, install Pagoo from GitHub by running the following code:

```
if (!require("devtools"))

devtools::install_github("iferres/pagoo")
```

Step 2: Loading pangenome data from scratch

Timing: 5 min

First, we provide a toy example that describes how to create Pagoo's data structure from scratch independently of the pangenome reconstruction software that was used to generate the data. This toy dataset is included when the user installs Pagoo package.

3. **Generate a Pagoo object.** To generate a Pagoo object, the only mandatory data structure is a data.frame with information relating organisms to genes to orthologous clusters. Other optional data can be also added as additional columns or tables. Load toy example files by running the code in Box 3:

Box 3

```
library(pagoo)

tgz <- system.file("extdata", "toy_data.tar.gz", package = "pagoo")

untar(tarfile = tgz, exdir = ".")

files <- list.files(full.names = TRUE, pattern = "tsv$|fasta$")
```

   a. We will see the case_df.tsv file that is the mandatory data.frame. Run the code in Box 4 to load and inspect it:

Box 4

```
data_file <- grep("case_df.tsv", files, value = TRUE)

data <- read.table(data_file, header = TRUE,

         sep = "\t", quote = "")

head(data)

##       gene        org cluster                              annot

## 1  gene081  organismA   OG001  Thioesterase superfamily protein
```

. Continued

```
## 2  gene122  organismB   OG001          Thioesterase superfamily
## 3  gene299  organismC   OG001  Thioesterase superfamily protein
## 4  gene186  organismD   OG001  Thioesterase superfamily protein
## 5  gene076  organismE   OG001          Thioesterase superfamily
## 6  gene352  organismA   OG002  Inherit from proNOG: Thioesterase
```

b. The first column in this file identifies each gene, the second column identifies the organism to which each gene belongs and the third column shows the orthologous cluster to which each gene was assigned in the pangenome reconstruction. The fourth column shows the annotation of each gene (optional as other additional metadata columns that can be added to this table). With only this data you can start working with Pagoo as follows:

```
p <- pagoo(data = data)
```

4. **Adding metadata to organisms.** Relevant data associated to each organism (i.e., geographic origin, collection date, phenotyping, etc.) can be added to the basic pangenome structure as detailed in Box 5:

Box 5

```
orgs_file  <- grep("case_orgs_meta.tsv", files, value = TRUE)

orgs_meta <- read.table(orgs_file, header = TRUE,

                        sep = "\t", quote = "")

head(orgs_meta)

##          org sero     country
## 1  organismA    a   Westeros
## 2  organismB    b   Westeros
## 3  organismC    c   Westeros
## 4  organismD    a      Essos
## 5  organismE    b      Essos
```

⚠ CRITICAL: Beware that organism names provided in orgs_meta$org must coincide with names provided in the data$org field, in order to correctly map each variable.

Note: If partial metadata is available, for example host data is only available for a subset of organisms, fields with missing data will be automatically filled with NAs.

5. **Adding metadata to orthologous clusters.** Relevant data can be also incorporated to each orthologous cluster, for example its functional annotation as detailed in Box 6 (see here for a guide for generating functional annnotations using the eggNOG database and related tools):

Box 6

```
clust_file  <- grep("case_clusters_meta.tsv", files, value = TRUE)

clust_meta <- read.table(clust_file, header = TRUE,

                         sep = "\t", quote = "")

head(clust_meta)
```

. Continued

```
##    cluster    kegg    cog
## 1    OG001    <NA>      S
## 2    OG002    <NA>      S
## 3    OG003    <NA>   <NA>
## 4    OG004    <NA>      D
## 5    OG005   K01990     V
## 6    OG006    <NA>      V
```

⚠ CRITICAL: Again, the column clust_meta$cluster must contain the same identifiers as the data$cluster column to be able to map one into the other.

6. **Adding sequences.** If the user wants to add sequences, these must be provided for all organisms in the dataset. The type of data needed are multi-FASTA files where each individual sequence represents a gene whose name can be mapped to the data$gene column. Sequences need to be loaded as a list where each element is named with an organism name that maps to data$org and org_meta$org. This can be done as described in Box 7:

Box 7

```
fasta_files <- grep("[.]fasta", files, value = TRUE)

names(fasta_files) <- sub("[.]fasta", "", basename(fasta_files))

library(Biostrings)

sq <- lapply(fasta_files, readDNAStringSet)

# Names are the same as in data$org

## [1] "organismA" "organismB" "organismC" "organismD" "organismE"
```

Note: Pagoo currently supports the incorporation of DNA sequences. However, once sequences are incorporated into the Pagoo object, these can be translated and used as protein sequences for downstream analysis using Biostrings and other R packages.

7. **Generate an object with multiple data.** Now, we can set up a Pagoo object integrating all this data, including presence/absence of each orthologous cluster in each organism, gene annotations, functional annotations, organisms metadata and sequences. To build the Pagoo object run the code in Box 8:

Box 8

```
p <- pagoo(data = data, # Required data

           org_meta = orgs_meta, # Organisms metadata

           cluster_meta = clust_meta, # Clusters metadata

           sequences = sq) # Sequences
```

8. **Adding more metadata after creating the Pagoo object.** The user can add new metadata as the outcome of downstream analysis or experiments. This can be achieved by adding new metadata columns either to each gene, cluster or organism defined in the Pagoo object. By running the following code, we illustrate how to add a new column to the $organisms field named host that describes the host where each organism was isolated from (see Box 9):

Box 9

```
host_df <- data.frame(org = p$organisms$org,
                      host = c("Cow", "Dog", "Cat", "Cow", "Sheep"))
p$add_metadata(map = "org", host_df)
p$organisms
##          org     sero     country     host
## 1  organismA        a   Westeros      Cow
## 2  organismB        b   Westeros      Dog
## 3  organismC        c   Westeros      Cat
## 4  organismD        a      Essos      Cow
## 5  organismE        b      Essos    Sheep
```

⚠ CRITICAL: To allow Pagoo to correctly map the data, values in the first column of the host_df table must be the same as in p$organisms$org, and its column header must also be named org.

Note: Preparing data from scratch and loading classes can be relatively laborious, but in real life working datasets this will be rarely needed. To avoid this, Pagoo provides helper functions that directly parse and read-in output files produced by most widely-used pangenome reconstruction software, avoiding any formatting or manipulation of data (see next step).

Step 3: Input from pangenome reconstruction software

Timing: 10 min

Pagoo allows read-in output files produced by most widely-used pangenome reconstruction software. Since its publication in 2015, Roary (Page et al., 2015) has been the standard and most cited software for pangenome reconstruction. More recently, other related software have emerged like PIRATE (Bayliss et al., 2019), Panaroo (Tonkin-Hill et al., 2020) and PEPPAN (Zhou et al., 2020) that improved different steps of pangenome reconstruction or provided new analytical approaches. Also, we have created our own pangenome reconstruction software called Pewit (unpublished but available at https://github.com/iferres/pewit), which automatically generates a Pagoo-like object to perform downstream analyses. This object contains all the methods and fields that Pagoo provides, but adding a set of methods and fields exclusive to Pewit (not covered in this protocol). Here, we provide full details on how to load output files from Roary and indicate how to load output files from other above-listed software.

9. To create a pangenome object using Pagoo from output files produced by Roary (as described in step 2), run the code described in Box 10 assuming you are placed in the output directory generated by Roary:

Box 10

```
gffs <- list.files(path = "../gffs/", pattern = "[.]gff$", full.names = TRUE)
gpa_csv <- "gene_presence_absence.csv"
library(pagoo)
p <- roary_2_pagoo(gene_presence_absence_csv = gpa_csv, gffs = gffs)
```

Note: Exactly the same approach can be used to load output files from Panaroo, by using the analogous function panaroo_2_pagoo(). Other software like PIRATE and PEPPAN provide scripts (PIRATE_to_roary.pl and PEPPAN_parse.py, respectively) that transform their output files into Roary's output format. Then, roary_2_pagoo() function can be used to generate the pangenome object from PIRATE and PEPPAN once this transformation has been applied.

Step 4: Querying pangenome data

Timing: 10 min

The user can easily explore information that is stored inside the Pagoo object using standard R notation. Indeed, this object has its own associated data and methods that can be easily queried with the '$' operator. These methods allow for the rapid subsetting, extraction and visualization of pangenome data.

10. **Summary statistics.** A pangenome can be stratified in different gene subsets according to their frequency. The core genes are defined as those present in every genome (also the term soft core is typically used for genes occurring in 95%–100% of genomes). The remaining genes are defined as the accessory genome, that can be subdivided in cloud genes or singletons (present in only one genome or only in genomes that) and shell genes which are those in the middle. Run the following code in Box 11 to see this:

Box 11

```
p$summary_stats

#    Category  Number

# 1    Total    7326

# 2    Core    1489

# 3    Shell    5010

# 4    Cloud    818
```

11. **Core level.** The core level defines the minimum number of genomes (as a percentage) in which a certain gene should be present to be considered a core gene. By default, Pagoo considers as core genes all those present in at least 95% of organisms. The core level can be modified to be more or less stringent defining the core genome. Modifying the core level will affect the pangenome object state resulting in different core, shell and cloud sets. See this running the command in Box 12:

Box 12

```
p$core_level       # [1] 95

p$core_level <- 100 # Change value

p$summary_stats    # Updated object

#    Category    Number

# 1     Total      7326

# 2      Core      1117

# 3     Shell      5391

# 4     Cloud       818
```

12. **Pangenome matrix.** The pangenome matrix is one of the most useful things when analyzing pangenomes. Typically, it represents organisms in rows and clusters of orthologous genes in columns informing about gene abundance (considering paralogues). The pangenome matrix looks like the one shown in Box 13 (printing only first 5 rows and columns):

Box 13

```
p$pan_matrix[1:5, 1:5]

#             aadK  aaeA_1  aaeA_2  aat  aat_2

# 16244_6#1      1       0       0    1      0

# 16244_6#10     0       0       0    1      0

# 16244_6#11     1       0       0    1      0

# 16244_6#12     0       0       0    1      0

# 16244_6#13     1       0       0    1      0
```

13. **Genes metadata.** Metadata associated with each individual gene can be accessed by the $genes suffix. It always contains the gene name, the organism to which it belongs, its assigned cluster and a gene identifier (gid). Optionally, it can typically include annotation data, genomic coordinates, etc. Gene metadata is splitted by cluster, so it consists of a list of dataframes (Box 14, showing only the first rows):

Box 14

```
p$genes[1]

# SplitDataFrameList of length 1

# $aadK

# DataFrame with 7 rows and 10 columns

#                    cluster          org               gene                    gid

#                   <factor>    <factor>          <factor>           <character>

# 16244_6#1_00636       aadK   16244_6#1    16244_61_00636    16244_6#1__16244_6..

# 16244_6#11_00101      aadK  16244_6#11  16244_6#11_00101   16244_6#11__16244_6..

# 16244_6#13_00100      aadK  16244_6#13  16244_6#13_00100  16244_6#13__16244_6#..
```

14. **Clusters metadata.** Groups of orthologous genes (clusters) are also stored in Pagoo objects as a table with a cluster identifier per row, and additional columns as optional metadata (Box 15, showing only first rows):

Box 15

```
p$clusters

# DataFrame with 7326 rows and 2 columns

#   cluster              Annotation

#  <factor>              <character>

# 1  aadK        hypothetical protein

# 2  aaeA_1    Ribonuclease P prote..

# 3  aaeA_2    N-carbamoyl-D-amino..
```

. *Continued*

```
# 4  aat       putative ABC transpo..
# 5  aat_2      hypothetical protein
```

15. **Sequences.** Although it is an optional field (it exists only if the user provides this data as an argument when the object is created), $sequences gives access to sequence data. Sequences are stored as a DNAStringSetList object as defined in the Biostrings package. The code in Box 16 will list all sequences in clusters (only showing first rows):

Box 16

```
p$sequences
# DNAStringSetList of length 7326
# [["COQ2"]] 16244_6#1__16244_6#1_01627=ATGGCTAAATTTACTCAAATTTTAAAAGATATAAACGAA...
# [["COQ3_1"]] 16244_6#1__16244_6#1_00352=ATGAGTAACGCAAACGCATGGGACGATATGTCAAATT...
# [["COQ3_2"]] 16244_6#10__16244_6#10_01654=ATGAAAAAAACGTTTTCATTTGGAAAAAACTGGCT...
# [["COQ3_3"]] 16244_6#1__16244_6#1_00772=ATGAAAGAAAAGTTTTTTGAACTAAAAGTTTTAAGCC...
```

Then, you can list all sequences of a single cluster by running the code in Box 17 (only showing first rows):

Box 17

```
p$sequences[["aadK"]]
# DNAStringSet object of length 7:
# [1] 858 ATGAAAATGAGAACAGAGAAACA...AAAAAGAAAAATATCAAAGATAA 16244_6#1__16244_...
# [2] 858 ATGAAAATGAGAACAGAGAAACA...AAAAAGAAAAATATCAAAGATAA 16244_6#11__16244...
# [3] 858 ATGAAAATGAGAACAGAGAAACA...AAAAAGAAAAATATCAAAGATAA 16244_6#13__16244...
# [4] 858 ATGAAAATGAGAACAGAGAAACA...AAAAAGAAAAATATCAAAGATAA 16244_6#6__16244_...
```

⚠ CRITICAL: Note that sequence names are created by pasting organism names and gene names, separated by a string that by default is sep = '__' (two underscores). This is the same as the gid column in the $genes field, and is initially set when a Pagoo object is created. If you think your dataset contains names with this separator, then you should set this parameter to another string to avoid conflicts.

Note: Sequences can be written to text as multi-FASTA format files using standard methods provided by the Biostrings package.

16. **Organisms metadata.** The $organisms field contains a table with organisms and metadata as additional columns if provided (Box 18, only showing first rows):

Box 18

```
p$organisms
# DataFrame with 168 rows and 10 columns
#     org      Accession.Number  Identifier    Strain    Year    Country
#     <factor> <character>  <character>  <character>  <integer>  <character>
# 1  16244_6#1   ERS672242          FR10    2006/367h       2006       France
```

. Continued

```
# 2  16244_6#10   ERS672251        FR19    2008/755h       2008       France
# 3  16244_6#11   ERS672252        FR20    2008/898h       2008       France
# 4  16244_6#12   ERS672253        FR21     2010/41h       2010       France
# 5  16244_6#13   ERS672254        FR22    2010/524h       2010       France
```

Step 5: Data subsetting

Timing: 10 min

Data subsetting is a fundamental operation when working with pangenome, enabling structured and more in depth analyses. Pagoo provides three ways of subsetting: (i) predefined subsets, (ii) classic R's subsetting operations using square bracket operators, and (iii) removal or recovering organisms from the dataset.

17. **Predefined subsets.** As explained in step 10, elements within a pangenome can be classified in different compartments given their frequency of occurrence: core, shell and cloud. Pagoo provides operators to directly access elements in these compartments, independently if they are genes, clusters or sequences. Look at the following table for all possible combinations:

	$core_*	$shell_*	$cloud_*
$*_genes	$core_genes	$shell_genes	$cloud_genes
$*_clusters	$core_clusters	$shell_clusters	$cloud_clusters
$*_sequences	$core_sequences	$shell_sequences	$cloud_sequences

    a. As seen in the above table, the notation is quite straightforward. See example in Box 19 using $clusters to illustrate this better:

Box 19

```
dim(p$clusters)[1]

# [1] 7326

dim(p$core_clusters)[1]

# [1] 1498

dim(p$shell_clusters)[1]

# [1] 5010

dim(p$cloud_clusters)[1]

# [1] 818
```

It can be appreciated that the total number of orthologous clusters in this pangenome is 7326, but only 1498 represent clusters of core genes.

18. **Standard R subsetting notation.** Elements represented as matrices or vectors contained in the Pagoo object can be subsetted using standard R notation using square brackets. Let's see a couple of examples.
    a. Subsetting the pangenome matrix (Box 20):

Box 20

```
p$pan_matrix[1:3, 10:15]

#             accB  accB_1  accC  accC_1  accD  accD_2

# 16244_6#1      1       0     1       0     1       0

# 16244_6#10     1       0     1       0     1       0

# 16244_6#11     1       0     1       0     1       0
```

b. Subsetting core sequences (Box 21):

Box 21

```
p$core_sequences[c(1,30)]

# DNAStringSetList of length 2

# [["COQ2"]] 16244_6#1__16244_6#1_01627=ATGGCTAAATTTACTCAAATTTTAAAAGATATAAACGAA...

# [["ansA"]] 16244_6#1__16244_6#1_00415=ATGTGCTTAAAAAAGGTGTTTATACTTATGCTGATTACG...
```

19. **Dropping and recovering organisms.** The possibility of hiding certain organisms from the dataset is useful if we want to remove some genome with abnormal characteristics (i.e., potentially contaminated), if we want to focus just in a subset of genomes of interest given any metadata value, or if we included an outgroup for phylogenetic purposes but we want to discard it from downstream analyses. The following steps show how this works:.
    a. We are working with 168 organisms in the dataset, out of which 74 are from human origin (see Box 22):

Box 22

```
table(p$organisms$Host)

# Bovine Human Monkey Ovine Turtle

#     78    74      1    13      2
```

b. We will hide these 74 from the dataset (see Box 23):

Box 23

```
to_drop <- which(p$organisms$host=="Human")

p$drop(to_drop)

table(p$organisms$host)

# Bovine  Monkey  Ovine  Turtle

#     78       1     13       2
```

Note: When the user hides a set of organisms, this will have an impact on all the information stored in the object. This means that all features associated with these genomes including genes, sequences, clusters and metadata will be hidden. It is important to note that the user does not have to reassign the object to a new one, it is self-modified (in place modification) according to R6 reference semantics.

c. To recover hidden organisms run the following (see Box 24):

Box 24

```
dropped <- p$dropped p$recover(dropped)
```

Step 6: Built-in methods and visualizations

Timing: 10 min

Pagoo provides basic but fundamental statistical analyses and visualizations for straightforward exploration of pangenome features. All these methods are embedded in the Pagoo object.

20. **Principal Components Analysis (PCA).** PCA is a fundamental statistical tool that can be applied to pangenome data to see how organisms are grouped based on the diversity of their accessory genes. PCA can be calculated directly from the Pagoo object as follows (Figure 1):
    a. Generate a standard PCA object for downstream analysis:

```
pca <- p$pan_pca()
```

    b. Directly visualizing the first 2 PCs using a biplot. This uses the previous method to perform the PCA but allows you to generate a customizable ggplot2 object on the fly (see Box 25):

Box 25

```
p$gg_pca(color = "Host", size = 4) +
    theme_bw(base_size = 15) +
    scale_color_brewer(palette = "Set2")
```

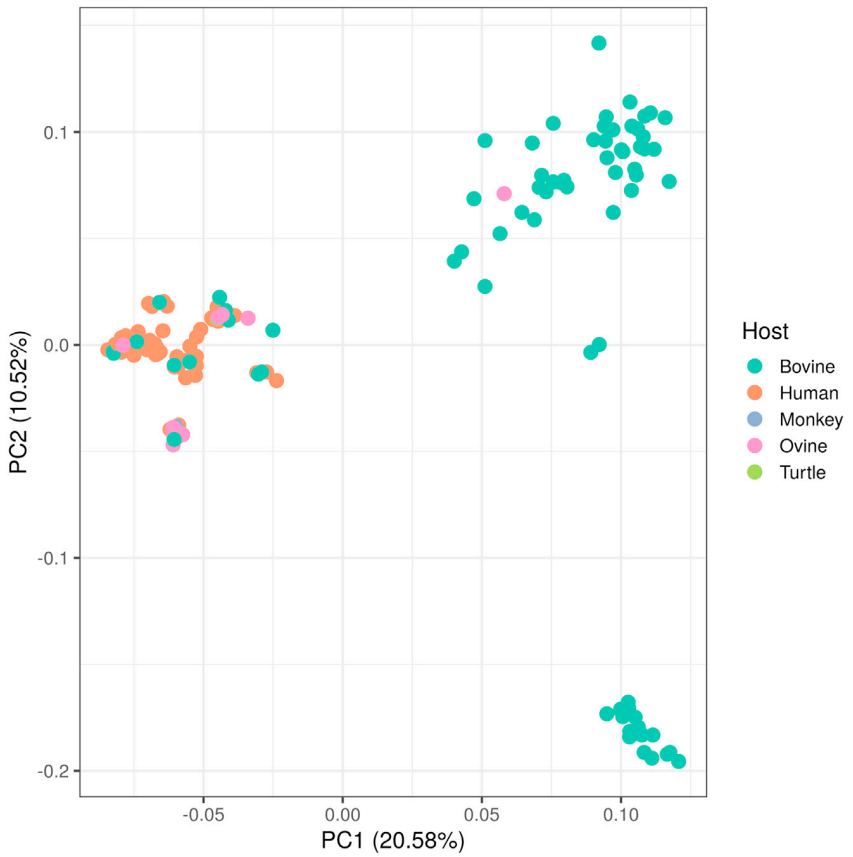


Figure 1. Principal components analysis

A PCA is generated directly from the gene presence/absence matrix and in this case organisms are colored by host of origin.

21. **Rarefaction curves.** Pangenome curves (Figure 2) show the number of gene clusters that are subsequently discovered as more genomes are added to the dataset. If the pangenome is open, more novel accessory genes will be discovered as new genomes are added and the size of the core genome will tend to decrease. Pagoo applies the Power-law distribution to fit the pangenome size and the Exponential decay function to fit the core genome size. Run this method and customize results as shown in Box 26:

Box 26

```
p$gg_curves(size = 2) +
  ggtitle("Pangenome curves") +
  geom_point(alpha = 0.1, size = 4) +
  theme_bw(base_size = 15) + ylim(0, 5000) +
  scale_color_brewer(palette = "Accent")
```

22. **Other methods.** Pagoo provides further methods for summary statistics whose application is analogous to the above described ones. These include pie charts using $gg_pie(), gene presence/absence bin maps using $gg_binmap() and gene frequency bar plots using $gg_barplot().
23. **Dynamic visualization.** The above-mentioned plots for summary statistics can be deployed through a R Shiny application, allowing responsive and dynamic exploration of pangenome data. The application can be run as follows:

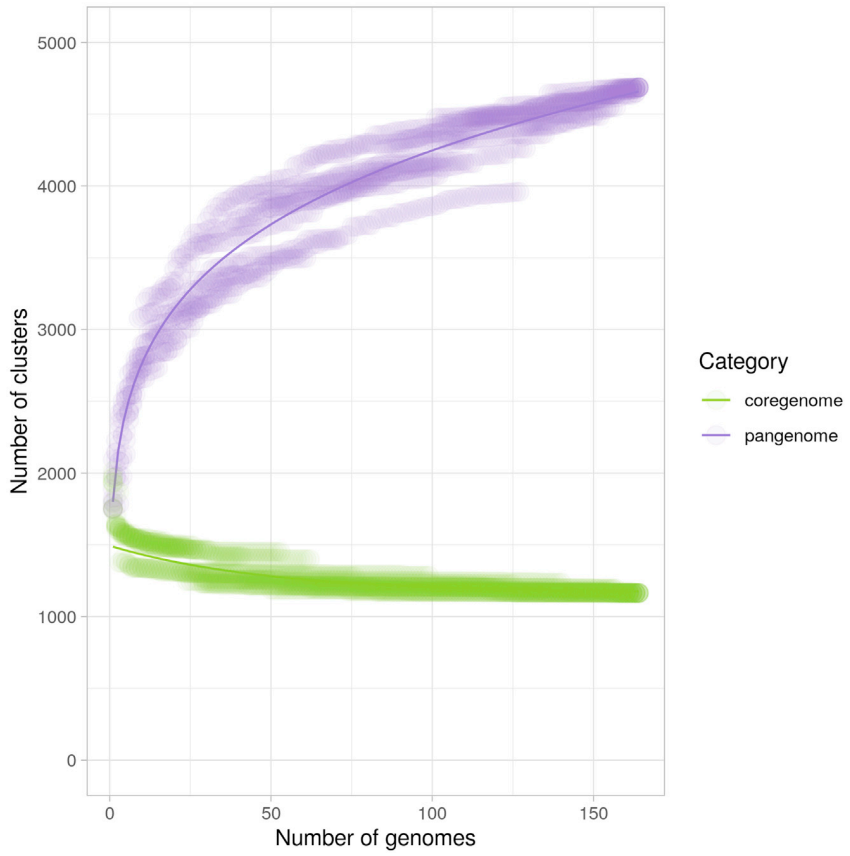


Figure 2. Pangenome curves
Pangenome curves show the accessory and core genome size and are indicative of the gene pool size in a certain dataset.

```
p$runShinyApp()
```

CAUTION: The method described in step 23 (R-Shiny application) is currently not intended for very large datasets, as it may render slow. We recommend it to work with dozens to hundreds of genomes. For bigger pangenomes we recommend the use of the R command line.

Note: An online example for a set of 69 genomes can be found here.

Step 7: Downstream analyses using recipes

Timing: 20 min

Here we show how Pagoo can interact with other R or external tools to generate more complex analyses. We introduce the concept of Pagoo recipes, that are concise pieces of code to complete different tasks. By using these recipes (or creating new ones) the user can take full profit of functionalities provided by Pagoo to perform a variety of analyses including phylogenetics, pangenome-wide association studies, sequence comparisons, ecological measures, preparation of publication-quality figures, among others. In this protocol, we provide a couple of examples of how to create these recipes, which can be found on GitHub.

24. **Publication quality figures.** The following recipe (Box 27) allows to produce a publication quality figure showing pangenome main features using the previously

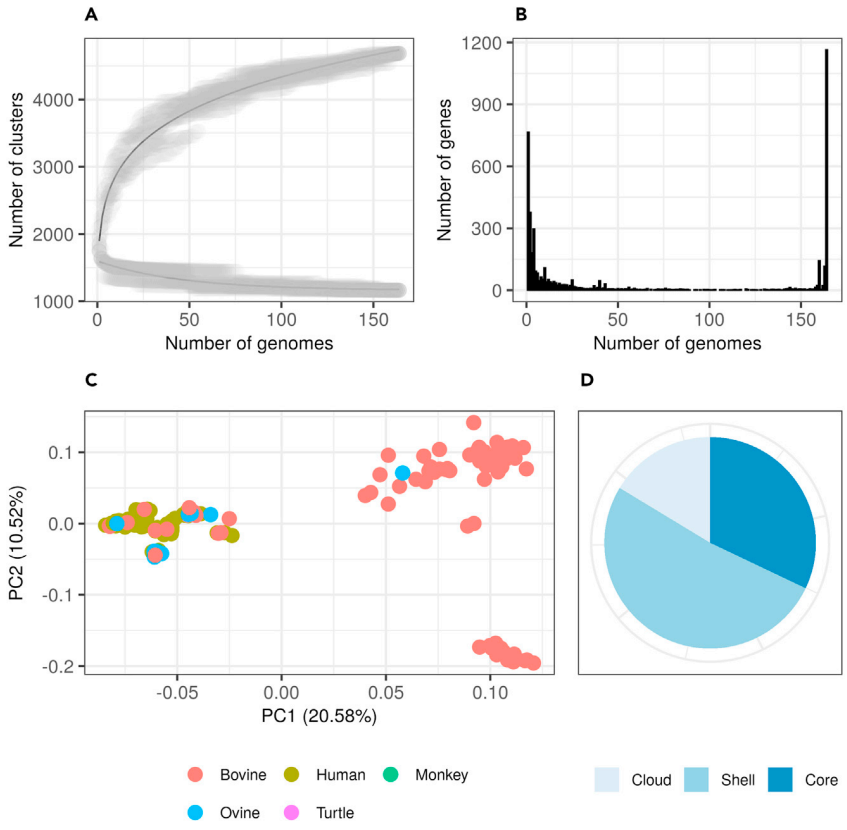


Figure 3. Visualization of pangenome features

Pagoo can be integrated with other R packages to produce publication-quality figures in a simple way. In this case, the figure shows an assembly of different analyses that summarize general features of this example pangenome: (A) pangenome curves, (B) gene frequency plots, (C) Accessory genes PCA and (D) pie chart with gene subsets.

mentioned methods (result shown in Figure 3), directly from the Pagoo object in the R session:

Box 27

```
# 1. Pangenome curves
panel1 <- p$gg_curves() +
    scale_color_manual(values = c("black", "black")) +
    geom_point(alpha = .05, size = 4, color = "grey") +
    theme_bw(base_size = 15) +
    labs(subtitle = "A") +
    theme(legend.position = "none",
      axis.title = element_text(size = 12),
      axis.text = element_text(size = 12))
# 2. Gene frequency bar plots
panel2 <- p$gg_barplot() +
    theme_bw(base_size = 15) +
    labs(subtitle = "B") +
    theme(axis.title = element_text(size = 12),
      axis.text = element_text(size = 12)) +
    geom_bar(stat = "identity", color = "black", fill = "black")
# 3. PCA of accessory genes colored by host
panel3 <- p$gg_pca(color = "Host", size = 4) +
    theme_bw(base_size = 15) +
    labs(subtitle = "C") +
    guides(color = guide_legend(nrow = 2, byrow = T)) +
    theme(legend.position = "bottom",
      legend.title = element_blank(),
      legend.text = element_text(size = 10),
      axis.title = element_text(size = 12),
      axis.text = element_text(size = 12))
# 4. Pie chart of core and accessory genes
panel4 <- p$gg_pie() + theme_bw(base_size = 15) +
    scale_fill_brewer(palette = "Blues") +
    scale_x_discrete(breaks = c(0, 25, 50, 75)) + labs(subtitle = "D") +
    theme(legend.position = "bottom", legend.title = element_blank(),
      legend.text = element_text(size = 10),
      legend.margin = margin(0, 0, 13, 0), legend.box.margin = margin(0, 0, 5, 0),
      axis.title = element_blank(), axis.ticks = element_blank(),
      axis.text.x = element_blank())
```

. *Continued*

```
# 5. Use patchwork to arrange plots using math operators
library(patchwork)
figure <- (panel1 + panel2) / (panel3 + panel4)
```

Note: Pagoo includes a responsive visualization dashboard through a R-Shiny application that allows the user to explore pangenome characteristics and perform standard comparative analyses like those shown in Figure 3. For a pangenome object named p, deploy the Shiny application running p$runShinyApp().

25. **Core genome phylogeny and population structure.** The following recipe (Box 28) shows how to build a phylogenetic tree directly from the Pagoo object by using concatenated alignments of each individual core gene. This is performed by interacting with diverse R packages like Biostrings and DECIPHER (Wright, 2015) for sequence handling and alignment, phangorn (Schliep, 2011) for phylogenetic reconstruction, and ggtree (Yu et al., 2017) for tree visualization.

Box 28

```
# Load required packages
library(magrittr)
library(DECIPHER)
library(Biostrings)
library(phangorn)
library(ggtree)
library(rhierbaps)
# Drop Cft and set core level to 100
cft <- p$organisms[which(p$organisms$Subspecies=="Cft"),"org"]
p$drop(cft)
p$core_level <- 100
# Align individual core genes
ali <- p$core_seqs_4_phylo() %>%
  lapply(DECIPHER::AlignTranslation)
# Identify neutral core clusters using Tajima's D
tajD <- ali %>%
    lapply(ape::as.DNAbin) %>%
    lapply(pegas::tajima.test) %>%
    sapply("[[", "D")
neutral <- which(tajD <= 2 & tajD >= -2)
# Concatenate neutral core gene clusters
concat_neu <- ali[neutral] %>%
     do.call(Biostrings::xscat, .) %>%
     setNames(p$organisms$org) %>%
```

. Continued

```
    as("matrix") %>%
    tolower()
# Find population structure with RhierBAPS
rhb <- hierBAPS(snp.matrix = concat_neu, n.pops = 10,
      max.depth = 1, n.extra.rounds = 5)
# Add lineage information to organisms metadata
res <- rhb$partition.df
lin <- data.frame(org = as.character(res[, 1]),
        lineage = as.factor(res[, 2]))
p$add_metadata(map = "org", data = lin)
# Compute phylogeny
tre <- concat_neu %>%
  phangorn::phyDat(type = "DNA") %>%
  phangorn::dist.ml() %>%
  phangorn::NJ()
# Draw phylogeny with lineage and host information
gg1 <- ggtree(tre, ladderize = T, layout = "slanted") %<+%
  as.data.frame(p$organisms) +
  geom_tippoint(aes(color = as.factor(lineage))) +
  labs(subtitle = "A") +
  scale_color_discrete("Lineage")
gg2 <- ggtree(tre, ladderize = T, layout = "slanted") %<+%
  as.data.frame(p$organisms) +
  geom_tippoint(aes(colour = as.factor(Host))) +
  labs(subtitle = "B") +
  scale_colour_discrete("Host")
fig2 <- gg1 + gg2
```

Results presented in Figure 4 exemplify how a relatively complex task that needs of many steps like extracting core genes, keeping those that show signal of neutral evolution, aligning and concatenating them, determining population structure to finally perform a phylogenetic reconstruction, can be achieved directly from the Pagoo pangenome object with different sequential code recipes using different microbial genomics packages within the R environment.

Step 8: Saving and loading pangenome data

Timing: 5 min

Once the pangenome object is created, Pagoo provides two methods for saving and reloading it to a new R session:

26. Saving as plain text files (Box 29):

Box 29

```
outdir <- paste(".", "my_pangenome", sep = "/")

p$write_pangenome(dir = outdir)

list.files(outdir, full.names = TRUE)

## [1] "./my_pangenome/clusters.tsv"

## [2] "./my_pangenome/data.tsv"

## [3] "./my_pangenome/organisms.tsv"
```

This will create a directory with 3 text files. The advantage of this approach is that you can analyze it outside R, the disadvantage is that full reproducibility can be compromised since reading text could be less stable (classes or number precision can be lost).

27. Saving as R data format (Box 30):

Box 30

```
rds <- paste("./my_pangenome", "pangenome.RDS", sep = "/")

p$save_pangenomeRDS(file = rds)

p2 <- load_pangenomeRDS(rds)
```

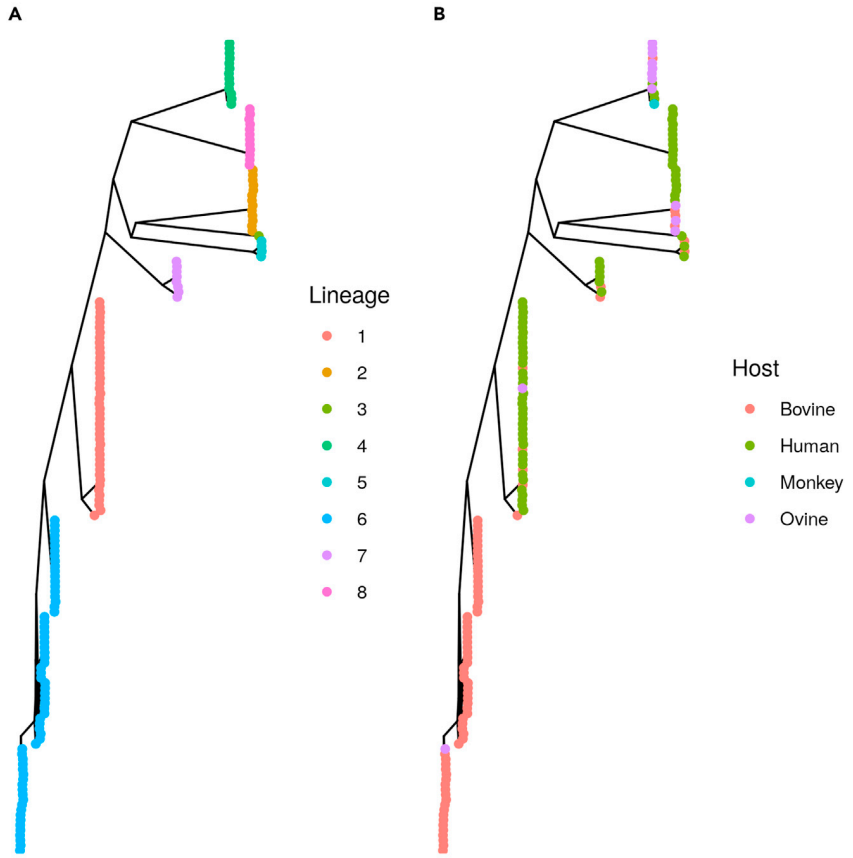


Figure 4. Core genome phylogenies

This shows the output of the above-described recipe aiming to generate a core genome phylogeny directly from the pangenome object. Panel (A) shows the three colored by lineage defined in the same recipe through a population structure analysis and panel (B) shows the tree colored by host.

This solution preserves data structures, such as column data types (numeric, character or factor). So, this method is more stable (compatible between Pagoo versions), secure (uses the same metadata classes), and convenient (the exact state of the object can be saved and restored, keeping all modifications performed to the object during the analysis). Importantly, this option allows to store all pangenome data in a single and easily shareable file.

Note: Authors recommend using the R data format for saving and loading Pagoo objects.

EXPECTED OUTCOMES

Pangenome analysis is a key approach to explore genetic diversity occurring in bacterial populations. Despite the availability of many different software tools for pangenome reconstruction, there are much less alternatives to perform downstream pangenome data analysis in a simple and standardized way. Pagoo is a flexible and extensible tool that systematize pagenome data in a single programming environment, allowing dynamic data analysis and integration with widely used packages for comparative genomics available in the R ecosystem. This protocol aims to be a guide for the use of Pagoo, providing the user with the fundamental skills to work with pangenome data within the R environment. Beyond the user will be able to perform standard phylogenetic analyses, genotype-phenotype association analyses or functional enrichment analyses, Pagoo is an open and flexible framework that allows the development of new modules or code recipes to fulfill specific tasks. Additionally, Pagoo facilitates data storage and sharing, since all data can be saved in a single file that can be recovered later in an independent R session or written to plain text.

LIMITATIONS

Despite the R ecosystem being vast and currently including dozens of packages that allow performing diverse comparative genomic analyses, it is possible that the user finds some specific tasks that are difficult to implement in R or are better covered by other pangenome analysis tools. Pagoo has been tested with hundreds to few thousands of genomes. A limiting aspect here is the time it can take to generate the Pagoo object from bigger datasets. This is particularly relevant for dynamic visualizations that Pagoo provides through its Shiny application.

TROUBLESHOOTING

Problem 1

Copying and modifying the copy of a pangenome object also modifies the original one.

Potential solution

R6 classes are environments which use reference semantics. That means that these kinds of objects need a special method to get copied and if they are simply assigned to a new variable, this variable will point to the original one, and will not get duplicated. To avoid this, use the $clone() method to generate an independent copy of the object.

Problem 2

An error occurs when trying to load a pangenome.

Potential solution

Check that key columns ('gene', 'cluster', and 'org') contain the same elements between the different tables (gene, cluster and organisms). Pagoo checks that these match each other and raises an error if there are missing or different key values. See step 2, Boxes 2, 3, 4, 5, 6, 7, 8, and 9.

Problem 3

I dropped some organisms but I don't remember which ones and also want to recover them.

Potential solution

Use the $dropped field to look if there are some hidden organisms, and then use the $recover() method to recover them. For example, if all dropped organisms want to be recovered from a pangenome object p, then use p$recover(p$dropped). See step 5, section 19.

Problem 4

The state of the pangenome object has changed after saving it and loading in a different R session.

Potential solution

Use save_pangenomeRDS() and load_pangenomeRDS() for saving and loading, respectively. This will load the pangenome object exactly in the same state as it was saved. See step 8, Box 30.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources and reagents should be directed to and will be fulfilled by the lead contact, Gregorio Iraola (giraola@pasteur.edu.uy).

Materials availability

This study did not generate new unique reagents.

Data and code availability

The published article includes all datasets/code generated or analyzed during this study.

ACKNOWLEDGMENTS

This work has been partially funded by Banco de Seguros del Estado (BSE) from Uruguay and Fondo de Convergencia Estructural del Mercosur (FOCEM) grant COF 03/11. I.F. is funded by grant ANII-POS_NAC_2018_1_151494 from Agencia Nacional de Investigación e Innovación (ANII), Uruguay. We thank Pablo Fresia, Andrés Parada, and Daniela Costa for insightful comments and suggestions during testing of Pagoo.

AUTHOR CONTRIBUTIONS

I.F. and G.I. conceived the idea. I.F. developed software and generated the analytical protocols described here. I.F. and G.I. wrote the manuscript.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

Bayliss, S.C., Thorpe, H.A., Coyle, N.M., Sheppard, S.K., and Feil, E.J. (2019). PIRATE: A fast and scalable pangenomics toolbox for clustering diverged orthologues in bacteria. GigaScience *8*, giz119.

Ding, W., Baumdicker, F., and Neher, R.A. (2018). panX: pan-genome analysis and exploration. Nucleic Acids Res. *46*, e5.

Ferrés, I., and Iraola, G. (2021). An object-oriented framework for evolutionary pangenome analysis. Cell Reports Methods *1*, S2667-2375(21)00140-5.

Iraola, G., Forster, S.C., Kumar, N., Lehours, P., Bekal, S., García-Peña, F.J., Paolicchi, F., Morsella, C., Hotzel, H., Hsueh, P.-R., et al. (2017). Distinct Campylobacter fetus lineages adapted as livestock pathogens and human pathobionts in the intestinal microbiota. Nat. Commun. *8*, 1367.

Page, A.J., Cummins, C.A., Hunt, M., Wong, V.K., Reuter, S., Holden, M.T.G., Fookes, M., Falush, D., Keane, J.A., and Parkhill, J. (2015). Roary: rapid large-scale prokaryote pan genome analysis. Bioinformatics *31*, 3691–3693.

Schliep, K.P. (2011). phangorn: phylogenetic analysis in R. Bioinformatics *27*, 592–593.

Seemann, T. (2014). Prokka: rapid prokaryotic genome annotation. Bioinformatics *30*, 2068–2069.

Tonkin-Hill, G., MacAlasdair, N., Ruis, C., Weimann, A., Horesh, G., Lees, J.A., Gladstone, R.A., Lo, S., Beaudoin, C., Floto, R.A., et al. (2020). Producing polished prokaryotic pangenomes with the Panaroo pipeline. Genome Biol. *21*, 180.

Wright, E.S. (2015). DECIPHER: harnessing local sequence context to improve protein multiple sequence alignment. BMC Bioinformatics *16*, 322.

Yu, G., Smith, D.K., Zhu, H., Guan, Y., and Lam, T.T.-Y. (2017). ggtree: an r package for visualization and annotation of phylogenetic trees with their covariates and other associated data. Methods Ecol. Evol. *8*, 28–36.

Zhou, Z., Charlesworth, J., and Achtman, M. (2020). Accurate reconstruction of bacterial pan- and core genomes with PEPPAN. Genome Res. *30*, 1667–1679.