# # Árboles filogenéticos

{#phylogenetic-trees-1}

## Resumen

Los **árboles filogenéticos** se utilizan para visualizar y describir el parentesco y la evolución de los organismos a partir de la secuencia de su código genético.

Pueden construirse a partir de secuencias genéticas utilizando métodos basados en la distancia (como el método de unión de vecinos) o métodos basados en los caracteres (como el método de máxima verosimilitud y el método Bayesiano Markov Chain Monte Carlo). La secuenciación de nueva generación (NGS) se ha vuelto más asequible y se está utilizando cada vez más en la sanidad pública para describir los patógenos causantes de enfermedades infecciosas. Los dispositivos de secuenciación portátiles reducen el tiempo de respuesta y prometen hacer que los datos estén disponibles para apoyar la investigación de brotes en tiempo real. Los datos de NGS pueden utilizarse para identificar el origen o la fuente de una cepa de un brote y su propagación, así como para determinar la presencia de genes de resistencia a los antimicrobianos. Para visualizar el parentesco genético entre las muestras se construye un árbol filogenético.

En esta página aprenderemos a utilizar el paquete **ggtree**, que permite la visualización combinada de árboles filogenéticos con datos de muestra adicionales en forma de dataframe. Esto nos permitirá observar patrones y mejorar la comprensión de la dinámica de los brotes.

## Preparación

### Cargar paquetes

Este trozo de código muestra la carga de los paquetes necesarios. En este manual destacamos p\_load() de **pacman**, que instala el paquete si es necesario y lo carga para su uso. También puede cargar los paquetes instalados con library() de **.** Consulta la página sobre [los fundamentos de R](#r-basics) para obtener más información sobre los paquetes de R.

### Importar datos

Los datos de esta página pueden descargarse con las instrucciones de la página de [descarga de manuales y datos](#download-handbook-and-data).

Hay varios formatos diferentes en los que se puede almacenar un árbol filogenético (por ejemplo, Newick, NEXUS, Phylip). Uno de los más comunes es el formato de archivo Newick (.nwk), que es el estándar para representar árboles en forma legible por ordenador. Esto significa que un árbol completo puede expresarse en un formato de cadena como "((t2:0,04,t1:0,34):0,89,(t5:0,37,(t4:0,03,t3:0,67):0,9):0,59);", enumerando todos los nodos y puntas y su relación (longitud de rama) entre sí.

Nota: Es importante entender que el archivo del árbol filogenético en sí mismo no contiene datos de secuenciación, sino que es simplemente el resultado de las distancias genéticas entre las secuencias. Por lo tanto, no podemos extraer datos de secuenciación de un archivo de árbol.

En primer lugar, utilizamos la función read.tree() del paquete **ape** para importar un archivo de árbol filogenético de Newick en formato .txt, y lo almacenamos en un objeto tipo lista "phylo". Si es necesario, utiliza la función here() del paquete **here** para especificar la ruta relativa del archivo.

Nota: En este caso el árbol newick se guarda como un archivo .txt para facilitar su manejo y descarga desde Github.

tree <- ape::read.tree("Shigella\_tree.txt")

Inspeccionamos nuestro objeto árbol y vemos que contiene 299 puntas (o muestras) y 236 nodos.

En segundo lugar, importamos una tabla almacenada en un archivo .csv con información adicional para cada muestra secuenciada, como el sexo, el país de origen y los atributos de resistencia a los antimicrobianos, utilizando la función import() del paquete **rio**:

sample\_data <- import("sample\_data\_Shigella\_tree.csv")

A continuación se muestran las primeras 50 filas de los datos:

### Limpiar e inspeccionar

Limpiamos e inspeccionamos nuestros datos: Para asignar los datos de muestra correctos al árbol filogenético, los valores de la columna Sample\_ID en el dataframe sample\_data deben coincidir con los valores de tip.labels en el archivo del árbol:

Comprobamos el formato de los tip.labels en el archivo de árbol mirando las 6 primeras entradas usando con head() de .

También nos aseguramos de que la primera columna de nuestro dataframe sample\_data sea Sample\_ID. Miramos los nombres de las columnas de nuestro dataframe utilizando colnames() de .

Miramos los Sample\_IDs en el dataframe para asegurarnos de que el formato es el mismo que en el tip.label (por ejemplo, las letras son todas mayúsculas, no hay barras bajas adicionales \_ entre las letras y los números, etc.)

También podemos comparar si todas las muestras están presentes en el archivo de árbol y viceversa, generando un vector lógico de TRUE o FALSE donde coinciden o no. Estos no se imprimen aquí, para simplificar.

Podemos utilizar estos vectores para mostrar cualquier ID de muestra que no esté en el árbol (no hay ninguna).

Al inspeccionar podemos ver que el formato de Sample\_ID en el dataframe corresponde al formato de los nombres de las muestras en el tip.labels. No es necesario que estén clasificados en el mismo orden para que coincidan.

Estamos listos para ir!

## Visualización simple de un árbol

### Diferentes disposiciones de los árboles

**ggtree** ofrece muchos formatos de presentación diferentes y algunos pueden ser más adecuados para su propósito específico que otros. A continuación se muestran algunas demostraciones. Para otras opciones, consulta este [libro en línea](http://yulab-smu.top/treedata-book/chapter4.html).

A continuación, algunos ejemplos de disposición de árboles:

### Árbol simple con datos de muestra

El operador **%<+%** se utiliza para conectar el dataframe sample\_data con el archivo del árbol. La anotación más sencilla de su árbol es la adición de los nombres de las muestras en las puntas, así como la coloración de los puntos de las puntas y, si se desea, de las ramas:

Este es un ejemplo de árbol circular:

Puedes exportar el gráfico de árbol con ggsave() como lo harías con cualquier otro objeto ggplot. Escrito de esta manera, ggsave() guarda la última imagen producida en la ruta de archivo que especifiques. Recuerde que puede utilizar here() y rutas de archivo relativas para guardar fácilmente en subcarpetas, etc.

## Manipulación de árboles

A veces puede tener un árbol filogenético muy grande y sólo le interesa una parte del árbol. Por ejemplo, si ha producido un árbol que incluye muestras históricas o internacionales para obtener una gran visión general de dónde puede encajar su conjunto de datos en el panorama general. Pero luego, para ver más de cerca sus datos, quiere inspeccionar sólo esa parte del árbol más grande.

Dado que el archivo del árbol filogenético es sólo el resultado del análisis de los datos de secuenciación, no podemos manipular el orden de los nodos y las ramas en el propio archivo. Estos ya han sido determinados en análisis anteriores a partir de los datos NGS en bruto. Sin embargo, podemos ampliar partes, ocultar partes e incluso subconjuntar partes del árbol.

### Acercarse

Si no quieres "cortar" tu árbol, sino sólo inspeccionar una parte de él más de cerca, puedes acercarte para ver una parte específica.

En primer lugar, trazamos todo el árbol en formato lineal y añadimos etiquetas numéricas a cada nodo del árbol.

Para acercarse a una rama en particular (que sobresale a la derecha), utiliza viewClade() en el objeto ggtree p y proporcione el número de nodo para verlo más de cerca:

### Ramas colapsadas

Sin embargo, podemos querer ignorar esta rama y podemos colapsarla en ese mismo nodo (nodo 452) utilizando collapse(). Este árbol se define como p\_collapsed.

Para mayor claridad, cuando imprimimos p\_collapsed, añadimos un geom\_point2() (un diamante azul) en el nodo de la rama colapsada.

### Subconjunto de un árbol

Si queremos hacer un cambio más permanente y crear un nuevo árbol reducido con el que trabajar, podemos subconjuntar parte de él con tree\_subset(). Luego se puede guardar como un nuevo archivo de árbol o archivo .txt.

En primer lugar, inspeccionamos los nodos del árbol y las etiquetas de las puntas para decidir qué se va a subconjuntar.

Ahora, digamos que hemos decidido hacer un subconjunto del árbol en el nodo 528 (mantener sólo las puntas dentro de esta rama después del nodo 528) y lo guardamos como un nuevo objeto sub\_tree1:

Veamos el subconjunto del árbol 1:

También puede hacer un subconjunto basado en una muestra en particular, especificando cuántos nodos "hacia atrás" quiere incluir. Vamos a subconjuntar la misma parte del árbol basándonos en una muestra, en este caso S17BD07692, retrocediendo 9 nodos y lo guardamos como un nuevo objeto sub\_tree2:

Veamos el subconjunto del árbol 2:

También puede guardar su nuevo árbol como un tipo Newick o incluso un archivo de texto utilizando la función write.tree() del paquete **ape**:

### Rotación de nodos en un árbol

Como ya hemos dicho, no podemos cambiar el orden de las puntas o de los nodos en el árbol, ya que éste se basa en su parentesco genético y no está sujeto a manipulación visual. Pero podemos rotar las ramas alrededor de los nodos si eso facilita nuestra visualización.

En primer lugar, trazamos nuestro nuevo subconjunto de árbol 2 con las etiquetas de los nodos para elegir el nodo que queremos manipular y lo almacenamos en un objeto ggtree plot p.

Luego podemos manipular los nodos aplicando ggtree**::rotate()** o **ggtree::flip()**: Nota: para ilustrar qué nodos estamos manipulando aplicamos primero la función **geom\_hilight()** de **ggtree** para resaltar las muestras de los nodos que nos interesan y almacenamos ese objeto de trazado de ggtree en un nuevo objeto p1.

Ahora podemos rotar el nodo 37 en el objeto p1 para que las muestras del nodo 38 se muevan hacia arriba. Almacenamos el árbol rotado en un nuevo objeto p2.

O podemos usar el comando flip para rotar el nodo 36 en el objeto p1 y cambiar el nodo 37 a la parte superior y el nodo 39 a la parte inferior. Almacenamos el árbol volteado en un nuevo objeto p3.

### Ejemplo de subárbol con anotación de datos de muestra

Digamos que estamos investigando el grupo de casos con expansión clonal que se produjo en 2017 y 2018 en el nodo 39 de nuestro subárbol. Añadimos el año de aislamiento de la cepa, así como el historial de viajes y el color por país para ver el origen de otras cepas estrechamente relacionadas:

Nuestra observación apunta a un evento de importación de cepas procedentes de Asia, que luego circularon en Bélgica a lo largo de los años y parecen haber causado nuestro último brote.

## Árboles más complejos: añadir mapas térmicos de datos de muestra

Podemos añadir información más compleja, como la presencia categórica de genes de resistencia a los antimicrobianos y valores numéricos para la resistencia realmente medida a los antimicrobianos en forma de mapa de calor utilizando la función **ggtree::gheatmap()**.

Primero necesitamos trazar nuestro árbol (puede ser lineal o circular) y almacenarlo en un nuevo objeto ggtree plot p: Utilizaremos el subárbol de la parte 3).

En segundo lugar, preparamos nuestros datos. Para visualizar las diferentes variables con nuevos esquemas de color, subestablecimos nuestro dataframe a la variable deseada. Es importante añadir el Sample\_ID como rownames de lo contrario no puede coincidir los datos con el árbol tip.labels:

En nuestro ejemplo, queremos observar el género y las mutaciones que podrían conferir resistencia a la ciprofloxacina, un importante antibiótico de primera línea utilizado para tratar las infecciones por Shigella.

Creamos un dataframe para el género:

Creamos un dataframe para las mutaciones en el gen gyrA, que confieren resistencia a la ciprofloxacina:

Creamos un dataframe para la concentración inhibitoria mínima (CIM) medida para la Ciprofloxacina del laboratorio:

Creamos un primer gráfico añadiendo un mapa de calor binario para el género al árbol filogenético y almacenándolo en un nuevo objeto de gráfico ggtree h1:

A continuación, añadimos información sobre las mutaciones en el gen gyrA, que confieren resistencia a la ciprofloxacina:

Nota: La presencia de mutaciones puntuales cromosómicas en los datos de WGS se determinó previamente utilizando la herramienta PointFinder desarrollada por Zankari et al. (véase la referencia en la sección de referencias adicionales)

En primer lugar, asignamos un nuevo esquema de colores a nuestro objeto de trazado existente h1 y lo almacenamos en un objeto ahora h2. Esto nos permite definir y cambiar los colores para nuestra segunda variable en el mapa de calor.

A continuación, añadimos la segunda capa del mapa de calor a h2 y almacenamos los gráficos combinados en un nuevo objeto h3:

Repetimos el proceso anterior, añadiendo primero una nueva capa de escala de colores a nuestro objeto existente h3, y luego añadiendo los datos continuos sobre la concentración inhibitoria mínima (CIM) de Ciprofloxacina para cada cepa al objeto resultante h4 para producir el objeto final h5:

Podemos hacer el mismo ejercicio para un árbol lineal:

Primero añadimos el género:

A continuación, añadimos las mutaciones de resistencia a la ciprofloxacina después de añadir otra capa de colores:

A continuación, añadimos la concentración mínima inhibitoria determinada por el laboratorio (MIC):

## Recursos

[http://hydrodictyon.eeb.uconn.edu/eebedia/index.php/Ggtree#](http://hydrodictyon.eeb.uconn.edu/eebedia/index.php/Ggtree)  Clade\_Colors <https://bioconductor.riken.jp/packages/3.2/bioc/vignettes/ggtree/inst/doc/treeManipulation.html><https://guangchuangyu.github.io/ggtree-book/chapter-ggtree.html><https://bioconductor.riken.jp/packages/3.8/bioc/vignettes/ggtree/inst/doc/treeManipulation.html>

Ea Zankari, Rosa Allesøe, Katrine G Joensen, Lina M Cavaco, Ole Lund, Frank M Aarestrup, PointFinder: una novedosa herramienta web para la detección basada en WGS de la resistencia a los antimicrobianos asociada a mutaciones puntuales cromosómicas en patógenos bacterianos, Journal of Antimicrobial Chemotherapy, Volume 72, Issue 10, October 2017, Pages 2764-2768, <https://doi.org/10.1093/jac/dkx217>