Introducción a Linux para Bioinformática

LandaLab

2025-09-08

Contents

Chapter 1

Bienvenid@s

Este repositorio contiene el material del taller "Introducción a Linux", un espacio abierto para todas las personas que quieran aprender a analizar y visualizar datos con Linux
No importa tu edad, formación o experiencia previa: si tienes curiosidad y ganas de aprender, este taller es para ti.

1.1 Sobre el taller

- **Lenguaje**: Linux
- **Nivel**: Principiante

- **Modalidad**: Práctico, con ejercicios guiados y codificación en vivo

- **Dirigido a**: Cualquier persona interesada en aprender Linux para Bioinformática

- **Requisitos**: Ninguno. No necesitas saber programación ni instalar ningún programa. Sólo necesitas una computadora con acceso a internet

---

1.2 ¿Qué aprenderás?

Al finalizar el taller, podrás:

- Comunicarte con Linux a través de Shell
- Manipular archivos

- Crear proyectos
- Automatizar tareas
- Escribir scripts reproducibles

- Comprender los principios básicos del análisis de datos en Linux
- Descargar secuencias genómicas
- Instalar programas

1.3 Plataforma

Trabajaremos en una plataforma en línea, por lo que **no necesitas instalar nada en tu computadora**. El acceso será gratuito y se proporcionará durante el taller.

1.4 Agradecimientos

Este taller es parte de un esfuerzo por compartir herramientas abiertas, accesibles y colaborativas. Queremos que más personas se acerquen al mundo de los datos y la ciencia sin barreras. ¡Gracias por formar parte!

1.5 ¿Dudas o comentarios?

Puedes abrir un Issue o escribirnos durante el taller.
¡Estamos aquí para aprender junt@s!

Chapter 2

Introducción a Linux

Chapter 3

Navegar en el sistema de ficheros

todo lo de este archivo hasta atajos en el teclado

https://github.com/DianaOaxaca/Introduccion_linux_para_bioinformatica/blob/main/Comandos_utiles.md

Chapter 4

Manejo de archivos

4.1 Código reproducible

"Los sucesos únicos no reproducibles, no tienen importancia para la ciencia" -Karl Popper, the Logic of Scientific, 1959.

¿Qué se necesita para reproducir el resultado de un análisis computacional?

- **Los datos:**
  - La fuente de donde se descargaron.
  - La versión de la fuente asociada a ellos.
- **Los programas utilizados para analizarlos:**
  - La versión de cada uno de los programas.
  - Los valores de cada uno de los argumentos utilizados.

Argumentos a tener en cuenta para tener buenas prácticas:

- Actualización de bases de datos.
- Siempre existirán excepciones que no cumplen con las suposiciones de tu código.
- **Qué un programa genere un resultado no significa que el resultado sea correcto.**
- Todo lo que se te puede olvidar, **¡se te va a olvidar!**
  - Fuente
  - Suposiciones iniciales
  - ¿Qué genera esa función compleja y rebuscada que parecía una joya en su momento?

SIEMPRE, documenta tu código

Comienza por la organización: La estructura de directorios debe estar organizada, lo mejor es tener un directorio de trabajo por proyecto, los pasos del proyecto se organizarán en subdirectorios.

La estructura de directorios propuesta para estas asesorías es:

`Linux` Este es el directorio principal del proyecto.

`data` En este directorio van los datos de entrada para el proyecto.

`src` En este directorio van los scripts ya probados y funcionales.

`results` En este directorio van los resultados generados.

4.2 Bases de datos y secuencias biológicas

Base de datos: Es una colección organizada de información estructurada, o datos, normalmente almacenados electrónicamente en un sistema informático.

En bioinformática se utilizan diversas bases de datos, algunos ejemplos son:

De diversos organismos:

NCBI: https://www.ncbi.nlm.nih.gov/

Genomas de referencia NCBI: https://ftp.ncbi.nlm.nih.gov/genomes/refseq/

ENSEMBL: https://www.ensembl.org/index.html

UCSC Table Browser: https://genome.ucsc.edu/cgi-bin/hgTables

Dedicadas a organismos específicos:

Ecocyc

Flybase

Wormbase

Especializadas en un tema particular:

ENCODE: Elementos funcionales del genoma humano

RegulonDB: Regulación transcripcional de E. coli

Pfam: Familias proteícas

miRBase: Secuencias de miRNA y sus blancos

Secuencias biológicas: Archivo que contiene la secuencia de genes, genomas y/o proteínas.

Fasta: Se compone de un identificador de la secuencia, seguido (por salto de línea) de la secuencia de nucleótidos o aminoácidos de un gene, genoma o proteína.

Fastq: Normalmente se conmpone de cuatro lineas por secuencia

`Line 1` Comienza con '@' seguido del identificador de la secuencia y una descripción opcional.

`Line 2` La secuencia cruda nucleótidos.

`Line 3` Comienza con un '+' opcionalmente incluye el identificador de la secuencia.

`Line 4` Indica los valores de calidad de la secuencia, debe contener el mismo número de símbolos que el número de nucleótidos.

De anotación: GeneBank o tabulares GFF, GTF, GFF3

Tabulares: Una línea por cada elemento. Cada línea DEBE contener 9 campos. Los campos DEBEN estar separados por tabuladores. Todos los campos DEBEN contener un valor, los campos vacíos se denotan con '.'

`seqname` Nombre del cromosoma

`source` Nombre del programa que generó ese elemento

`feature` Tipo de elemento

`start` Posición de inicio

`end` Posición de final

`score` Un valor de punto flotante

`strand` La cadena (+ , -)

`frame` Marco de lectura

`attribute` Pares tag-value, separados por coma, que proveen información adicional

```
>ID_secuencia,metadatos de identificación
ATGCCCGGTAAAGGATCCCCCCTATGCCGTATAGC
>ID_secuencia,metadatos de identificación
MIPEKRIIRRIQSGGCAIHCQDCSISQLCIPFTLNEHELDQLDNI
```

```
@SEQ_ID
GATTTGGGGTTCAAAGCAGTATCGATCAAATAGTAAATCCATTTGTTCAACTCACAGTTT
+
!''*((((***+))%%%++)(%%%%).1***-+*''))**55CCF>>>>>>CCCCCCC65
```

Agunos foros para pedir ayuda:

Stackoverflow: https://stackoverflow.com/

Biostars: https://www.biostars.org/

Researchgate: https://www.researchgate.net/

IAs

4.3 Manejo de archivos

wget, curl, shasum, md5sum, diff, scp, rsync, gunzip, unzip, tar, head, tail, more, less, cat, >, », nano

Link para el genoma de Raoultella terrigena:

https://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/ Raoultella_terrigena/representative/GCF_0120

Para descargar archivos a nuestra computadora desde la terminal, se requiere utilizar protocolos de transferencia, los comandos `wget` o `curl` funcionan para ello. Veámos un ejemplo:

- Crea el directorio principal de trabajo para este proyecto y sus subdirectorios asociados.
- Accede al directorio data
- Descarga el genoma representativo de Raoultella terrigena de la Refseq de NCBI
- Ahora descarga las secuencias proteícas

¿Notaste algún cambio? - Vuelve a descargar el genoma de Raoultella terrigena, pero ahora asegúrate de guardar la secuencia en un archivo llamado Raoultella_terrigena.fasta.gz

```
wget
https://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/Raoultella_terrigena/r
epresentative/GCF_012029655.1_ASM1202965v1/GCF_012029655.1_ASM1202965v1_gen
omic.fna.gz .
```

```
#Para mobaxterm
wget --no-check-certificate
https://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/Raoultella_terrigena/r
epresentative/GCF_012029655.1_ASM1202965v1/GCF_012029655.1_ASM1202965v1_cds
_from_genomic.fna.gz .

#--no-check-certificate para saltarse los certificados (MobaXterm)
```

Descarga el archivo de aminoácidos de Raoultella terrigena

```
wget --no-check-certificate -O Raoultella_terrigena.faa.gz
https://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/Raoultella_terrigena/r
epresentative/GCF_012029655.1_ASM1202965v1/GCF_012029655.1_ASM1202965v1_pro
tein.faa.gz .
```

¿Qué ocurrió?

`curl` Realiza la misma función básica que wget , las diferencias principales son: El output lo imprime a standar output, imprime varias estadísticas útiles sobre la descarga.

```
curl
https://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/Raoultella_terrigena/r
epresentative/GCF_012029655.1_ASM1202965v1/GCF_012029655.1_ASM1202965v1_pro
tein.faa.gz .

curl
https://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/Raoultella_terrigena/r
epresentative/GCF_012029655.1_ASM1202965v1/GCF_012029655.1_ASM1202965v1_pro
tein.faa.gz > Raoultella_terrigena2.faa.gz
```

¿Cómo comprobamos que dos archivos son idénticos? Hemos descargado dos veces la secuencia genómica de *Raoultella terrigena*. Comprobemos que ambos archivos son idénticos.

Prueba con `diff -s`

```
diff GCF_012029655.1_ASM1202965v1_genomic.fna.gz Raoultella_terrigena.fasta.gz
```

¿Y si comparas el faa vs fasta?

```
diff ../data/Raoultella_terrigena.faa.gz ../data/Raoultella_terrigena.fasta.gz
```

4.4 Redireccionamientos y evaluación de la integridad

> Permite direccionar un resultado a un archivo nuevo. Crea el archivo si no existe y lo sobre escribe si existe.

>> Permite redireccionar un resultado en pantalla o un archivo a otro, sin remplazar o sobreescribir. Crea el archivo si no existe y agrega el nuevo contenido al final, si el archivo existe.

`shasum` y **`md5sum`** son programas que generan una suma encriptada única para cada archivo.

Revisemos la integridad de los archivos, descarga el archivo md5checksums.txt del directorio genomes/refseq/bacteria/Raoultella terigena de NCBI

```
curl
https://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/Raoultella_terrigena/r
epresentative/GCF_012029655.1_ASM1202965v1/md5checksums.txt >
md5sum_R.terrigena.txt
```

```
shasum Raoultella_terrigena.fasta.gz
md5sum Raoultella_terrigena.fasta.gz
more md5sum_R.terrigena.txt
```

Redireccionemos los resultados de integridad a un archivo nuevo.

```
md5sum Raoultella_terrigena.fasta.gz >
../resuts/R.terrigena.md5sum.check

cat ../results/R.terrigena.msd5sum.check
cat md5sum_R.terrigena.txt >> ../results/R.terrigena.md5sum.check
cat ../results/R.terrigena.msd5sum.check
```

4.5 Transferencia de archivos

Esta parte no la vamos a practicar porque estamos usando una interfaz gráfica con acceso al servidor, pero si no tienes esta forma de acceso es necesario usar estos comandos, asi que te dejamos el ejemplo. :)

```
`scp` [FUENTE][DESTINO]
#FUENTE=Nombre del archivo que quieres transferir
#DESTINO=Ruta de destino
#Ejemplo de mi computadora al servidor:
scp md5sum_R.terrigena.txt hoaxaca@132.248.220.35:/space31/PEG/hoaxaca
#Me pedirá el password
#Funciona de manera inversa si quieres bajar del servidor a tu computadora
scp hoaxaca@132.248.220.35:/space31/PEG/hoaxaca/md5sum_R.terrigena.txt .#ojo el
destino puede ser con ruta absoluta o relativa, aquí fue relativa "."
#También puedes usar rsync
`rsync -e ssh` [FUENTE][DESTINO]
# -e ssh indica que nos conectaremos al servidor a través de una conexión de
tipo ssh
```

4.6 Compresión y descompresión

Para descomprimir archivos usamos **gunzip**.

```
gunzip Raoultella_terrigena.fasta.gz
gunzip *.gz
```

Para comprimir usamos **gzip**.

```
gzip *.f*
```

¿Y si quiero comprimir directorios? Para ello utilizo tar que es un método de ultra compresión, muy utilizada en datos genómicos. Vamos a comprimir el directorio practica1

```
cd ../../
tar cvzf practica1.tar.gz practica1/
# c = crea un nuevo directorio
# v = muestra el progreso dde la compresión
# z = genera un archivo comprimido en zip (.gz)
# f = para indicar el nombre del archivo comprimido
```

Y para descomprimir?

```
rm -r practica1/
tar -xvf practica1.tar.gz
# ¿qué hace el flag x?
```

4.7 Explorar archivos

Veamos las primeras líneas del archivo de anotación .gtf del genoma de Raoultella terrigena

```
#¿Y, cuál es ese, lo tenemos?
cd sesion2/test/
#Ahora vamos a descargar desde test a data/
wget -O Raoultella_terrigena.gtf
https://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/Raoultella_terrigena/repre
sentative/GCF_012029655.1_ASM1202965v1/GCF_012029655.1_ASM1202965v1_genomic.gtf
.gz ../data/
gunzip ../data/Raoultella_terrigena.gtf
#Ahora si
head ../data/Raoultella_terrigena.gtf
# ¿Y si quiero ver las primeras 20 lineas?
head -n 20 ../data/Raoultella_terrigena.gtf
```

Ahora quiero ver las últimas líneas de un archivo

```
tail ../data/Raoultella_terrigena.gtf
```

Veamos el genoma

```
more ../data/Raoultella_terrigena.fasta
# Enter => Navegar hacia abajo de línea en línea
# Espacio => Navega hacia abajo de pantalla en pantalla
```

Pero podemos ver archivos con algo más potente

```
less ../data/Raoultella_terrigena.gtf
```

Espacio OR Enter => Navegar hacia abajo

b OR flecha arriba => Navegar hacia arriba

/WORD => Búsqueda forward

n => Siguiente

?WORD => Búsqueda backward

N => Anterior

G => Ir al final del archivo

g => Ir al inicio del archivo

-S => Mostrar una línea por renglón

4.8 Ejercicio 02

Estás realizando tu proyecto con Raoultella terrigena y tu directora de tesis te pregunta si hay genes nitrogenasa ubicados en la posición 501417 o 3010433 del genoma representativo de los genomas de referencia de esta especie. Tú debes responderle si están o no en esa posición, y si no están, avisarle en qué posición se encuentran y cuántos son.

Pseudocódigo Es la manera en la que planeas, diseñas la ruta de trabajo que usarás para responder una pregunta. Es el diseño experimental.

Pseudocódigo:

1. Crea el directorio Practica2
2. Descarga el archivo de anotación .gff de Raoultella terrigena en el directorio correspondiente.
3. Revisa su integridad y redirecciona el resultado al directorio correspondiente
4. Descomprime el archivo de anotación
5. Navega al final del archivo
6. Navega al inicio del archivo
7. Busca el gene que inicie en la posición 501417
8. Busca el CDS que termine en la posición 3010433
9. Busca los genes nitrogenasa

10. Escribe los resultados

Chapter 5

Filtrar información

En esta sesión, exploraremos los comandos `|`, `sort`, `cut`, `uniq`, `wc` y `grep` para analizar el genoma de *Raoultella terrigena*. Estos comandos son esenciales para procesar datos en la terminal de manera eficiente.

5.1 Recordatorio: Comandos Básicos

1. Lista los contenidos del directorio raíz y busca si existe un directorio llamado `home` usando `less`:

```
ls /
ls / | less
# Resultado esperado: /home
```

2. Lista los primeros 10 archivos del directorio raíz:

```
ls / | head -10
```

3. Lista los últimos 5 archivos del directorio raíz:

```
ls / | tail -5
```

5.2 Introducción a los Comandos

- **Pipe (`|`)**: Permite conectar la salida de un programa con la entrada de otro, procesando datos en RAM para mayor rapidez, evitando escritura/lectura en disco. A diferencia de `&&`, que ejecuta comandos independientes, `|` requiere que la salida de un comando sea la entrada del siguiente.
- **sort**: Ordena líneas de texto.
- **cut**: Extrae secciones de cada línea.

- **uniq**: Filtra líneas repetidas (requiere que el archivo esté ordenado).
- **wc**: Cuenta líneas, palabras, caracteres o bytes.
- **grep**: Busca patrones en archivos.

Estructura de un Archivo de Anotación (GFF)

Un archivo GFF tiene las siguientes columnas:

seqname: Nombre del cromosoma.

source: Programa que generó el elemento.

feature: Tipo de elemento (e.g., gene, CDS).

start: Posición de inicio.

end: Posición de final.

score: Valor de punto flotante.

strand: Cadena (+, -).

frame: Marco de lectura.

attribute: Pares tag-value con información adicional.

5.3 Features en el Genoma

Tamaño del Genoma de Raoultella terrigena

¿Qué archivo necesitamos?

El archivo Raoultella_terrigena.fasta.

¿Qué comando nos ayuda?

```
wc data/Raoultella_terrigena.fasta
```

Nota: Este comando da una estimación aproximada, ya que incluye bytes del encabezado y saltos de línea.

Guarda los resultados en un archivo:

```
mkdir -p results/sesion3
```

```
echo 'El genoma de Raoultella puede medir:' > results/sesion3/Raoultella_caracteristica
```

```
wc data/Raoultella_terrigena.fasta >> results/sesion3/Raoultella_caracteristicas.txt
```

Número de Cromosomas

¿Qué archivo necesitamos?

El archivo Raoultella_terrigena.gff.

¿Qué comando nos ayuda?

```
cut -f1 data/Raoultella_terrigena.gff | head
```

Nota: El resultado incluye las 8 líneas del encabezado. Para excluirlas:

```
grep -v "#" data/Raoultella_terrigena.gff | cut -f1 | sort | uniq
```

Guarda el resultado

```
echo 'El genoma de Raoultella tiene un cromosoma y es' >> results/sesion3/Raoultella_caracteristi
grep -v "#" data/Raoultella_terrigena.gff | cut -f1 | uniq >> results/sesion3/Raoultella_caracter
```

Número de features

¿Qué archivo necesitamos?

El archivo Raoultella_terrigena.gff.

¿Qué comandos requerimos?

```
cut -f3 data/Raoultella_terrigena.gff | uniq
```

Nota: uniq requiere que las líneas estén ordenadas. Prueba:

```
cut -f3 data/Raoultella_terrigena.gff | sort | uniq
# Alternativa:
cut -f3 data/Raoultella_terrigena.gff | sort -u
```

Número de tipos de features

```
cut -f3 data/Raoultella_terrigena.gff | sort -u | wc -l
```

Quita las lineas comentadas

Fuentes de los datos de anotación

```
cut -f2 data/Raoultella_terrigena.gff | sort -u
```

Número de genes y CDS

Pseudocódigo

1. Acceder a la columna 3 (feature).
2. Contar ocurrencias únicas de cada elemento.

```
cut -f3 data/Raoultella_terrigena.gff | sort | uniq -c
```

Para evitar contar elementos repetidos:

```
cut -f3-5 data/Raoultella_terrigena.gff | sort -u | cut -f1 | sort | uniq -c
```

Genes por cadena

Pseudocódigo

1. Cortar las columnas feature y strand.
2. Ordenar y contar ocurrencias únicas.

```
cut -f3,7 data/Raoultella_terrigena.gff | sort | uniq -c
```

Crea un archivo de anotación ordenado por cadena y por región genómica

Pseudocódigo

1. Acceder a las columnas de cadena (strand) y posiciones genómicas.
2. Ordenar por cadena y luego por posición (numéricamente).

```
sort -k7,7 -k4,4n data/Raoultella_terrigena.gff > results/R_terrigena_strand.gff
```

Cuántos genes hay con diferente nombre?

Pseudocódigo

1. Filtrar registros de tipo gene.
2. Acceder a la columna 9 (atributos).
3. Separar por ; y extraer nombres.
4. Contar valores únicos.

```
grep -P "\tgene\t" data/Raoultella_terrigena.gff | cut -f9 | cut -d ';' -f3 | sort -u
```

5.4 Ejercicios

- **1. Cuántos genes hay con distinto ID?**

  Pseudocódigo

1. Filtrar registros de tipo gene.
2. Acceder a la columna 9.
3. Separar por ; y =, quedándote con el ID.
4. Contar valores únicos.

- **2. Cuántas secuencias proteícas?**

  Tip: usa el archivo de proteinas

- **3. Cuánto mide la secuencia de la proteína WP_000448832.1?**

  Pseudocódigo

  1. Localizar el ID `WP_000448832.1`.
  2. Extraer la secuencia y contar caracteres.

- 4. Cuál es el ID del gene `fnr`

Chapter 6

En construccion

Chapter 7

En construcción

Chapter 8

En construcción también