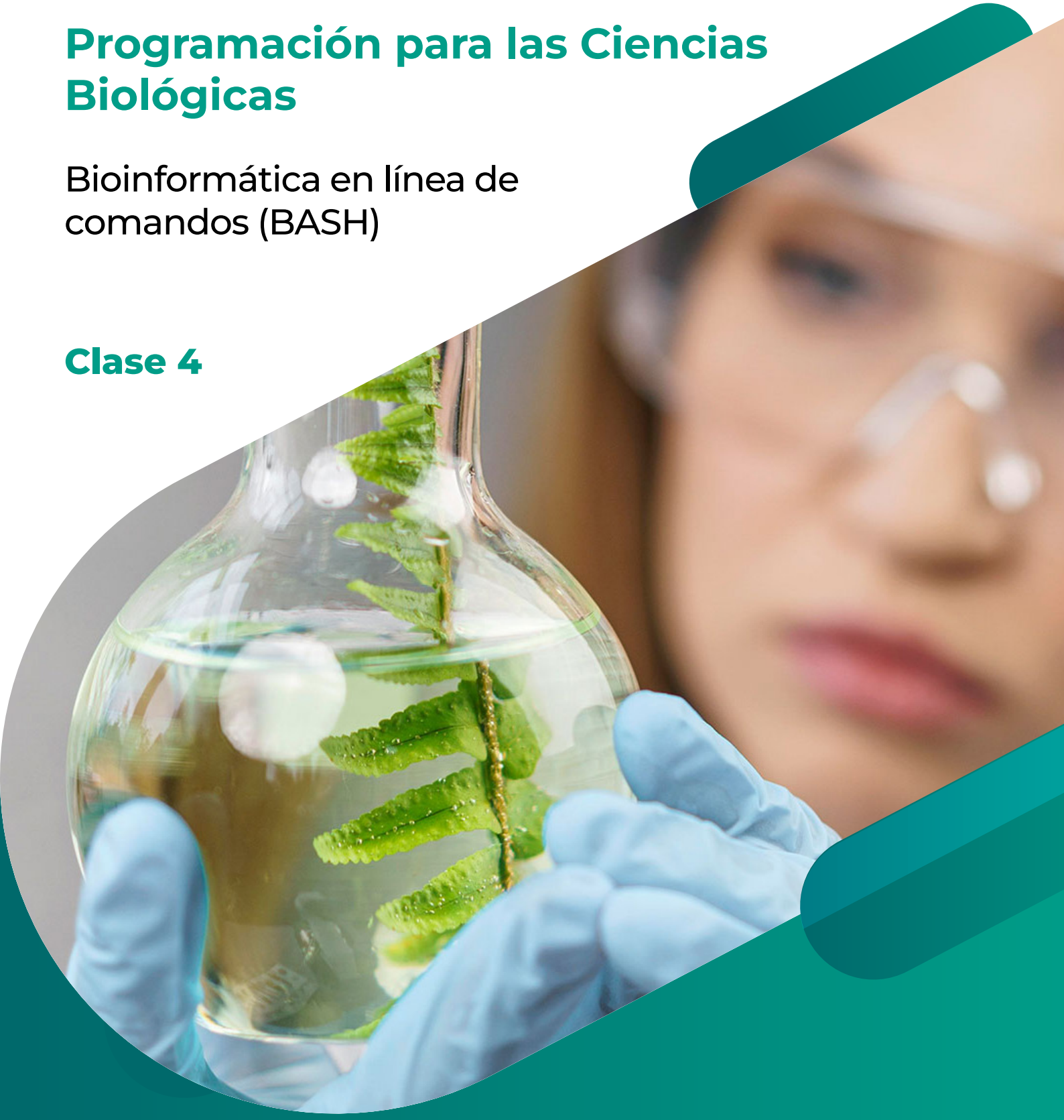
Programación para las Ciencias Biológicas

Bioinformática en línea de comandos (BASH)



Clase 4

MAESTRÍA EN MAESTRÍA EN BIOLOGÍA COMPUTACIONAL

La excelencia no se improvisa

1. INTRODUCCIÓN DE LA CLASE

Bienvenidos a la última sesión de programación con BASH, en la cual desarrollaremos scripts para el análisis bioinformático

Clase 4:

RDA1. Desarrollar algoritmos con lenguaje de programación estructurado de línea de comandos para secuencias biológicas.

1.7. Bioinformática en línea de comandos (BASH)

- 1.7.1. Manejo de archivos de secuencias con scripts en BASH
- 1.7.1.1. Desarrollo de scripts para realizar búsquedas, análisis y transformación de secuencias de ADN y ARN.
- 1.7.1.2. Lectura y procesamiento de archivos en formato FASTA y FASTQ desde scripts en BASH.
- 1.7.2. Fundamentos para desarrollar scripts en BASH aplicados a bioinformática
- 1.7.2.1. Introducción a las bibliotecas y herramientas bioinformáticas que se integran con BASH.
- 1.7.2.2. Fundamentos de buenas prácticas en el desarrollo de scripts bioinformáticos, incluyendo manejo de errores, comentarios y modularización.

1.8. Desarrollo de scripts avanzados para trabajar con secuencias bioinformáticas

- 1.8.1. Análisis automatizado de secuencias de ADN y ARN
- 1.8.1.1. Desarrollo de scripts avanzados para análisis de secuencias largas, transcripción de ADN a ARN, identificación de patrones en secuencias, y alineamientos básicos.
- 1.8.2. Automatización de tareas bioinformáticas repetitivas
- 1.8.2.1. Creación de scripts que permitan automatizar procesos bioinformáticos como la conversión de secuencias y el análisis de datos en múltiples archivos.
- 1.8.2.2. Ejercicios de automatización en el procesamiento de grandes volúmenes de datos biológicos.

1.9. Scripts para transcriptómica en línea de comandos (BASH)

- 1.9.1. Desarrollo de scripts especializados para transcriptómica:
- 1.9.1.1. Conversión automatizada de secuencias de ADN a ARN.
- 1.9.1.2. Análisis de la composición de bases nitrogenadas en secuencias transcriptómicas.
- 1.9.1.3 Ejemplos prácticos de procesamiento masivo de datos transcriptómicos desde el terminal.
- 1.9.1.4 Integración con herramientas bioinformáticas como samtools para el análisis de alineamientos de secuencias de ARN.

Referencias citadas en la Clase 3.

Videos:

1. Ejercicios de automatización en el procesamiento de grandes volúmenes de datos biológicos

https://seracademia.com/academia/Bio_Info/M4/Videos/Cuaderno5_BASH.mp4
https://seracademia.com/academia/Bio_Info/M4/Videos/Cuaderno6_BASH.mp4

2. Script Bash para transcribir ADN a ARN
https://seracademia.com/academia/Bio_Info/M4/Videos/Cuaderno6_BASH.mp4

Cuadernos de trabajo Jupyter

1. Cuaderno5_BASH_Apellido_Nombre.ipynb
2. Cuaderno6_BASH_Apellido_Nombre.ipynb

Referencias Bibliográficas

1. SeqKit - a cross-platform and ultrafast toolkit for FASTA/Q file manipulation, Disponible en:
<https://github.com/shenwei356/seqkit?tab=readme-ov-file>

2. Wei Shen*, Botond Sipos, and Liuyang Zhao. 2024. SeqKit2: A Swiss Army Knife for Sequence and Alignment Processing. iMeta e191. doi:10.1002/imt2.191.

3. crontab Command (IBM, Zadnje ažuriranje, 2023), disponible en
<https://www.ibm.com/docs/hr/aix/7.1?topic=c-crontab-command>

Definición de los términos citados en la Clase 4.

BASH Script: Un script en BASH es un archivo de texto que contiene una serie de comandos y estructuras de programación diseñadas para ser ejecutadas por el intérprete BASH (Bourne Again SHell). Estos scripts permiten automatizar tareas en sistemas operativos Unix/Linux, como manipular archivos, gestionar procesos, o ejecutar secuencias complejas de comandos.

Transcripción (ADN): Proceso biológico mediante el cual la información genética de una cadena de ADN se copia en una molécula de ARN mensajero (ARNm). Esta conversión se realiza gracias a la enzima ARN polimerasa, que "lee" la hebra molde de ADN (orientación 3'→5') y sintetiza una cadena complementaria de ARN (orientación 5'→3'). El ARN resultante sirve como intermediario para trasladar las instrucciones genéticas desde el núcleo celular hasta los ribosomas, donde se traducirán en proteínas. Es el primer paso de la expresión génica y no implica la replicación del ADN original.

Shebang: En sistemas Unix/Linux, el shebang (también llamado hashbang) es una línea especial al inicio de un script (por ejemplo, #!/bin/bash) que indica al sistema operativo qué intérprete debe usar para ejecutar el código. Compuesto por los caracteres #! seguidos de la ruta del ejecutable (como /bin/bash para Bash o /usr/bin/python3 para Python), permite que el script se autodetermine y se ejecute directamente desde la terminal sin necesidad de especificar manualmente el intérprete. Es fundamental para garantizar la portabilidad y funcionalidad de los scripts en diferentes entornos.

Cron: Sistema de planificación de tareas en entornos Unix/Linux que permite ejecutar automáticamente comandos, scripts o programas a horas, días o intervalos específicos (ej: cada minuto, a las 3 AM diariamente). Los trabajos programados (cron jobs) se definen en un archivo llamado crontab (tabla cron), siguiendo un formato estándar

Crontab: Utilidad en sistemas Unix/Linux que define y gestiona tareas programadas (cron jobs) mediante un archivo de configuración (o tabla cron), donde cada línea especifica un comando o script a ejecutar en intervalos regulares (minuto, hora, día, etc.), permitiendo automatizar procesos como copias de seguridad, actualizaciones o ejecución de programas sin intervención manual.

Se accede y edita con el comando crontab -e, y sigue el formato: <minuto> <hora> <día-del-mes> <mes> <día-de-la-semana> <comando>.

stderr (Standard Error): Flujo de salida estándar (file descriptor 2) utilizado por programas y comandos en sistemas Unix/Linux para enviar mensajes de error, advertencias o información de depuración. A diferencia de stdout (Standard Output), que se usa para la salida normal de un programa, stderr está diseñado específicamente para manejar errores de manera independiente, lo que permite redirigir o suprimir estos mensajes sin afectar la salida principal. Por defecto, stderr se muestra en la terminal, pero puede redirigirse a archivos, descartarse (usando /dev/null) o combinarse con stdout para un manejo más flexible en scripts y comandos.

find: Comando en sistemas Unix/Linux que permite buscar archivos y directorios en una jerarquía de directorios basándose en criterios específicos, como nombre, tipo, tamaño, fecha de modificación o permisos. Es una herramienta poderosa y flexible que admite la ejecución de acciones sobre los resultados encontrados (por ejemplo, eliminar, mover o ejecutar comandos).

Profundización Clase 4.

Clase 4:

RDA1: Desarrollar algoritmos con lenguaje de programación estructurado de línea de comandos para secuencias biológicas.

1.7. Bioinformática en línea de comandos (BASH)

Los scripts en BASH son herramientas poderosas para bioinformática, permitiendo la automatización de tareas repetitivas y el procesamiento eficiente de grandes volúmenes de datos biológicos. Su facilidad de uso en la línea de comandos de sistemas Unix/Linux o WSL[1] los hace ideales para la manipulación de archivos, integración de herramientas bioinformáticas y creación de flujos de trabajo reproducibles. Con este lenguaje de programación es posible realizar desde tareas simples, como renombrar archivos masivamente, hasta complejos análisis de secuencias utilizando herramientas externas como BLAST, SAMtools y BEDtools.

1.7.1. Manejo de archivos de secuencias con scripts en BASH

El manejo de archivos de secuencias es una tarea común en bioinformática y BASH proporciona herramientas eficientes para su procesamiento tanto con el manejo individual de archivo como de manera global considerando su capacidad para procesar conjuntos de archivos lo que lo vuelve ideal para extraer información de fuentes contenidas en distintos archivos, entre otras es posible realizar:

- Lectura de archivos
- Extracción de información
- Filtrado de secuencias
- Conversión de formatos

Estas operaciones que pueden realizarse manualmente en la línea de comandos, es posible automatizarlas a través de scripts, e incluso de ser necesario calendarizar su ejecución. En nuestro caso, luego de desarrollar las competencias y familiarizarnos con el uso del terminal GNU/Linux , estamos listos para elaborar nuestros propios scripts.

1.7.1.1. Desarrollo de scripts para realizar búsquedas, análisis y transformación de secuencias de ADN y ARN.

---

[1] El Subsistema de Windows para Linux (WSL) permite usar GNU/Linux de manera nativa sobre Windows, sin embargo puede ser menos eficiente que un sistema GNU/Linux instalado directamente en la máquina.

El análisis de secuencias de ADN y ARN es una tarea fundamental en bioinformática, y BASH permite realizar búsquedas, análisis y transformaciones de manera eficiente. Algunas de las operaciones comunes incluyen:

- **Búsqueda de secuencias específicas:** Usar herramientas como grep para localizar secuencias de interés en archivos FASTA o FASTQ.
- **Cálculo de contenido GC:** Determinar la proporción de bases guanina y citosina en una secuencia.
- **Transcripción de ADN a ARN:** Sustitución de timinas (T) por uracilos (U).
- **Traducción de ARN a proteínas:** Aplicar tablas de codones para obtener la secuencia de aminoácidos.
- **Filtrado y eliminación de secuencias:** Uso de herramientas como sed y awk para limpiar los datos.

1.7.1.2. Lectura y procesamiento de archivos en formato FASTA y FASTQ desde scripts en BASH.

La lectura y procesamiento de archivos en formato FASTA y FASTQ usando scripts en BASH es un proceso común en bioinformática, estos scripts, de acuerdo a las necesidades del investigador, puede ser desarrollados para tareas especificas o se puede usar una gran cantidad de programas ya creados por la comunidad y que pueden ser lanzados desde terminal, independientemente si estos fueron elaborados en BASH o no, pero que pueden correr en un terminal GNU/Linux.

Entre otras, las siguientes herramientas se pueden usar desde un terminal

seqkit: Es una biblioteca de software de procesamiento de secuencias de ADN, ARN y proteínas para sistemas operativos Linux. Fue diseñada para ser rápida, eficiente y escalable.

```
jupyter@Debian12-server:~/Scripts/Bioinfo$ seqkit stats sampledata/t-coffee/sproteases_small_dna.fasta
file                                          format  type  num_seqs  sum_len  min_len  avg_len  max_len
sampledata/t-coffee/sproteases_small_dna.fasta  FASTA   DNA         19   14,601      657    768.5      825
```

Para mayores detalles de seqkit, puede revisar su sitio oficial o la ayuda contextual de la herramienta.

fastqc: un comando para verificar la calidad de los datos en archivos FASTQ y que se usó previamente.

De manera complementaria y respondiendo a sus propias necesidades los investigadores pueden desarrollar sus propios scripts, lo que veremos con mayor detalle mas adelante.

1.7.2. Fundamentos para desarrollar scripts en BASH aplicados a bioinformática

Para poder desarrollar scripts en BASH es necesario que que tenga dominio de los siguientes temas:

1. **Conocimientos de BASH**: conocimiento de la sintaxis y los comandos básicos de BASH.
2. **Sistema operativo GNU/Linux**: tener acceso a un sistema operativo Linux con el entorno de BASH instalado.
3. **Herramientas bioinformáticas**: familiarizarse con las herramientas de bioinformática comunes, como fastqc, seqkit, etc.
4. **Comandos de línea de comandos**: conocer comandos básicos de BASH, como `cd`, `pwd`, `ls`, `mkdir`, etc.
5. **Conocer las estructuras de programación en BASH**: if, for, while, etc

Si desconoce alguno de estos elementos puede regresar a los capítulos anteriores de este curso para fortalecer sus conocimientos y regresar a este punto una vez que los domine.

Adicionalmente es deseable tener las siguientes habilidades:

1. **Programación**: Conocimientos básicos de programación en BASH o lenguajes de programación similares.
2. **Análisis y resolución de problemas**: Habilidad para analizar problemas y encontrar soluciones efectivas.
3. **Comunicación**: Habilidad para comunicar ideas y resultados a otros.
4. **Colaboración**: Habilidad para trabajar en equipo con otros bioinformáticos.

De manera complementaria, es recomendable utilizar los siguientes recursos:

1. **Documentación oficial**: Documentación oficial de BASH y herramientas de bioinformática.
2. **Manuales y guías**: Manuales y guías de instrucción para BASH y herramientas de bioinformática.
3. **Comunidades en línea**: Comunidades en línea como Stack Overflow, Reddit, etc.
4. **Cursos y talleres**: Cursos y talleres en línea sobre BASH y bioinformática.

Finalmente es importante recordar que experiencia que se va alcanzando con la práctica es fundamental para desarrollar scripts de BASH para bioinformática

1.7.2.1. Introducción a las bibliotecas y herramientas bioinformáticas que se integran con BASH.

En el campo de la bioinformática, el uso de herramientas y bibliotecas que se integran con el entorno de línea de comandos de Unix/Linux, específicamente con BASH (Bourne Again SHell), es fundamental para el procesamiento, análisis y visualización de datos biológicos. BASH es ampliamente utilizado debido a su flexibilidad, potencia y capacidad para automatizar tareas repetitivas. A continuación, se presenta una introducción a algunas de las bibliotecas y herramientas bioinformáticas más comunes que se integran con BASH.

1. Herramientas de procesamiento de secuencias

Las herramientas de procesamiento de secuencias son indispensables en el análisis bioinformático para manipular, filtrar y transformar datos genómicos de manera eficiente. Entre ellas destacan FASTX-Toolkit, FASTQ, BEDTools y SAMTools. Estas herramientas, al operar desde la línea de comandos de BASH, permiten automatizar flujos de trabajo complejos, integrándose en pipelines para tareas como limpieza de secuencias, análisis de variantes o comparación de regiones genómicas, garantizando precisión y reproducibilidad en estudios de genómica, transcriptómica o epigenética.

FASTX-Toolkit: Conjunto de herramientas de línea de comandos para el procesamiento de secuencias de ADN y ARN en formato FASTQ/FASTA. Permite realizar tareas como recorte de adaptadores, filtrado por calidad y conversión de formatos.

BEDTools: Utilidades para manipular archivos en formato BED (Browser Extensible Data), que se utilizan para representar regiones genómicas. Permite operaciones como intersecciones, uniones y coberturas entre conjuntos de datos genómicos.

SAMtools: Herramienta para manipular archivos en formato SAM/BAM (Sequence Alignment/Map), que contienen alineamientos de secuencias. Permite la visualización, ordenación, indexación y extracción de datos de alineamientos.

2. Herramientas de alineamiento de secuencias

BWA (Burrows-Wheeler Aligner): Alineador de secuencias cortas que utiliza el algoritmo de Burrows-Wheeler para mapear secuencias de ADN contra un genoma de referencia. Es ampliamente utilizado en estudios de resecuenciación.

Bowtie: Alineador rápido y eficiente para mapear secuencias cortas contra un genoma de referencia. Es especialmente útil en aplicaciones de ChIP-seq y RNA-seq.

STAR (Spliced Transcripts Alignment to a Reference): Alineador diseñado específicamente para datos de RNA-seq, capaz de manejar empalmes (splicing) y alinear lecturas a través de intrones.

3. Herramientas de análisis de datos genómicos

GATK (Genome Analysis Toolkit): Conjunto de herramientas desarrollado por el Broad Institute para el análisis de variantes genéticas. Incluye funcionalidades para la detección de SNPs, indels y variantes estructurales.

VCFtools: Utilidades para manipular y analizar archivos en formato VCF (Variant Call Format), que contienen información sobre variantes genéticas. Permite filtrar, comparar y calcular estadísticas sobre variantes.

Plink: Herramienta para realizar análisis de asociación genética y manipulación de datos de genotipos. Es ampliamente utilizado en estudios de asociación de todo el genoma (GWAS).

4. Herramientas de visualización y reportes

IGV (Integrative Genomics Viewer): Aunque IGV es una herramienta gráfica, se puede integrar con BASH para generar visualizaciones de datos genómicos a partir de archivos BAM, VCF y otros formatos.

MultiQC: Herramienta que agrega y visualiza informes de calidad de múltiples herramientas bioinformáticas (como FastQC, STAR, SAMtools, etc.) en un solo informe HTML. Es útil para evaluar la calidad de los datos y los resultados de los análisis.

5. Bibliotecas y lenguajes de programación integrados con BASH

Biopython: Biblioteca de Python que proporciona herramientas para el manejo de datos biológicos, como secuencias, estructuras y alineamientos. Se puede integrar con BASH mediante scripts de Python.

BioPerl: Similar a Biopython, pero en Perl. Ofrece una amplia gama de módulos para el análisis de datos biológicos y se puede ejecutar desde la línea de comandos.

R (Bioconductor): Aunque R es un lenguaje de programación independiente, se puede integrar con BASH para realizar análisis estadísticos y visualizaciones avanzadas de datos genómicos. Bioconductor es un repositorio de paquetes de R específicos para bioinformática.

6. Automatización y flujos de trabajo

Snakemake: Sistema de gestión de flujos de trabajo que permite definir y ejecutar pipelines de análisis de datos en BASH. Es especialmente útil para automatizar tareas complejas y reproducibles en bioinformática.

Nextflow: Herramienta para la creación de flujos de trabajo escalables y reproducibles. Permite integrar múltiples herramientas y lenguajes de programación en un solo pipeline.

1.7.2.2. Fundamentos de buenas prácticas en el desarrollo de scripts bioinformáticos, incluyendo manejo de errores, comentarios y modularización.

El desarrollo de scripts bioinformáticos en BASH es crucial para automatizar tareas y procesar grandes volúmenes de datos biológicos, estos se deben realizar siguiendo buenas prácticas de programación, lo que permitirá mejorar la eficiencia, la mantenibilidad y la reproducibilidad del código. Esto se traduce en un código fiable y de calidad que permite optimizar la colaboración entre investigadores y facilita la reproducibilidad de los análisis.

Algunos fundamentos clave para un desarrollo eficaz de scripts bioinformáticos en BASH a considerarse son:

Manejo de errores

El manejo adecuado de errores es fundamental para garantizar la robustez de un script. Algunos principios clave incluyen:

- **Validación de entradas**: Verificar que los archivos y parámetros proporcionados sean correctos utilizando condicionales (`if` y `case`).
- **Captura de errores**: Uso de `set -e` para detener la ejecución en caso de error y `set -u` para detectar variables no definidas.
- **Mensajes de error informativos**: Proporcionar retroalimentación clara al usuario utilizando `echo` y `stderr`.
- **Comprobación de comandos**: Verificar la ejecución exitosa de comandos mediante `$?` o `||`.

Comentarios y Documentación
La documentación clara es esencial para la comprensión y mantenimiento del código, en ese sentido las buenas prácticas incluyen:

- **Comentarios en el código**: Explicar el propósito de cada sección utilizando #.
- **Encabezados informativos**: Incluir información sobre el propósito, autor y fecha en la cabecera del script.
- **Nombres descriptivos**: Utilizar nombres significativos para variables y funciones.
- **Ejemplos de uso**: Proporcionar ejemplos de ejecución en los comentarios.

Modularización
La modularización consiste en dividir el código en funciones reutilizables, lo cual mejora la mantenibilidad y facilita la colaboración, así las prácticas recomendadas incluyen:

- **Definir funciones**: Encapsular tareas repetitivas en funciones reutilizables.
- **Separación de lógica y configuración**: Definir variables globales en un archivo de configuración separado.
- **Uso de scripts modulares**: Dividir funcionalidades en varios scripts que se llamen entre sí.
- **Control de flujo**: Usar `return` y `exit` de manera adecuada para controlar el flujo del script.

1.8. Desarrollo de scripts avanzados para trabajar con secuencias bioinformáticas

En el ámbito de la bioinformática, el desarrollo de scripts avanzados se ha convertido en una herramienta esencial para manipular, analizar y visualizar secuencias biológicas (ADN, ARN, proteínas) de manera eficiente. Estos scripts, escritos en lenguajes como Bash, permiten automatizar tareas complejas como el procesamiento de archivos FASTA/FASTQ, la identificación de patrones genéticos, la conversión entre formatos de secuencias o la integración con herramientas bioinformáticas (BLAST, samtools, etc.). Al combinar técnicas de programación con conocimientos biológicos, los scripts no solo optimizan el flujo de trabajo en investigación genómica o proteómica, sino que también facilitan la reproducibilidad, escalabilidad y personalización de análisis, siendo clave en estudios de mutaciones, filogenia o edición genética.

Los cuadernos de trabajo 5 y 6 se desarrollan sobre esta dinámica.

1.8.1. Análisis automatizado de secuencias de ADN y ARN

Dado el elevado tiempo requerido para analizar secuencias de ADN y ARN, donde cada etapa (alineación, filtrado, anotación) depende de resultados intermedios, la automatización mediante scripts y herramientas como cron se vuelve crítica. Estos flujos de trabajo, al programarse en scripts (Bash, Python), permiten encadenar procesos secuenciales (ej: procesar datos de NGS con FastQC, BWA, GATK) sin intervención manual, ejecutándose incluso en horarios no laborables para optimizar recursos. Herramientas como cron facilitan la planificación periódica de tareas (ej: análisis nocturnos), mientras que gestores de workflows (Snakemake, Nextflow) garantizan la continuidad ante fallos. Esta automatización no solo reduce errores humanos y horas de trabajo, sino que habilita estudios masivos y reproducibles, esenciales en proyectos de epidemiología, evolución molecular o diagnóstico clínico, donde la escalabilidad y la precisión son prioritarias.

1.8.1.1. Desarrollo de scripts avanzados para análisis de secuencias largas, transcripción de ADN a ARN, identificación de patrones en secuencias, y alineamientos básicos.

Las soluciones informáticas buscan satisfacer necesidades de procesos que pueden automatizarse a través de estas herramientas y en el campo de la Biología Computacional, como ejemplo, las operaciones que se realizan con las secuencias de ADN se pueden automatizar a través de scripts, en este caso con el lenguaje de programación BASH:

Operaciones con secuencias de ADN

- Replicación (se duplica o replica)
- Transcripción (el ADN se convierte en ARN)
- Traducción (el ARN se usa como instrucciones para crear proteínas)

La transcripción consiste en la síntesis de ARN tomando como molde el ADN y significa el paso de la información contenida en el ADN hacia el ARN. La transferencia de la información del ADN hacia el ARN se realiza siguiendo las reglas de complementaridad de las bases nitrogenadas y es semejante al proceso de transcripción de textos, motivo por el que ha recibido este nombre. El ARN producto de la transcripción recibe el nombre de transcritoy para obtenerlo se puede crear un programa que realice los reemplazos necesarios en la cadena de ADN para conseguir el transcrito

La dirección en la que las ARN polimerasas sintetizan ARN es siempre 5'P→3'OH, es decir el ARN producto de la transcripción crece solamente en esta dirección. Recuerde que la dirección en la que las ADN polimerasas sintetizan ADN es también la misma 5'P→3'OH.

Asimetría de la transcripción: la asimetría de la transcripción significa que solamente se transcribe para cada gen una de las dos hélices de ADN, la hélice que se toma como molde para producir el ADN se la denomina hélice codificadora o hélice con sentido y la otra hélice de ADN, la que no se transcribe, se la denomina hélice estabilizadora o hélice sin sentido.

Para traducir desde el ADN al ARNm mensajero tomaremos en cuenta las siguientes equivalencias de traducción partiendo desde la hélice codificadora:

ADN	**ARN**
T (Timina):	A (Adenina)
A (Adenina):	(Uracilo)
C (Citosina):	G (Guanina)
G (Guanina):	C (Citosina)

Tabla: Equivalencias de traducción desde el ADN al ARNm mensajero

Como podemos ver en la traducción la Timina se reemplaza por una Adenina, la Adenina por un Uracilo, la Citosina por una Guanina y la Guanina por una Citocina.

Adicionalmente el ARNm mensajero tiene definidos los codones (tripletes) de inicio y fin:

Inicio	**Fin**
AUG	UAA
AUG	UAG
AUG	UGA

1.8.2. Automatización de tareas bioinformáticas repetitivas

La automatización de tareas bioinformáticas repetitivas puede realizarse mediante la creación de scripts en BASH que ejecuten las tareas necesarias utilizando comandos y herramientas de bioinformática. Por ejemplo, para automatizar una tarea como la búsqueda de BLAST, se puede crear un script que utilice el comando `blastn` para buscar homología entre secuencias de ADN o proteínas. El script puede ser personalizado para incluir opciones de búsqueda específicas y archivos de entrada/output.

También se pueden utilizar herramientas como cron para programar la ejecución del script a intervalos regulares, por ejemplo, cada semana o mes. Otra forma de automatizar tareas bioinformáticas es mediante el uso de frameworks y bibliotecas de código abierto, como Biopython o BioConductor. Estos framework permiten interactuar con herramientas de bioinformática como BLAST, Genbank, Fastq, etc., utilizando una sintaxis más legible y fácil de entender.

1.8.2.1. Creación de scripts que permitan automatizar procesos bioinformáticos como la conversión de secuencias y el análisis de datos en múltiples archivos.

La creación de scripts para automatizar procesos bioinformáticos comienza con una planificación cuidadosa y un entendimiento claro de los objetivos del análisis. A continuación, se describen una ruta posible para iniciar este proceso:

1. Definir el objetivo y los requisitos del script

- **Identificar la tarea a automatizar:** Determinar qué proceso bioinformático se desea automatizar, como la conversión de secuencias (por ejemplo, FASTQ a FASTA), el análisis de calidad de datos, el alineamiento de secuencias o la detección de variantes.

- **Listar las herramientas necesarias:** Seleccionar las herramientas bioinformáticas que se utilizarán, como FASTX-Toolkit, SAMtools, BWA, STAR, GATK, entre otras.

- **Especificar los formatos de entrada y salida:** Definir los tipos de archivos que se manejarán (por ejemplo, FASTQ, BAM, VCF) y cómo se organizarán los resultados (por ejemplo, en directorios específicos o en un informe consolidado).

2. Diseñar el flujo de trabajo

- **Desglosar el proceso en pasos:** Dividir la tarea en pasos lógicos y secuenciales. Por ejemplo, para un análisis de RNA-seq, los pasos podrían incluir: control de calidad, alineamiento, cuantificación de genes y análisis diferencial.

- **Planificar la gestión de archivos:** Decidir cómo se manejarán múltiples archivos, ya sea mediante bucles en BASH o utilizando herramientas como find o xargs. También es importante definir cómo se nombrarán y organizarán los archivos de salida.

- **Considerar la escalabilidad y reproducibilidad:** Asegurarse de que el script pueda manejar grandes volúmenes de datos y que sea fácil de ejecutar en diferentes entornos o por otros usuarios.

3. Escribir el script en BASH

- **Crear un archivo de script:** Iniciar un nuevo archivo con extensión .sh (por ejemplo, procesamiento.sh) y agregar el shebang (#!/bin/bash) en la primera línea para indicar que es un script de BASH.

- **Implementar los pasos del flujo de trabajo:** Escribir los comandos necesarios para cada paso del proceso. Por ejemplo, usar fastq_to_fasta de FASTX-Toolkit para convertir archivos FASTQ a FASTA, o samtools sort para ordenar archivos BAM.

- **Agregar bucles para manejar múltiples archivos:** Utilizar estructuras como for file in *.fastq; do ... done para procesar todos los archivos en un directorio.

- **Incluir validaciones y manejo de errores:** Agregar verificaciones para asegurarse de que los archivos existan y que los comandos se ejecuten correctamente. Usar if statements y redireccionar mensajes de error (2> error.log).

1.8.2.2. Ejercicios de automatización en el procesamiento de grandes volúmenes de datos biológicos.

Actividad:
Desarrolle el cuaderno de trabajo 5 "**Cuaderno5_BASH_Apellido_Nombre.ipynb**"

1.9. Scripts para transcriptómica en línea de comandos (BASH)

Actividad:
Desarrolle el cuaderno de trabajo 5 "**Cuaderno5_BASH_Apellido_Nombre.ipynb**"

En este cuaderno:

- Trabajará con múltiples carpetas y archivos
- Usará el wildcard "*"
- Programará tareas con cron

1.9.1. Análisis de la composición de bases nitrogenadas en secuencias transcriptómicas.

El análisis de la composición de bases nitrogenadas en secuencias transcriptómicas es fundamental para comprender la regulación génica y la diversidad funcional del ARN. Este estudio explora la distribución de adenina, citosina, guanina y uracilo en transcritos, vinculando patrones composicionales con características como estabilidad estructural, eficiencia de traducción o especificidad tisular. La variabilidad en el contenido de GC, por ejemplo, puede influir en la termoestabilidad del ARN o en la adaptación evolutiva de organismos. Además, estos perfiles pueden revelar sesgos en la expresión génica o mutaciones asociadas a enfermedades, ofreciendo aplicaciones en biotecnología, medicina personalizada y estudios evolutivos.

Con el uso de "**grep**" se puede filtrar y contar patrones como "GC" y cuantificar su porcentaje, sin embargo al conectar comandos en cascada es recomendable manejar la salida estándar de error "**stderr**"

- **stderr** es un flujo de salida (file descriptor 2) dedicado a mensajes de error.
- A diferencia de stdout (Standard Output, file descriptor 1), que muestra la salida normal de un comando, **stderr** se usa para errores, advertencias o información de depuración.
- Por defecto, **stderr** se muestra en la terminal, pero puede redirigirse o suprimirse.

  Redirección de stderr

  a) Redirigir stderr a un archivo: (comando 2> errores.txt)
  b) Redirigir stderr a stdout (comando 2>&1)
  c) Redirigir stderr a /dev/null (comando 2>/dev/null)

1.9.2. Ejemplos prácticos de procesamiento masivo de datos transcriptómicos desde el terminal.

Desarrolle los cuadernos de trabajo 5 y 6

1.9.3. Integración con herramientas bioinformáticas como samtools para el análisis de alineamientos de secuencias de ARN.

La integración con herramientas bioinformáticas como Samtools es fundamental para el análisis de alineamientos de secuencias de ARN, ya que permite manipular, filtrar y visualizar

datos de manera eficiente. Samtools facilita la conversión de formatos (SAM a BAM), el ordenamiento e indexación de archivos, y la extracción de regiones específicas, lo que es crucial para estudios de expresión génica, identificación de variantes o análisis de splicing. Además, su capacidad para generar estadísticas de cobertura y profundidad complementa pipelines de análisis, integrando datos de secuenciación de ARN con otras herramientas como GATK, IGV o DESeq2, optimizando así el procesamiento y la interpretación de resultados.

Para instalar samtools ejecute:

```
sudo apt install samtools
```

Resumen

Hemos cubierto el uso de scripts en BASH en bioinformática, comenzando con conceptos básicos y avanzando hacia aplicaciones especializadas. En la primera parte, se cubre el manejo de archivos de secuencias (FASTA, FASTQ), incluyendo búsquedas, análisis y transformación de secuencias de ADN y ARN. Se introducen buenas prácticas en el desarrollo de scripts, como manejo de errores, comentarios y modularización, junto con la integración de herramientas bioinformáticas. Además, se abordan tareas avanzadas como la automatización de procesos repetitivos, el análisis de secuencias largas y la identificación de patrones, preparando al usuario para trabajar con grandes volúmenes de datos.

La segunda parte se enfoca en aplicaciones específicas para transcriptómica, incluyendo la conversión automatizada de ADN a ARN, el análisis de la composición de bases nitrogenadas y el procesamiento masivo de datos transcriptómicos. Se revisa superficialmente Samtools para el análisis de alineamientos de secuencias de ARN, como una hoja de ruta clara para dominar el uso de BASH en bioinformática, desde tareas básicas hasta aplicaciones avanzadas y especializadas.

PUCE Virtual

La excelencia no se improvisa

síguenos

MBC - 12 - 241