

Introducción a la Bioinformática

Instalación de Linux - BioPython - Blast Suite

Para poder desarrollar el trabajo práctico deberán tener instalados el sistema operativo Linux, el lenguaje de programación Python con las librerías BioPython y los programas de Blast.

Para disponer de Linux pueden realizar una partición del disco o bien instalar una máquina virtual, o trabajar directamente sobre el Workspace ITBA (escritorio Linux).

Las distribuciones Linux que suelen utilizarse en bioinformática son Debian o Ubuntu. Existen distribuciones como Bio-Linux cuyos paquetes pueden ser instalados sobre Debian o Ubuntu, o la máquina virtual de DNALinux, ambas ya tienen Python3 preinstalado, EMBOSS y otras herramientas que les serán de utilidad ya incluidas. Los programas Python son llamados scripts y tienen extensión *.py, es una lenguaje interpretado o de scripting (aunque también hay compiladores) cuya estructura deriva del C y toma cosas de la programación shell. BioPython es proyecto comunitario open source de módulos Python integrados para trabajar con secuencias y anotaciones, acceder a bases de datos remotas, parsear el output de programas como BLAST, FASTA, etc. Es prácticamente esencial para todo bioinformático.

Instalar BioPython de ser necesario a través del gesto PIP

BLAST local (BLAST+ is a new suite of BLAST tools that utilizes the NCBI) La descarga de los programas blast la hacen desde:

<ftp://ftp.ncbi.nlm.nih.gov/blast/executables/blast+/LATEST/>

Existen distintas maneras de correr Blast de manera local.

Se puede ejecutar con algún comando específico de BioPython o bien desde línea de comando:

```
> blastp -d swissprot -i demo.fasta -o myblast.report (con standalone Blast)
```

```
> blastall -p blastp -d swissprot -i demo.fasta -o myblast.report (con Blast+)
```

BLAST help: <http://www.ncbi.nlm.nih.gov/books/NBK52637/> 2

Introducción a la Bioinformática

Trabajo Práctico (final – parte 1)

El presente trabajo práctico tiene por objetivo adquirir las primeras habilidades en el campo de la Bioinformática. Se incluyen varios ejercicios donde deberán desarrollar pequeños scripts para resolver problemas específicos. Los mismos pueden ser desarrollados utilizando cualquiera de los lenguajes de programación bioinformática de código abierto como BioPerl, BioJava y BioRuby, que son ampliamente utilizados en la investigación bioinformática y de biología computacional, aunque se sugiere la utilización de BioPython para facilitar la resolución de los ejercicios. Las herramientas computacionales escritas en estos lenguajes proporcionan múltiples funcionalidades para crear soluciones personalizadas y realizar análisis de datos biológicos. Un ultimo ejercicio esta relacionado con la comprensión de la información en bases de datos de biología molecular. La ejecución de los scripts debe ser llevada a cabo a través de scripts en Bash, de modo que puedan automatizar el flujo de tareas, realizar control de errores, validación de formatos y logueo de tareas.

Para comenzar el trabajo deben entrar en la base de datos Online Mendelian Inheritance in Man (OMIM) donde encontrarán el catálogo online genes humanos asociados a trastornos genéticos más importante de la actualidad. En grupo decidan sobre que enfermedad que quieren investigar y luego a partir de la información en OMIM seleccionen uno más genes asociados a esta patología para comenzar con el ejercicio 1. Este mismo gen o genes deben utilizarse en el ejercicio 5.

Cada grupo tendrá 10 minutos para exponer como realizó el trabajo práctico y comentar sobre su investigación (y no sobre el código realizado). Por favor preparen una presentación. La correcta exposición del trabajo realizado por los miembros del grupo también entra en la evaluación.

Ejercicio 1 – PROCESAMIENTO DE SECUENCIAS. Escribir un script que lea una o más secuencias (de nucleótidos) de un archivo que contenga la información en formato GenBank de un mRNA de su gen (o genes) de interés, las traduzca a sus secuencias de amino ácidos posibles (tener en cuenta los Reading Frames) y escriba los resultados en un archivo en formato FASTA. Ustedes deben generarse su archivo GenBank de secuencias input, por ejemplo realizando una consulta de los mRNA del gen INS (que está asociado a la Diabetes) en la base de datos de NCBI-Gene y obtener uno o más resultados en formato GenBank en un archivo de texto. Si no desean seguir trabajando con las seis secuencias de aa posibles, pueden utilizar alguna función o programa que les permita saber cual el es marco de lectura correcto y seguir con esa secuencia.

NOTA: Ver aclaración de este ejercicio al final del documento.

– **Input:** Archivo de secuencias Genbank (ej. NMxxxx.gbk con una o más secuencias).

– **Output:** Archivo de secuencias Fasta de cada ORF (ej. Xxxxx.fas con una o más secuencias de aminoácidos).

Deben entregar el script Ex1.pm (si lo hacen con BioPerl, sino será otra extensión) y el input file que utilicen con una breve descripción de lo que hicieron y como se debe ejecutar para probarlo. 3

Ejercicio 2.a - BLAST. Escribir un script que realice un BLAST de una o varias secuencias (si son varias se realiza un Blast por cada secuencia input) y escriba el resultado (blast output) en un archivo. Nota: Pueden ejecutar BLAST de manera remota o bien localmente (si hacen ambos tienen más puntos!), para esto deben instalarse BLAST localmente del FTP del NCBI, luego bajarse la base de datos

<ftp://ftp.ncbi.nlm.nih.gov/blast/db/FASTA/swissprot.gz> y descomprimirla en un dir por ej. `ncbi-blast2.3.0+/data/`, luego usar el comando `ncbi-blast-2.3.0+/bin/makeblastdb` sobre el archivo `swissprot` (el original ya está en formato FASTA) para darle formato de BLAST DB. Dependiendo de la versión de Blast suite que tengan instalado puede que en vez de `makeblastdb` deban utilizar el comando `formatdb`.

- **Input: Secuencia Fasta (ej. `Xxxx.fas` con una o más secuencias de aminoácidos obtenidas en Ej.1).**
- **Output: Reporte Blast (ej. `blast.out`, si deciden hacer múltiples pueden generar un único o varios archivos). Deben entregar el script `Ex2.pm` y su input file con una breve descripción de lo que hicieron, con una interpretación de los resultados del Blast, y mencionar como se debe ejecutar para probarlo.**

Ejercicio 2.b – Interpretación del resultado del Blast. Dar una explicación del resultado blast obtenido en términos de las secuencias encontradas y dar una explicación sobre que significan los valores estadísticos asociados a las secuencias encontradas (el capítulo 4 del libro de David Mount puede ayudarlos).

Ejercicio 3 – Multiple Sequence Alignment (MSA). Descargarse las secuencias (en formato fasta) de los 10 mejores resultados Blast y realizar un alineamiento múltiple con la secuencia de consulta más estas 10 encontradas. Si no pueden hacerlo localmente pueden utilizar algún programa de MSA online. Intenten realizar una interpretación del resultado del alineamiento múltiple. Entregar información del MSA.

Aclaración para el Ejercicio 1

Para bajar una secuencia de nuestro gen elegido que funcione para en el Ejercicio 1 y para los demás, deberán bajarse alguno de los RNA mensajeros maduros (transcripto) de su gen de interés. Es decir, una secuencia de mRNA que ya haya sido procesada y no tenga intrones, esta es la secuencia que deben bajarse en formato Genbank y hacer la traducción a su secuencia de aminoácidos.

Por ejemplo, para el gen de la insulina humano (INS Homo sapiens):

1. Hacer una búsqueda en la base de datos de Genes e ir a las secuencias de Referencia y seleccionar algunos de los mRNA (NMxxxx)

NCBI Reference Sequences (RefSeq)

RefSeq maintained independently of Annotated Genomes

These reference sequences exist independently of genome builds. [Explain](#)

Genomic

NG_007114.1 RefSeqGene

Range 4968..8476

Download GenBank, FASTA, Sequence Viewer (Graphics)

mRNA and Protein(s)

NM_000207.2 - NP_000206.1 Insulin preproprotein

See identical proteins and their annotated locations for NP_000206.1

Status: REVIEWED

Description Transcript Variant: This variant (1) represents the shortest variant. All variants encode the same protein.

Source sequence(s) BC050205, BM010398

Consensus CDS CDS729.1

UniProtKB/TrEMBL I9AC29

UniProtKB/Swiss-Prot P31388

Conserved Domains (1) summary

cd054367 IGF_insulin_like_IGF_like_family_insulin_like_subgroup_specific_to_vertebrates. Members include a number of peptides including insulin and insulin-like growth factors I and II, which play a variety of roles in controlling processes such as metabolism, growth and ...

Location:26 -- 110

NM_001180907.1 - NP_001172026.1 Insulin preproprotein

See identical proteins and their annotated locations for NP_001172026.1

Status: REVIEWED

Description Transcript Variant: This variant (2) differs in the 5' UTR, compared to variant 1. All variants encode the same protein.

Source sequence(s) AY89504, BM010347, BF332143

Consensus CDS CDS729.1

UniProtKB/TrEMBL I9AC29

UniProtKB/Swiss-Prot P31388

Related ENSP0000029071, OTTHUMP0000011152, ENSG0000023071, OTTHUMT0000000394

Conserved Domains (1) summary

cd054367 IGF_insulin_like_IGF_like_family_insulin_like_subgroup_specific_to_vertebrates. Members include a number of peptides including insulin and insulin-like growth factors I and II, which play a variety of roles in controlling processes such as metabolism, growth and ...

Location:26 -- 110

NM_001180908.1 - NP_001172027.1 Insulin preproprotein

See identical proteins and their annotated locations for NP_001172027.1

Status: REVIEWED

Description Transcript Variant: This variant (3) differs in the 5' UTR, compared to variant 1. All variants encode the same protein.

Source sequence(s) AC13017, BM010367, BF332143

Consensus CDS CDS729.1

UniProtKB/TrEMBL I9AC29

UniProtKB/Swiss-Prot P31388

Related ENSP0000038043, OTTHUMP0000013813, ENSG0000037262, OTTHUMT0000004305

Conserved Domains (1) summary

cd054367 IGF_insulin_like_IGF_like_family_insulin_like_subgroup_specific_to_vertebrates. Members include a number of peptides including insulin and insulin-like growth factors I and II, which play a variety of roles in controlling processes such as metabolism, growth and ...

Location:26 -- 110

NM_001201887.1 - NP_001278026.1 Insulin preproprotein

See identical proteins and their annotated locations for NP_001278026.1

Status: REVIEWED

Description Transcript Variant: This variant (4) differs in the 5' UTR, compared to variant 1. All variants encode the same protein.

Source sequence(s) AC13017, BM010367

Consensus CDS CDS729.1

UniProtKB/TrEMBL I9AC29

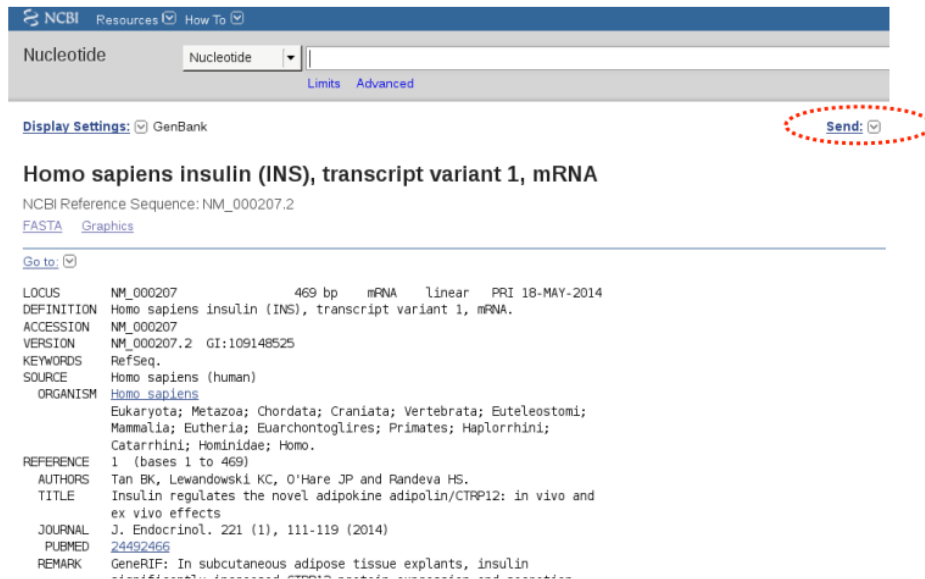
UniProtKB/Swiss-Prot P31388

Conserved Domains (1) summary

cd054367 IGF_insulin_like_IGF_like_family_insulin_like_subgroup_specific_to_vertebrates. Members include a number of peptides including insulin and insulin-like growth factors I and II, which play a variety of roles in controlling processes such as metabolism, growth and ...

Location:26 -- 110

2. Seleccionar uno de los transcritos del gen en formato GenBank (en lo posible la isoforma 1):



NCBI Resources How To

Nucleotide Nucleotide Limits Advanced

Display Settings: GenBank **Send:**

Homo sapiens insulin (INS), transcript variant 1, mRNA

NCBI Reference Sequence: NM_000207.2

[FASTA](#) [Graphics](#)

Go to:

LOCUS NM_000207 469 bp mRNA linear PRI 18-MAY-2014

DEFINITION Homo sapiens insulin (INS), transcript variant 1, mRNA.

ACCESSION NM_000207

VERSION NM_000207.2 GI:109148525

KEYWORDS RefSeq.

SOURCE Homo sapiens (human)

ORGANISM [Homo sapiens](#)

Eukaryota; Metazoa; Chordata; Craniata; Vertebrata; Euteleostomi; Mammalia; Eutheria; Euarchontoglires; Primates; Haplorrhini; Catarrhini; Hominidae; Homo.

REFERENCE 1 (bases 1 to 469)

AUTHORS Tan BK, Lewandowski KC, O'Hare JP and Rande HS.

TITLE Insulin regulates the novel adipokine adiponin/CTRP12: in vivo and ex vivo effects

JOURNAL J. Endocrinol. 221 (1), 111-119 (2014)

PUBMED [24492466](#)

REMARK GeneRIF: In subcutaneous adipose tissue explants, insulin

3. ORF (Open Reading Frame)

Una vez que tienen la secuencia bajada tengan en cuenta que ustedes desconocen cuál es el marco de lectura correcto de los 6 posibles. Por lo tanto deberán calcular los 6 marcos de lectura posibles, evaluar todos ellos en el ejercicio 2, y así darse cuenta cuál de los 6 es el real. Existen funciones en BioPerl para hacer esto, o mismo pueden usar el programa OrfFinder para ayudarse