Fecha límite de entrega: última clase de Octubre de 2017

Introducción a la Bioinformática
Instalación de Linux - BioPerl - Blast Suite

Para poder desarrollar el trabajo práctico deberán tener instalados el sistema operativo Linux (opcional), el lenguaje de programación Perl con las librerías BioPerl y los programas de Blast.

Linux
Pueden realizar una partición del disco o bien instalar una máquina virtual.
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 Perl preinstalado, EMBOSS y otras herramientas que les serán de utilidad ya incluidas.

Perl
Los programas Perl son llamados scripts y tienen extensión *.pl (o bien *.cgi si son aplicaciones web). Perl es una lenguaje interpretado o de scripting (aunque también hay compiladores) cuya estructura deriva del C y toma cosas de la programación shell. Instalación y documentación en: http://www.perl.org

BioPerl
BioPerl es proyecto comunitario open source de módulos Perl 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.
BioPerl-core tiene los módulos principales, BioPerl-run es una colección de módulos que facilitan la ejecución de programas locales como EMBOSS suite. Instalar BioPerl desde http://www.bioperl.org

BLAST local (BLAST+ is a new suite of BLAST tools that utilizes the NCBI)
La descarga de los programas blast la hacen desde:
https://blast.ncbi.nlm.nih.gov/Blast.cgi?CMD=Web&PAGE_TYPE=BlastDocs&DOC_TYPE=Download

Existen distintas maneras de correr Blast de manera local.
Se puede ejecutar con algún comando específico of bioperl 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+)
```

Desde BioPerl:
Utilizando el objeto Bio::Tools::Run::StandAloneBlast.pm (con Blast instalado local)
Utilizando el objeto Bio::Tools::Run::RemoteBlast.pm (usando el Blast del NCBI)

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

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 cuatro 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 BioPerl 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 quinto ejercicio esta relacionado con la comprensión de la información en bases de datos de biología molecular.

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.

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-blast-2.3.0+/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).

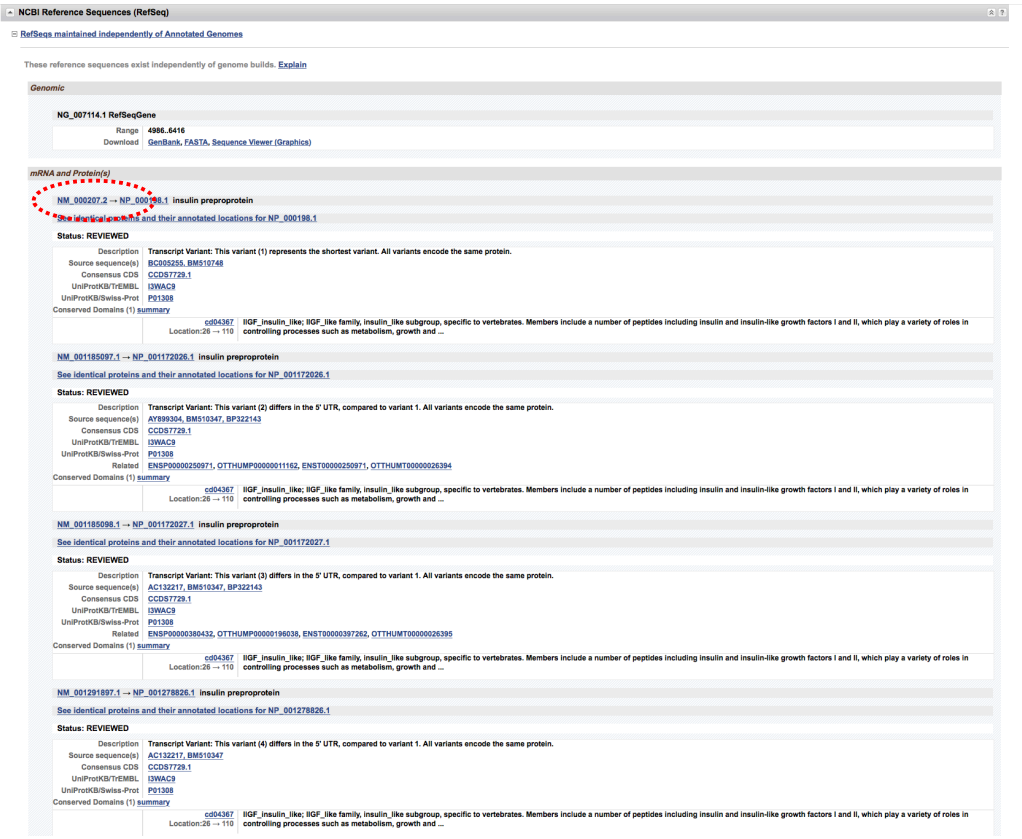
Ejercicio 3 – Multiple Sequence Alignment (MSA). Descargarse las secuencias (en formato fasta) de los 10 mejores resultados Blast y realizar un alineamiento múltiple con la secuencia de consulta más estas 10 encontradas. Si no pueden hacerlo localmente pueden utilizar algún programa de MSA online. Intenten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

Aclaración para el Ejercicio 1:

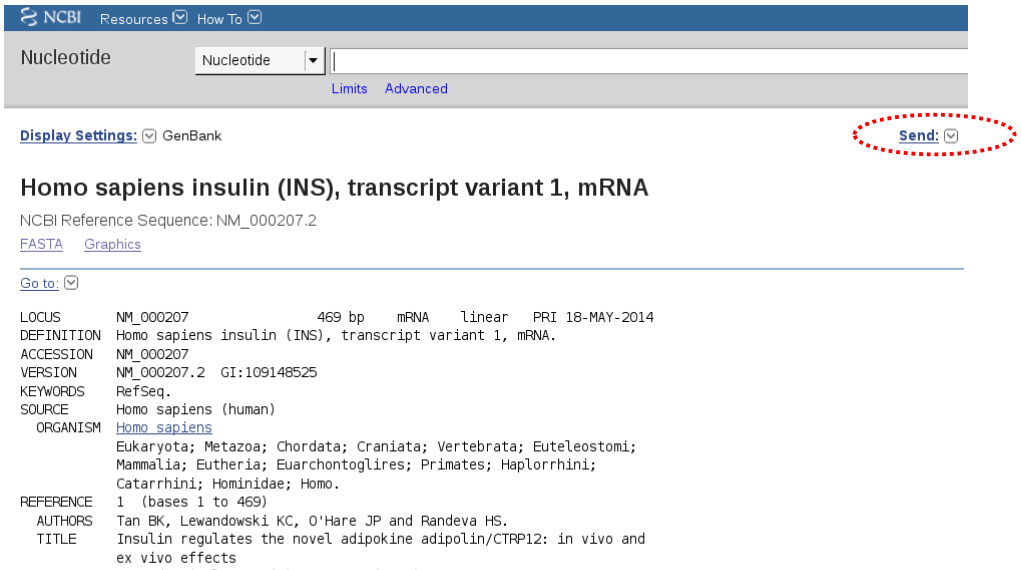
Para bajar una secuencia de nuestro gen elegido que funcione para en el Ejercicio 1 y para los demás, deberán bajarse alguno de los RNA mensajeros maduros (transcripto) de su gen de interés. Es decir, una secuencia de mRNA que ya haya sido procesada y no tenga intrones, esta es la secuencia que deben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



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



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.