

# Bitácora creada con comandos necesarios y útiles para los análisis bioinformáticos

**Elaborado por: Dra. Edith Elizondo Reyna**

Como parte de la estancia posdoctoral en el Departamento de Acuicultura bajo la dirección del Dr. Miguel Ángel del Río Portilla, CICESE, Ensenada. Período 2020-2021

## Ruta del directorio actual

```
In [1]: pwd
```

```
Out[1]: '/home/elizondo/jupyter'
```

## ubicar el directorio del archivo

```
In [ ]: cd
```

## listado de contenido del directorio actual

```
In [2]: ls
```

```
prueba_blastn.ipynb
```

## lista de los archivos del directorio actual

```
In [ ]: ls -lh
```

## Corroborar directorio de las bases de datos del blast

```
In [ ]: ls /LUSTRE/bioinformatica_data/BD/blast/db/NT
```

Describir el siguiente comando %%bash Ejecutar comando desde terminal  
export BLASTDB=/LUSTRE/bioinformatica\_data/BD/blast/db/NT Exportar la base de datos del blastn  
cd ~/data/chrysogaster/ Dirigirse al directorio donde se encuentra el archivo a analizar  
date > tiempo\_blastn.tx Nos muestra el tiempo que dura el analisis  
time /LUSTRE/apps/bioinformatica/ncbi-blast-2.6.0/bin/blastn Busca el programa de blast instalado en la computadora  
-query Ochrysogaster\_map\_contiglist2reads.fasta Busca el archivo a analizar  
-db /LUSTRE/bioinformatica\_data/BD/blast/db/NT/nt Busca la base de datos del blastn  
-out Ochrysogaster\_map\_contiglist2.tsv Nombra el nuevo archivo que contendra el analisis completo. Y el .tsv nos arroja un archivo separado por tabuladores.  
-evaluate 1E-6 \ Valor de algoritmo Best-Hits  
-max\_target\_seqs 1 \ Numero de coincidencias de secuencias homologas.  
-num\_threads 2 \ Numero de subprocesos utilizados en en CPU  
-outfmt "6 std sskingdoms stitle staxids sscinames scomnames sbblastnames strand" \  
Muestra las características de cada alineacion: Reino, Titulo por el blast, Taxon, Nombre científico, nombre comun, nombre del blast, longitud de secuencia.  
date >> tiempo\_blastn.tx Muestra el tiempo de lo que tardo el proceso.

## corroborar directorio Swiss Prot

```
In [ ]: ls /LUSTRE/bioinformatica_data/BD/SwissProt2/
```

## Cargar la paquetería correspondiente

```
from pandas import Series, DataFrame import pandas as pd from Bio import SeqIO, AlignIO, SeqRecord import matplotlib.pyplot as plt
```

## Comando que nos muestra el archivo a analizar y es equivalente a ls /home/elizondo/data/chrysogaster/

```
ls ~/data/ochrysogaster/
```

## Crear un nuevo directorio *mkdir*

```
mkdir ~/data/microalgas/cloroplasto/
```

## Nos muestra las primeras secuencias que se encuentran en el archivo a analizar

```
!head Ochrysogaster_map_contiglist2reads.fasta
```

## Comando que muestra el total de contigs en el archivo a analizar que es equivalente a !grep ">" T1\_Trinity.txt

```
!grep -c ">" ~/data/ochrysogaster/Ochrysogaster_map_contiglist2reads.fasta
```

## Se verifica el archivo de salida del Blast ejemplos

```
In [ ]: !head -2 Ochrysogaster_map_contiglist2reads_blastx.tsv  
        !tail -2 Ochrysogaster_map_contiglist2reads_blastx.tsv
```

## Se inicia el analisis de datos del blastx

Organizar los datos en tabla, el encabezado contiene la definición de cada columna de los resultados de blast. Se tomó la información del manual del Blast (<https://www.ncbi.nlm.nih.gov/books/NBK279690/>).

```
encabezado =("qseqid", "sseqid", "pident", "length", "mismatch", "gapopen", "qstart",  
             "qend", "sstart", "send", "eval", "bitscore", "sskingdoms", "stitle", "staxids",  
             "sscinames", "scomnames", "sblastnames")
```

encabezado =

"qseqid" Nombre de cada contig del archivo original  
"sseqid" Sujeto de la secuencia contig  
"pident" Porcentaje de coincidencias idénticas  
"length" Longitud de alineamiento  
"mismatch" Cantidad de nucleótidos que no coinciden en la secuencia  
"gapo pen" Número de aperturas o espacios  
"qstart" Inicio de la alineación de la secuencia de análisis (contigs)  
"qend" Final de la alineación de la secuencia de análisis (contigs)  
"sstart" Inicio de la identidad de alineación de la secuencia homóloga  
"send" Final de la identidad de alineación de la secuencia homóloga  
"eval" Valor de identidad de homología  
"bitscore" Puntuación de bit u homología  
"s kingdoms" Reino de la secuencia homóloga  
"stitle" Nombre de la proteína homóloga  
"staxids" Taxonomía de la secuencia homóloga separados por orden numérico  
"sscinames" Nombre científico de la secuencia homóloga  
"scomames" Nombre común de la secuencia homóloga  
"sblastnames" Nombre otorgado por blast a la secuencia homóloga

## Se inicia el análisis de datos del blastx

Organizar los datos en tabla, el encabezado contiene la definición de cada columna de los resultados de blast. Se tomó la información del manual del Blast (<https://www.ncbi.nlm.nih.gov/books/NBK279690/>).

```
encabezado =("qseqid", "sseqid", "pident", "length", "mismatch", "gapo pen", "qstart",  
"qend", "sstart", "send", "eval", "bitscore", "sskingdoms", "stitle", "staxids",  
"sscinames", "scomnames", "sblastnames")
```

encabezado =

```
"qseqid" Nombre de cada contig del archivo original  
"sseqid" Sujeto de la secuencia contig  
"pident" Porcentaje de coincidencias idénticas  
"length" Longitud de alineamiento  
"mismatch" Cantidad de nucleótidos que no coinciden en la secuencia  
"gapo pen" Número de aperturas o espacios  
"qstart" Inicio de la alineación de la secuencia de análisis (contigs)  
"qend" Final de la alineación de la secuencia de análisis (contigs)  
"sstart" Inicio de la identidad de alineación de la secuencia homóloga  
"send" Final de la identidad de alineación de la secuencia homóloga  
"eval" Valor de identidad de homología  
"bitscore" Puntuación de bit u homología  
"sskingdoms" Reino de la secuencia homóloga  
"stitle" Nombre de la proteína homóloga  
"staxids" Taxonomía de la secuencia homóloga separados por orden numérico  
"sscinames" Nombre científico de la secuencia homóloga  
"scomames" Nombre común de la secuencia homóloga  
"sblastnames" Nombre otorgado por blast a la secuencia homóloga
```

## Comando para buscar la tabla anterior con el encabezado

.....

```
In [ ]: encabezado =("qseqid", "sseqid", "pident", "length", "mismatch", "gapopen", "qstart", "qend", "sstart", "send", "evaluate", "bitscore", "sskingdoms", "stitle", "staxids", "sscinames", "scomnames", "sblastnames")
```

**Con el siguiente comando especifico que del archivo obtenido con el blast me cree una tabla con el encabezado anterior**

```
In [ ]: ftsv=pd.read_csv("Ochrysogaster_map_contiglist2reads_blastx.tsv", sep = "\t", header=None , names= encabezado, engine="python")
ftsv.head()
```

**Guardando los datos de la tabla creada con los comandos anteriores en formato csv**

```
In [ ]: ftsv.to_csv("Ochrysogaster_map_contiglist2reads_blastx.csv", header=True, index= None)
```

**Nos muestra en tabla los Reinos de la clasificacion de cada secuencia y el total**

Es necesario que se haya cargado la biblioteca DataFrame

```
ftab1= ftsv.groupby("sskingdoms")["qseqid"].count() ftab1 = DataFrame(ftab1) ftab1
```

**Nos despliega una tabla con los Reinos, el nombre que tiene en el Blast y la cantidad de cada clasificacion**

```
ftab2= ftsv.groupby(["sskingdoms", "sblastnames"])["qseqid"].count() ftab2 =  
DataFrame(ftab2) ftab2
```

## **Tabla del Reino Bacteria**

```
ftab2= ftsv.groupby(["sskingdoms"]).get_group('Bacteria') ftab2
```

## **Guardamos un archivo nuevo con el Reino Bacteria de la tabla anterior ftab2**

```
ftab2.to_csv("Ochrysogaster_blastx_bacterias.csv", header=True, index=None)
```

## **Obtener una grafica de barras de los sblastnames del archivo nuevo bacterias, con eje x frecuencia y eje y sblastnames**

poner ftsv. si así lo guardaste o ftab.



```
In [ ]: ftab2.plot(kind='barh', figsize=(8,6))
plt.axis([-1, int(max(ftab2)+5), -1, ftab2.count()], label=None)
plt.legend().set_visible(False)
plt.xlabel("Frecuencia")
plt.ylabel("sblastnames")
plt.title("Bacterias encontradas blastn")
yes = input("save figure? (y/) ")
if yes.lower()=="y":
    archivo = f[:f.find("Ochrysogaster_blastx_bacterias.csv")]+'bacterias.png'
    plt.savefig(archivo, dpi=400, bbox_inches='tight')
plt.show()
```

**Obtener una grafica de barras de los skingdoms del archivo del blastx, donde en el eje "x" se muestra la frecuencia y en el eje "y" skingdoms**

```
In [ ]: ftab1.plot(kind='barh', figsize=(8,6))
plt.axis([-1, int(max(ftab1)+5), -1, ftab1.count()], label=None)
plt.legend().set_visible(False)
plt.xlabel("Frecuencia")
plt.ylabel("skingdoms")
plt.title("Reinos blastx")
yes = input("save figure? (y/) ")
if yes.lower()=="y":
    archivo = f[:f.find("Ochrysogaster_map_contiglist2reads_blastx.csv")]+'bacterias.png'
    plt.savefig(archivo, dpi=400, bbox_inches='tight')
plt.show()
```

## Tabla del Reino Virus

```
In [ ]: ftab2= ftsv.groupby(["sskingdoms"]).get_group('Viruses')
        ftab2
```

**Comando que muestra todos los nombres del Blast, como su cantidad, exclusivos de Reino Eucariota de los datos analizados**

```
f_eucaria3= f_eucaria.groupby(["sblastnames"])["qseqid"].count()
f_eucaria3.sort_values(axis = 0, ascending=False, inplace=True)
```

**ftab3 = DataFrame(ftab3)**

f\_eucaria3

**Grafica de barras con los blastnames del Reino Eucariota**

```
In [ ]: f_eucaria3.plot(kind='barh', figsize= (8,6))
        plt.axis([-1, int(max(f_eucaria3)+5), -1, f_eucaria3.co
        unt()], label=None)
        plt.legend().set_visible(False)
        plt.xlabel("Frecuencia")
        plt.ylabel("sblastnames")
        plt.title("Eucariotas blastn")
        yes = input("save figure (y/? ) ")
        if yes.lower()=="y":
            archivo = f[:f.find("Ochrysogaster_eucariota.csv")]
            +'eucaria.png'
            plt.savefig(archivo, dpi=400, bbox_inches='tight')
        plt.show()
```

## Se carga la base de datos GO

```
In [ ]: fgo= pd.read_csv('/LUSTRE/bioinformatica_data/lga/bigdata/go_to_goslim.csv',engine="python")
        fgo.head(2)
```

```
In [ ]: f2=pd.merge(ftab, fspid, on="uniprotid", how='inner')
        f2.head(2)
```

```
In [ ]: f3=pd.merge(f2, fgo, on="GO_id", how='inner')
        f3.head()
```

## Guardar archivo nuevo .csv

```
In [ ]: f3.to_csv("Ochrysogaster_map_contiglist2reads_blastx.csv", index = None)
```

## Descomprimir archivo .gz (Gzip)

```
In [ ]: !gunzip 9_Synechococcus_s6assembly.fasta.gz
```

## Comando de corrida en blast dentro del Jupyter

```
In [ ]: %%bash
export BLASTDB=/LUSTRE/bioinformatica_data/BD/blast/db/NT/
cd ~/data/synechococcus/
date > tiempo_blastn.txt
time /LUSTRE/apps/bioinformatica/ncbi-blast-2.11.0/bin/blastn \
  -query 9_Synechococcus_s6assembly.fasta \
  -db /LUSTRE/bioinformatica_data/BD/blast/db/NT/nt \
  -out 9_Synechococcus_s6assembly_blastn.tsv \
  -evalue 1E-6 \
  -max_target_seqs 1 \
  -num_threads 24 \
  -outfmt "6 std sskingdoms stitle staxids sscinames sco
mnames sbblastnames strand"

date >> tiempo_blastn.txt
```

**se copian los archivos .tsv desde Lustre hasta mi caprteta tsv en mi direccion de omica**

```
In [ ]: %%bash
for f in ls */*.tsv
do
echo $f
cp $f ~/data/microalgas/tsv/
done
```

**El comando tar es usado para comprimir los archivos de interés que posteriormente serán descargados del directorio y así ser analizados en excel cada resultado de blastn. Y para empaquetar una compresión de alguna carpeta se debe usar el algo así como comprimir archivo/s o carpeta/s, se debe realizar de la siguiente manera:**

**!tar -czv**

```
In [ ]: !tar -czvf tsv.tar.gz ./tsv
```

## Comando de corrida de blastn en Slurum

```
In [ ]: fout = open("blastn_CN2.sh", "w")
linea=""#!/bin/sh

#
#SBATCH -p cicese
#SBATCH --job-name=blastn
#SBATCH -e blastn.%N.%j.err

#
#SBATCH -o blastn.%N.%j.log
#SBATCH -t 6-00:00:00
#
#SBATCH -N 1
#SBATCH -n 24
#
#SBATCH --exclusive

cd $SLURM_SUBMIT_DIR
#

shell=`/bin/basename \`/bin/ps -p $$ -ocomm=\`
if [ -f /usr/share/Modules/init/$shell ]
then
    . /usr/share/Modules/init/$shell
else
    . /usr/share/Modules/init/sh
fi

module load gcc-7.2
export LD_LIBRARY_PATH=/LUSTRE/apps/bioinformatica/ncbi
-blast-2.11.0/lib:$LD_LIBRARY_PATH
export BLASTDB=/LUSTRE/bioinformatica_data/BD/blast/db/
NT

#
cd /LUSTRE/bioinformatica_data/lga/edith/data/microalga
```

```
s/CN2
date > tiempoCN2_blastn.txt
time /LUSTRE/apps/bioinformatica/ncbi-blast-2.11.0/bin
/blastn \\\
  -query CN2.fasta \\\
  -db /LUSTRE/bioinformatica_data/BD/blast/db/NT/nt \\\
  -out CN2_blastn.tsv \\\
  -evaluate 1E-6 \\\
  -max_target_seqs 1 \\\
  -num_threads 24 \\\
  -outfmt "6 std sskingdoms stitle staxids sscinames sco
mnames sbblastnames strand"
date >> tiempoCN2_blastn.txt

head CN2_blastn.tsv
echo ""
grep -c CN2_blastn.tsv

""
fout.write(linea)
fout.close()
```