

Tutorial: Local BLAST.

Autor: A. Cumsille/R. Durán

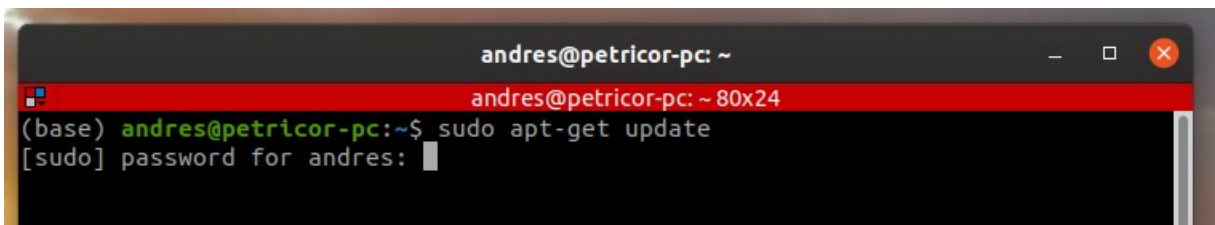
1. Instalar local BLAST

1.1 Vía Conda

Si tienes instalado Anaconda en Ubuntu, solo escribes en tu terminal `conda install -c bioconda blast`

1.2 Vía sudo

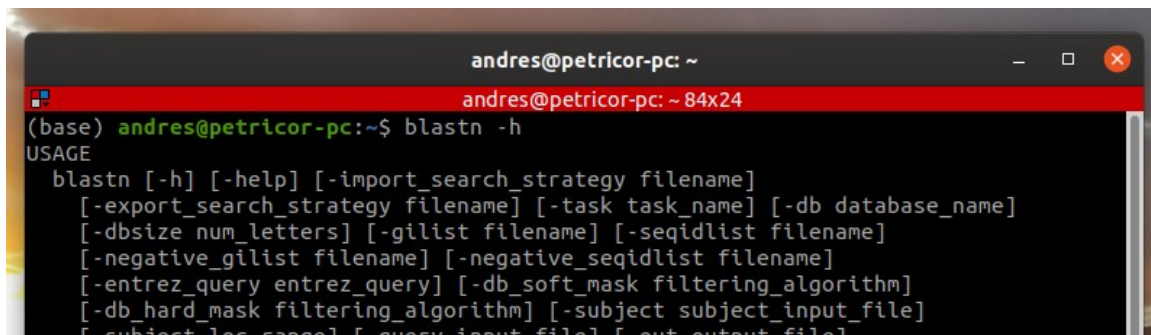
Si no tienes conda, entonces puedes hacerlo a través del comando `sudo` (comando super user do). Primero escribes `sudo apt-get update` para actualizar el repositorio de programas para instalar. Te va a pedir tu contraseña para esto.



```
andres@petricor-pc: ~
(base) andres@petricor-pc:~$ sudo apt-get update
[sudo] password for andres:
```

Luego escribes `sudo apt-get install ncbi-blast+`

Nota: En ambos casos para probar que la instalación funciona, escribes en tu terminal `blastn -h`. Este comando sirve para mostrar la ayuda de nucleotide BLAST. Si la instalación funcionó bien, entonces deberías ver algo similar a esto.



```
andres@petricor-pc: ~
(base) andres@petricor-pc:~$ blastn -h
USAGE
blastn [-h] [-help] [-import_search_strategy filename]
        [-export_search_strategy filename] [-task task_name] [-db database_name]
        [-dbsize num_letters] [-gilist filename] [-seqidlist filename]
        [-negative_gilist filename] [-negative_seqidlist filename]
        [-entrez_query entrez_query] [-db_soft_mask filtering_algorithm]
        [-db_hard_mask filtering_algorithm] [-subject subject_input_file]
        [-subject_loc range] [-query input_file] [-out output_file]
```

2. Vamos a crear una nueva carpeta y a copiar los archivos fasta que usamos en el tutorial anterior a ella. Para esto ingresaremos e iremos con el comando `cd` a la carpeta que contiene la carpeta `Bioinf_taller2` del tutorial anterior, en mi caso escribiendo `cd Documents/Bioinf/`

Ahí voy a crear una carpeta llamada `BLAST_tutorial` con el comando `mkdir BLAST_tutorial` y posteriormente voy a copiar todos los archivos fasta de la carpeta `Bioinf_taller2` a la carpeta `BLAST_tutorial` con el comando `cp Bioinf_taller2/*fasta BLAST_tutorial/`

Finalmente vamos a ingresar a esa carpeta y revisar su contenido. Si realizaste todo bien, deberías ver todos los archivos fasta ahí.

```
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial 87x24
(base) andres@petricor-pc:~$ cd Documents/Bioinf/
(base) andres@petricor-pc:~/Documents/Bioinf$ ls
Bioinf_taller2 Linux_Cheatsheet.jpg Tutorial_BLAST.odt
(base) andres@petricor-pc:~/Documents/Bioinf$ mkdir BLAST_tutorial
(base) andres@petricor-pc:~/Documents/Bioinf$ cp Bioinf_taller2/*fasta BLAST_tutorial/
(base) andres@petricor-pc:~/Documents/Bioinf$ cd BLAST_tutorial/
(base) andres@petricor-pc:~/Documents/Bioinf/BLAST_tutorial$ ls
Acinetobacter_baumannii_16S.fasta   Pseudomonas_stutzeri_16S.fasta
Acinetobacter_baumannii_23S.fasta   Pseudomonas_stutzeri_23S.fasta
Acinetobacter_junii_16S.fasta        Rhodococcus_erythropolis_16S.fasta
Concatenado_16S_actinos.fasta        Rhodococcus_erythropolis_23S.fasta
Concatenado_Abaumannii.fasta         Rhodococcus_ruber_16S.fasta
Concatenado_total.fasta              Rhodococcus_triadomae_16S.fasta
Nocardia_farcinica_16S.fasta          Streptomyces_albus_16S.fasta
Nocardia_farcinica_23S.fasta          Streptomyces_albus_23S.fasta
Nocardia_seriolae_16S.fasta           Streptomyces_venezuelae_16S.fasta
Nocardia_terpenica_16S.fasta          Streptomyces_venezuelae_23S.fasta
Pseudomonas_alcaligenes_16S.fasta     Streptomyces_violaceusniger_16S.fasta
(base) andres@petricor-pc:~/Documents/Bioinf/BLAST_tutorial$
```

3. Correr BLAST

BLASTN (query → nucleotide / db → nucleotide)

3.1 Crear una base de datos

Para poder realizar un análisis, local BLAST necesita una base de datos en donde realizar la búsqueda. Esta base de datos se puede crear con nuestros datos o datos recopilados desde repositorios en la web (GenBank, RefSeq, UniprotKB/Swissprot, UniprotKB/TrEMBLE, etc...). Siendo una gran utilidad cuando tenemos genomas que aún no han sido subidos a GenBank.

Para esto usaremos el comando `makeblastdb`. Veamos las opciones que entrega este comando

```
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial 87x24
(base) andres@petricor-pc:~/Documents/Bioinf/BLAST_tutorial$ makeblastdb -h
USAGE
makeblastdb [-h] [-help] [-in input_file] [-input_type type]
             -dbtype molecule_type [-title database_title] [-parse_seqids]
             [-hash_index] [-mask_data mask_data_files] [-mask_id mask_algo_ids]
             [-mask_desc mask_algo_descriptions] [-gi_mask]
             [-gi_mask_name gi_based_mask_names] [-out database_name]
             [-max_file_sz number_of_bytes] [-logfile File_Name] [-taxid TaxID]
             [-taxid_map TaxIDMapFile] [-version]
```

Nosotros crearemos nuestra base de datos utilizando el archivo `Concatenado_total.fasta`, que contiene todas las secuencias dentro de ese archivo

Para eso usaremos el comando

```
makeblastdb -in Concatenado_total.fasta -dbtype nucl -input_type fasta -out BlastDB
```

-in: El nombre del archivo con las secuencias que usaremos

-dbtype: el tipo de base de datos, en nuestro caso de nucleótidos

- input_type: el tipo de archivo, en nuestro caso es un archivo .fasta. Esta opción no era necesario agregarla en este caso, ya que si uno no agrega nada, el comando considera que es un archivo fasta por defecto.
- out: el nombre del archivo que será creado.

```
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial 122x28
(base) andres@petricor-pc:~/Documents/Bioinf/BLAST_tutorial$ makeblastdb -in Concatenado_total.fasta -dbtype nucl -input_type fasta -out BlastDB
Building a new DB, current time: 05/05/2020 17:53:24
New DB name: /home/andres/Documents/Bioinf/BLAST_tutorial/BlastDB
New DB title: Concatenado_total.fasta
Sequence type: Nucleotide
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA: added 21 sequences in 0.0812559 seconds.
```

Veámos con ls qué archivos se crearon.

```
(base) andres@petricor-pc:~/Documents/Bioinf/BLAST_tutorial$ ls
Acinetobacter_baumannii_16S.fasta  Nocardia_farcinica_16S.fasta  Rhodococcus_ruber_16S.fasta
Acinetobacter_baumannii_23S.fasta  Nocardia_farcinica_23S.fasta  Rhodococcus_triatoeae_16S.fasta
Acinetobacter_junii_16S.fasta       Nocardia_seriolae_16S.fasta   Streptomyces_albus_16S.fasta
BlastDB.nhr                         Nocardia_terpenica_16S.fasta  Streptomyces_albus_23S.fasta
BlastDB.nin                         Pseudomonas_alcaligenes_16S.fasta Streptomyces_venezuelae_16S.fasta
BlastDB.nsq                         Pseudomonas_stutzeri_16S.fasta Streptomyces_venezuelae_23S.fasta
Concatenado_16S_actinos.fasta       Pseudomonas_stutzeri_23S.fasta Streptomyces_violaceusniger_16S.fasta
Concatenado_Abaumanni.fasta         Rhodococcus_erythropolis_16S.fasta
Concatenado_total.fasta             Rhodococcus_erythropolis_23S.fasta
```

Ahora usemos BLAST con una de nuestras secuencias. En este caso voy a utilizar el 16S de *Streptomyces venezuelae* como “query”.

```
blastn -db BlastDB -query Streptomyces_venezuelae_16S.fasta -out S_venezuelae_16S_blast.txt
```

- db: el nombre de nuestra base de datos
- query: el nombre de nuestro archivo para realizar BLAST
- out: el archivo de salida

```
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial 129x28
(base) andres@petricor-pc:~/Documents/Bioinf/BLAST_tutorial$ blastn -db BlastDB -query Streptomyces_venezuelae_16S.fasta -out S_venezuelae_16S_blast.txt
```

Esto creara un archivo en la misma carpeta con el resultado del blast realizado. revisemos este archivo con less S_venezuelae_16S.txt

```
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial 113x35
Database: Concatenado_total.fasta
        21 sequences; 42,006 total letters

Query= Streptomyces_venezuelae_JCM4526
Length=1483

Sequences producing significant alignments:

          Score   E
          (Bits) Value
Streptomyces_venezuelae_JCM4526      2739   0.0
Streptomyces_violaceusniger_NBRC13459 2422   0.0
Streptomyces_albus_DSM40313          2392   0.0
Rhodococcus_triatoae_IMMIB_RIV-085   1945   0.0
Rhodococcus_ruber_DSM43338           1905   0.0
Nocardia_farcinica_N898              1903   0.0
Rhodococcus_erythropolis_N11         1895   0.0
Nocardia_seriolae_DSM44129           1862   0.0
Nocardia_terpenica_IFM0706           1794   0.0
Pseudomonas_stutzeri_ATCC17588        917   0.0
Acinetobacter_baumannii_DSM30007      917   0.0
Acinetobacter_junii_DSM6964           917   0.0
Acinetobacter_baumannii_DSM30007      917   0.0
Pseudomonas_alcaligenes_IAM12411      845   0.0

> Streptomyces_venezuelae_JCM4526
Length=1483

Score = 2739 bits (1483), Expect = 0.0
Identities = 1483/1483 (100%), Gaps = 0/1483 (0%)
Strand=Plus/Plus
```

Como podemos ver, y era de esperar, esta secuencia dio un 100% de identidad consigo misma y posteriormente va decreciendo el porcentaje de identidad a medida que es alineada con otras cepas.

3.2 Correr BLAST con su propia base de datos

Para correr BLAST utilizando la base de datos del NCBI se puede realizar con el siguiente comando

```
blastn -db nt -query Streptomyces_venezuelae_16S.fasta -out S_venezuelae_16S_remote_blast.txt -remote
```

-db: utiliza la base de datos de nucleótidos

-remote: es la opción que sirve para utilizar la base de datos remota de los servidores del NCBI.

```
andres@petricor-pc: ~/Documents/Bioinf/BLAST_tutorial 100x35
(base) andres@petricor-pc:~/Documents/Bioinf/BLAST_tutorial$ blastn -db nt -query Streptomyces_venezuelae_16S.fasta -out S_venezuelae_16S_remote_blast.txt -remote
```

Este comando toma un par de minutos en correr, pero el resultado lo podemos revisar utilizando less.

```

Database: Nucleotide collection (nt)
        58,021,211 sequences; 282,264,328,272 total letters

Query= Streptomyces_venezuelae_JCM4526
Length=1483

RID: B3UCHDVF014

Sequences producing significant alignments:

Score      E
(Bits)     Value
NR_024764.1 Streptomyces venezuelae strain JCM 4526 16S ribosoma... 2739    0.0
CP029195.1 Streptomyces venezuelae strain ATCC 10595 chromosome,... 2734    0.0
CP029196.1 Streptomyces venezuelae strain ATCC 21113 chromosome,... 2734    0.0
CP029197.1 Streptomyces venezuelae ATCC 10712 chromosome, comple... 2734    0.0
CP018074.1 Streptomyces venezuelae strain NRRL B-65442 genome 2734    0.0
MK053659.1 Streptomyces sp. adm13(2018) strain adm13 16S ribosom... 2728    0.0
KY362394.1 Streptomyces sp. strain RKND-444 16S ribosomal RNA ge... 2728    0.0
FR845719.1 Streptomyces venezuelae ATCC 10712 complete genome 2728    0.0
GU045539.1 Streptomyces sp. SXY124 16S ribosomal RNA gene, parti... 2728    0.0
CP046905.1 Streptomyces sp. QHH-9511 chromosome, complete genome 2723    0.0
HF935089.1 Streptomyces zaomyceticus partial 16S rRNA gene, stra... 2723    0.0
NR_044144.1 Streptomyces zaomyceticus strain NRRL B-2038 16S rib... 2723    0.0

```

La utilidad de BLAST local, es que puedes correr varios archivos a la vez y dejarlo trabajando mientras uno realiza otras actividades.

3.3 Realizar BLAST de multiples queries (un archivo query con varios fasta).

Usaremos para esto el archivo Concatenado_16S_actinos.fasta, que posee todos los rRNA16S de las actinobacterias en la carpeta, y para optimizar tiempo usaremos la base de datos que creamos.

```
blastn -db BlastDB -query Concatenado_16S_actinos.fasta -out Actinos_16S_blast.txt
```

Podemos revisar el archivo de salida utilizando less, donde podemos comprobar que está el BLAST de todas las cepas.


```

Database: Concatenado_total.fasta
      21 sequences; 42,006 total letters

Query= Nocardia_farcinica_N898
Length=1474

Sequences producing significant alignments:

```

	Score (Bits)	E Value
Nocardia_farcinica_N898	2723	0.0
Rhodococcus_triatae_IMMIB_RIV-085	2471	0.0
Rhodococcus_erythropolis_N11	2394	0.0
Nocardia_seriolae_DSM44129	2372	0.0
Nocardia_terpenica_IFM0706	2337	0.0
Rhodococcus_ruber_DSM43338	2320	0.0
Streptomyces_albus_DSM40313	1982	0.0
Streptomyces_violaceusniger_NBRC13459	1917	0.0
Streptomyces_venezuelae_JCM4526	1903	0.0
Pseudomonas_stutzeri_ATCC17588	872	0.0
Pseudomonas_alcaligenes_IAM12411	832	0.0
Acinetobacter_junii_DSM6964	822	0.0
Acinetobacter_baumannii_DSM30007	815	0.0

Si bajamos más en el archivo podemos ver el resto de las cepas

```

Query= Nocardia_seriolae_DSM44129
Length=1492

Sequences producing significant alignments:

```

	Score (Bits)	E Value
Nocardia_seriolae_DSM44129	2756	0.0
Rhodococcus_triatae_IMMIB_RIV-085	2429	0.0
Nocardia_terpenica_IFM0706	2374	0.0
Nocardia_farcinica_N898	2372	0.0
Rhodococcus_erythropolis_N11	2320	0.0
Rhodococcus_ruber_DSM43338	2235	0.0
Streptomyces_albus_DSM40313	1991	0.0
Streptomyces_violaceusniger_NBRC13459	1899	0.0
Streptomyces_venezuelae_JCM4526	1862	0.0
Pseudomonas_stutzeri_ATCC17588	905	0.0
Pseudomonas_alcaligenes_IAM12411	863	0.0
Acinetobacter_junii_DSM6964	833	0.0
Acinetobacter_baumannii_DSM30007	826	0.0
Acinetobacter_baumannii_DSM30007	826	0.0

BLASTP (query → aminoacid / db → aminoacid)

En este caso queremos buscar en una base de datos de aminoácidos una secuencia aminoacídica. Esto es común cuando estamos buscando secuencias homólogas en algún genoma de interés no publicado o queremos buscar algo en un grupo de bacterias de interés.

1. Crear base de datos: makeblastdb

Vamos a buscar un genoma de interés que quieran investigar. Para eso entrar a la base de datos Genome de GenBank (<https://www.ncbi.nlm.nih.gov/genome/?term=>)

Oceanococcus atlanticus
Representative genome: Oceanococcus atlanticus
 Download sequences in FASTA format for **genome**, **protein**
 Download genome annotation in **GFF**, **GenBank** or **tabular** format
 BLAST against Oceanococcus atlanticus **genome**, **protein**

NEW Try **NCBI Datasets** - a new way to download genome sequence and annotation we're testing in NCBI Labs

Display Settings: Overview Send to: ▼

Organism Overview ID: 5404E

Oceanococcus atlanticus

Lineage: Bacteria[27143]; Proteobacteria[8744]; Gammaproteobacteria[3295]; Chromatiales[128]; Ectothiorhodospiraceae[47]; Oceanococcus[1]; Oceanococcus atlanticus[1]

Summary

Submitter:	The Third Institute of State Oceanic Administration (SOA)
Assembly level:	Contig
Assembly:	GCA_002088235.1 ASM208823v1 scaffolds: 14 contigs: 14 N50: 859,326 L50: 2
BioProjects:	PRJNA196734
Whole Genome Shotgun (WGS):	RefSeq: NZ_AQQV000000000.1; INSDC: AQQV000000000.1
Statistics:	total length (Mb): 3.652 protein count: 3279 GC%: 60.6

Como necesitamos solo las secuencias aminoacídicas no necesitamos todos los archivos de anotación (necesitamos solo el de proteínas, `.fasta` amino acid o `.faa`).

Copien el archivo a la carpeta que están usando.

Para ver rápidamente la cantidad de secuencias de proteínas que posee este archivo pueden realizar el siguiente comando:

```
grep -c '>' *.faa
```

esto contará todos los `>` que encuentre en el archivo (esto sirve para buscar todo tipo de patrón de texto que quieran buscar).

```
makeblastdb -in *.faa -dbtype prot -input_type fasta -out AADB
```

2. Buscar tus secuencias query:

Para realizar una búsqueda debemos buscar alguna proteína que nos interese. Para esto la mejor idea es tener secuencias identificadas previamente que posean alguna evidencia experimental. UniprotKB es una buena base de datos para encontrar esto.

UniprotKB (<https://www.uniprot.org/>) posee dos bases de datos internas:

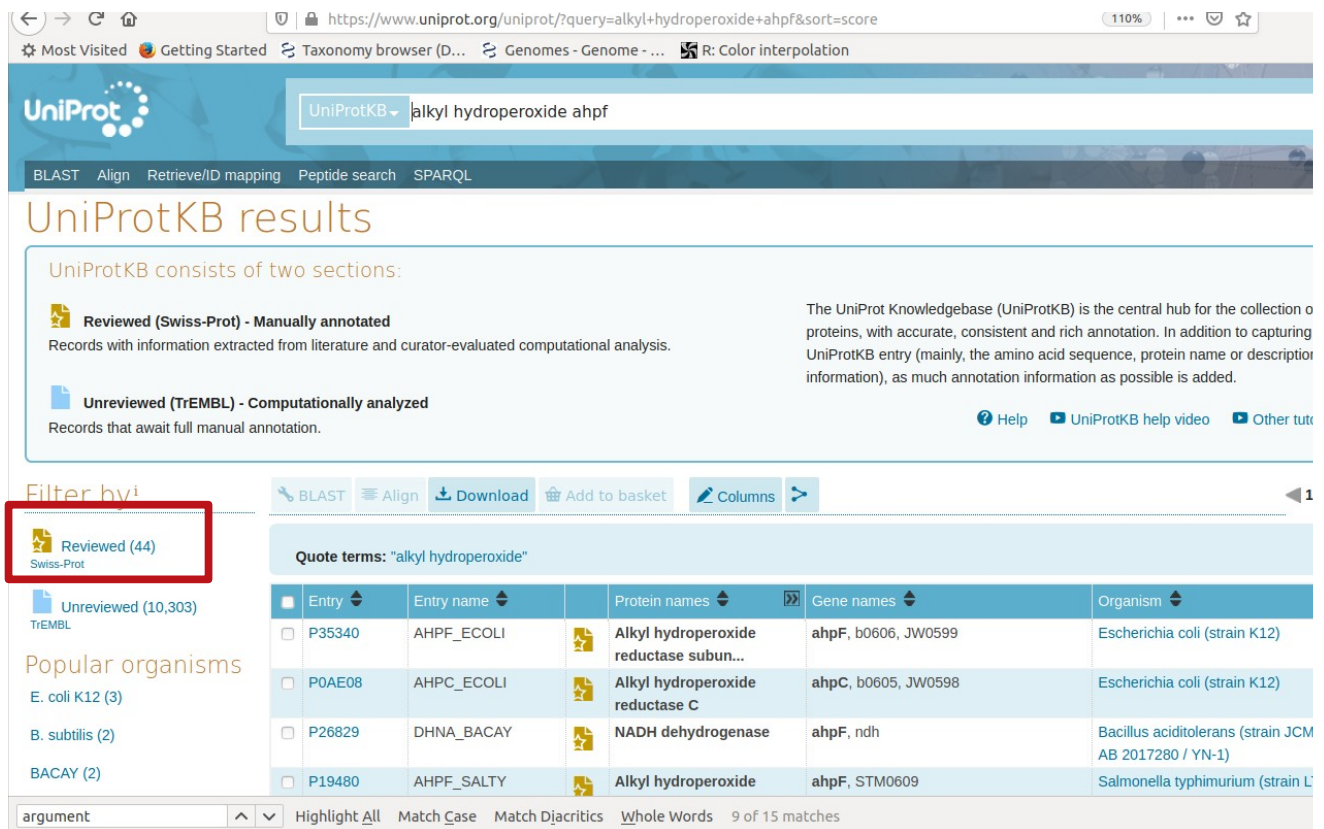
I. Swiss/Prot: base de datos curada a mano y validada por evidencia en publicaciones científicas (500.000 secuencias, 06/Mayo/2020).

II. TrEMBL: base de datos con todas las proteínas de todos los organismos reportados (hasta el momento más de 180 millones de secuencias; 06/Mayo/2020)

En una primera instancia es bueno utilizar solo proteínas que encuentren en bases de datos curadas (como Swiss-prot), si no encuentran homólogos en estas bases de datos pueden empezar a realizar otro tipo de búsquedas.

En mi caso buscaré una proteína relacionada a control de estrés oxidativo: alquil hidropéroxido reductasa (AhpF):

Búsqueda: **alkyl hydroperoxide reductase ahpF** y filtrar por la base de datos swiss-prot (a la izquierda)



UniProtKB results

UniProtKB consists of two sections:

- Reviewed (Swiss-Prot) - Manually annotated**
Records with information extracted from literature and curator-evaluated computational analysis.
- Unreviewed (TrEMBL) - Computationally analyzed**
Records that await full manual annotation.

The UniProt Knowledgebase (UniProtKB) is the central hub for the collection of proteins, with accurate, consistent and rich annotation. In addition to capturing UniProtKB entry (mainly, the amino acid sequence, protein name or description information), as much annotation information as possible is added.

Filter by:

- Reviewed (44)** (Swiss-Prot)
- Unreviewed (10,303) (TrEMBL)

Popular organisms:

- E. coli K12 (3)
- B. subtilis (2)
- BACAY (2)

Quote terms: "alkyl hydroperoxide"

Entry	Entry name	Protein names	Gene names	Organism
<input type="checkbox"/> P35340	AHPF_ECOLI	Alkyl hydroperoxide reductase subun...	ahpF, b0606, JW0599	Escherichia coli (strain K12)
<input type="checkbox"/> P0AE08	AHPC_ECOLI	Alkyl hydroperoxide reductase C	ahpC, b0605, JW0598	Escherichia coli (strain K12)
<input type="checkbox"/> P26829	DHNA_BACAY	NADH dehydrogenase	ahpF, ndh	Bacillus aciditolerans (strain JCM AB 2017280 / YN-1)
<input type="checkbox"/> P19480	AHPF_SALTY	Alkyl hydroperoxide	ahpF, STM0609	Salmonella typhimurium (strain L)

argument ^ v Highlight All Match Case Match Diacritics Whole Words 9 of 15 matches

Vamos a ordenar las por gene names, para que esten todas las ahpF juntas y seleccionaremos las secuencias que queremos (casilla blanca de al lado izquierdo).

Luego download y dejar todo igual, si quieren otro formato pueden cambiar la configuración.

https://www.uniprot.org/uniprot/?query=alkyl+hydroperoxide+ahpF+AND+reviewed%3Ayes&sort=genes

Records with information extracted from literature and curator-evaluated computational analysis.

Unreviewed (TrEMBL) - Computationally analyzed
Records that await full manual annotation.

Filter by¹

Reviewed (44)
B. subtilis (2)
E. coli K12 (2)
PSEPK (1)
STAAS (2)
XANCH (1)
Other organisms
Go

Popular organisms

Search terms

Filter "ahpF" as:
gene name (18)

Filter "hydroperoxide" as:
gene ontology (18)

Filter "alkyl" as:
argument

Quote terms: "alkyl"

Download selected (6)
Download all (44)

Format: FASTA (canonical)
Compressed Uncompressed
Preview first 10

Gene names	Organism	Length
6 result(s) selected (Clear Selection)		
ahpF, PP_2440	Pseudomonas putida (strain ATCC 47054 / DSM 6125 / NCIMB 11950 / KT2440)	520
ahpF	Xanthomonas campestris pv. phaseoli	530
ahpF, SAS0357	Staphylococcus aureus (strain MSSA476)	507
ahpF	Acinetobacter calcoaceticus	16
ahpF, SE_2358	Staphylococcus epidermidis (strain ATCC 12228)	507
ahpF, b0606, JW0599	Escherichia coli (strain K12)	521
ahpF, SERP0059	Staphylococcus epidermidis (strain ATCC 35984 / RP62A)	507
ahpF	Pseudomonas putida (Arthrobacter siderocapsulatus)	520
ahpF, SAR0398	Staphylococcus aureus (strain MRSA252)	507

argument Highlight All Match Case Match Diacritics Whole Words 9 of 15 matches

En mi caso abre una nueva ventana con todas las secuencias en fasta, por lo que seleccionan todo y lo copian a un archivo nuevo (nano query_aa.fa).

```

Activities Terminal
File Edit View Search Terminal Help
mié 14:41
roboto@roboto-UX310UQK: ~/BIOINF_tutorial

-r, --recursive like --directories=recurse
-R, --dereference-recursive likewise, but follow all symlinks
--include=FILE_PATTERN search only files that match FILE_PATTERN
--exclude=FILE_PATTERN skip files and directories matching FILE_PATTERN
--exclude-from=FILE skip files matching any file pattern from FILE
--exclude-dir=PATTERN directories that match PATTERN will be skipped.
-L, --files-without-match print only names of FILES with no selected lines
-l, --files-with-matches print only names of FILES with selected lines
-c, --count print only a count of selected lines per FILE
-T, --initial-tab make tabs line up (if needed)
-Z, --null print 0 byte after FILE name

Context control:
-B, --before-context=NUM print NUM lines of leading context
-A, --after-context=NUM print NUM lines of trailing context
-C, --context=NUM print NUM lines of output context
-NUM same as --context=NUM
--color[=WHEN], use markers to highlight the matching strings;
--colour[=WHEN] WHEN is 'always', 'never', or 'auto'
-U, --binary do not strip CR characters at EOL (MSDOS/Windows)

When FILE is '-', read standard input. With no FILE, read '-' if
recursive, '-' otherwise. With fewer than two FILES, assume '-'.
Exit status is 0 if any line is selected, 1 otherwise;
if any error occurs and -q is not given, the exit status is 2.

Report bugs to: bug-grep@gnu.org
GNU grep home page: <http://www.gnu.org/software/grep/>
General help using GNU software: <http://www.gnu.org/gethelp/>
roboto@roboto-UX310UQK:~/BIOINF_tutorial$ grep -c "GCF_002088235.1_ASM208823v1_protein.faa" 3278
roboto@roboto-UX310UQK:~/BIOINF_tutorial$ makeblastdb -in *.faa -dbtype prot -input_type fasta -out AADB
Building a new DB, current time: 05/06/2020 14:14:33
New DB name: /home/roboto/BIOINF_tutorial/AADB
New DB title: GCF_002088235.1_ASM208823v1_protein.faa
Sequence type: Protein
Keep MBits: T
Maximum file size: 1000000000B
Adding sequences from FASTA; added 3278 sequences in 0.119927 seconds.
roboto@roboto-UX310UQK:~/BIOINF_tutorial$ ls
AADB.phr AADB.pin AADB.psq GCF_002088235.1_ASM208823v1_protein.faa
roboto@roboto-UX310UQK:~/BIOINF_tutorial$ nano query_aa.fa

```

Todas estas secuencias serán su query.

3. Realizar el BLASTP

```
blastp -db AADB -query query_aa.fa -out query_proteome.txt
```

vamos a inspeccionar el archivo con *less*:

```
less query_proteome.txt
```

Si se dan cuenta es un archivo muy largo. Revisemos cuantas líneas tiene el archivo entregado, para saber más o menos cuán grande es el output.

```
wc -l query_proteome.txt
```

Si queremos ver rápidamente cuantos *hits* hay en nuestro proteoma podemos usar la herramienta *grep* que usamos antes

```
grep -c 'Score =' query_proteome.txt
```

Lo más probable es que hayan muchos homólogos en el proteoma que estoy buscando, o no?. Si revisamos el archivo con *less* nos daremos cuenta que hay muchos con un valor bajo de e-value.

```
File Edit View Search Terminal Help
roboto@roboto-UX310UQK: ~/BIOINF_tutorial
KT2440) OX=160488 GN=ahpF PE=3 SV=1
Length=528
Sequences producing significant alignments:
Score      E
(Blts)     Value
WP_083560717.1 alkyl hydroperoxide reductase subunit F [Oceanococcus atlanticus] 751 0.0
WP_083560962.1 thioredoxin-disulfide reductase [Oceanococcus atlanticus] 150 3e-42
WP_083559142.1 FAD-dependent oxidoreductase [Oceanococcus atlanticus] 58.5 4e-10
WP_083562513.1 dihydrolipoyl dehydrogenase [Oceanococcus atlanticus] 53.1 2e-08
WP_083560745.1 dihydrolipoyl dehydrogenase [Oceanococcus atlanticus] 51.2 8e-08
WP_083559775.1 NAD(P)-binding domain-containing protein [Oceanococcus atlanticus] 49.7 2e-07
WP_083563030.1 glutathione-disulfide reductase [Oceanococcus atlanticus] 48.9 5e-07
WP_083561090.1 NAD(P)/FAD-dependent oxidoreductase [Oceanococcus atlanticus] 47.0 2e-06
WP_083561969.1 NAD(P)/FAD-dependent oxidoreductase [Oceanococcus atlanticus] 47.0 2e-06
WP_146680254.1 FAD-dependent oxidoreductase [Oceanococcus atlanticus] 44.7 8e-06
WP_083562037.1 NAD(P)/FAD-dependent oxidoreductase [Oceanococcus atlanticus] 44.3 1e-05
WP_083558889.1 FAD-dependent oxidoreductase [Oceanococcus atlanticus] 37.7 0.001
WP_083560331.1 dihydrolipoyl dehydrogenase [Oceanococcus atlanticus] 35.4 0.007
WP_083561210.1 NAD(P)/FAD-dependent oxidoreductase [Oceanococcus atlanticus] 35.0 0.011
WP_083562423.1 5L-specific NAD(P)(+) transhydrogenase [Oceanococcus atlanticus] 33.1 0.038
WP_083560816.1 tRNA uridine-5-carboxymethylaminomethyl(34) synthetase [Oceanococcus atlanticus] 33.1 0.039
WP_146680117.1 mercury(II) reductase [Oceanococcus atlanticus] 33.1 0.042
WP_083561297.1 NAD(P)-binding domain-containing protein [Oceanococcus atlanticus] 31.2 0.15
WP_083561393.1 NAD(P)/FAD-dependent oxidoreductase [Oceanococcus atlanticus] 30.4 0.29
WP_083560734.1 electron transfer flavoprotein-ubiquinone oxidoreductase [Oceanococcus atlanticus] 30.0 0.36
WP_083563207.1 FAD-dependent oxidoreductase [Oceanococcus atlanticus] 29.6 0.52
WP_083562132.1 FAD-binding protein [Oceanococcus atlanticus] 28.5 1.1
WP_083561559.1 NADPH-dependent 2,4-dienoyl-CoA reductase [Oceanococcus atlanticus] 28.5 1.1
WP_083561077.1 choline dehydrogenase [Oceanococcus atlanticus] 28.1 1.6
WP_083559368.1 acetaldehyde dehydrogenase (acetylating) [Oceanococcus atlanticus] 27.7 1.8
WP_083562136.1 NAD(P)-binding domain-containing protein [Oceanococcus atlanticus] 27.3 2.1
WP_083562807.1 FAD-binding protein [Oceanococcus atlanticus] 27.3 2.2
WP_083561914.1 serine hydrolase [Oceanococcus atlanticus] 26.9 2.8
WP_083561391.1 Flavin monooxygenase family protein [Oceanococcus atlanticus] 26.9 3.5
WP_083561704.1 NAD(P)-binding protein [Oceanococcus atlanticus] 26.6 4.4
WP_083559089.1 leucyl aminopeptidase [Oceanococcus atlanticus] 26.2 5.9
WP_083559610.1 GODEF domain-containing protein [Oceanococcus atlanticus] 25.8 7.1
WP_158523182.1 hypothetical protein [Oceanococcus atlanticus] 25.8 7.2
WP_083559025.1 aromatic ring-hydroxylating dioxygenase subunit [Oceanococcus atlanticus] 25.8 8.6
WP_146680191.1 NAD(P)-binding domain-containing protein [Oceanococcus atlanticus] 25.4 8.6
WP_083560647.1 GMC family oxidoreductase [Oceanococcus atlanticus] 25.4 8.9
```

Vamos a refinar un poco la búsqueda.

```
blastp -db AADB -query query_aa.fa -out query_proteome2.txt -evalue 0.000000001
wc -l query_proteome2.txt
grep -c 'Score' query_proteome2.txt
```

Hay menos líneas en el segundo resultado? Revisen el archivo de output con *less*.

Por supuesto que esto es lo básico de BLAST, el manual de usuario lo pueden encontrar en <https://www.ncbi.nlm.nih.gov/books/NBK279690/>