Running command-line BLAST

The goal of this tutorial is to run you through a demonstration of the command line, which you may not have seen or used much before.

Start up an m1.xlarge Amazon EC2 instance.

All of the commands below can copy/pasted.

Install software

Copy and paste the following commands

sudo apt-get update && sudo apt-get -y install python ncbi-blast+

This updates the software list and installs the Python programming language and NCBI BLAST+.

Get Data

Grab some data to play with. Grab some cow and human RefSeq proteins:

wget ftp://ftp.ncbi.nih.gov/refseq/B\_taurus/mRNA\_Prot/cow.1.protein.faa.gz

wget ftp://ftp.ncbi.nih.gov/refseq/H\_sapiens/mRNA\_Prot/human.1.protein.faa.gz

This is only the first part of the human and cow protein files - there are 24 files total for human.

The database files are both gzipped, so lets unzip them

gunzip \*gz

ls

Take a look at the head of each file:

head cow.1.protein.faa

head human.1.protein.faa

These are protein sequences in FASTA format. FASTA format is something many of you have probably seen in one form or another – it’s pretty ubiquitous. It’s just a text file, containing records; each record starts with a line beginning with a ‘>’, and then contains one or more lines of sequence text.

Note that the files are in fasta format, even though they end if ”.faa” instead of the usual ”.fasta”. This NCBI’s way of denoting that this is a fasta file with amino acids instead of nucleotides.

How many sequences are in each one?

grep -c '^>' cow.1.protein.faa

grep -c '^>' human.1.protein.faa

This grep command uses the c flag, which reports a count of lines with match to the pattern. In this case, the pattern is a regular expression, meaning match only lines that begin with a >.

This is a bit too big, lets take a smaller set for practice. Lets take the first two sequences of the cow proteins, which we can see are on the first 6 lines

head -6 cow.1.protein.faa > cow.small.faa

BLAST

Now we can blast these two cow sequences against the set of human sequences. First, we need to tell blast about our database. BLAST needs to do some pre-work on the database file prior to searching. This helps to make the software work a lot faster. Because you installed your own version of the sotware, you need to tell the shell where the software is located. Use the full path and the makeblastdb command:

makeblastdb -in human.1.protein.faa -dbtype prot

ls

Note that this makes a lot of extra files, with the same name as the database plus new extensions (.pin, .psq, etc). To make blast work, these files, called index files, must be in the same directory as the fasta file.

Now we can run the blast job. We will use blastp, which is appropriate for protein to protein comparisons.

blastp -query cow.small.faa -db human.1.protein.faa

This gives us a lot of information on the terminal screen. But this is difficult to save and use later - Blast also gives the option of saving the text to a file.

blastp -query cow.small.faa -db human.1.protein.faa -out cow\_vs\_human\_blast\_results.txt

ls

Take a look at the results using less. Note that there can be more than one match between the query and the same subject. These are referred to as high-scoring segment pairs (HSPs).

less cow\_vs\_human\_blast\_results.txt

So how do you know about all the options, such as the flag to create an output file? Lets also take a look at the help pages. Unfortunately there are no man pages (those are usually reserved for shell commands, but some software authors will provide them as well), but there is a text help output

blastp -help

To scroll through slowly

blastp -help | less

To quit the less screen, press the q key.

Parameters of interest include the -evalue (Default is 10?!?) and the -outfmt

Lets filter for more statistically significant matches with a different output format:

blastp \

-query cow.small.faa \

-db human.1.protein.faa \

-out cow\_vs\_human\_blast\_results.tab \

-evalue 1e-5 \

-outfmt 7

I broke the long single command into many lines with by “escaping” the newline. That forward slash tells the command line “Wait, I’m not done yet!”. So it waits for the next line of the command before executing.

Check out the results with less.

Lets try a medium sized data set next

head -199 cow.1.protein.faa > cow.medium.faa

What size is this db?

grep -c '^>' cow.medium.faa

Lets run the blast again, but this time lets return only the best hit for each query.

blastp \

-query cow.medium.faa \

-db human.1.protein.faa \

-out cow\_vs\_human\_blast\_results.tab \

-evalue 1e-5 \

-outfmt 6 \

-max\_target\_seqs 1

Summary

Review:

* command line programs such as blast use flags to get information about how and what to do
* blast options can be found by typing *blastp -help*
* break a command up over many lines by using [`](https://angus.readthedocs.io/en/2016/running-command-line-blast.html#id1)` to “escape” the new line

**Reminder: shut down your instance!**

**LICENSE:** This documentation and all textual/graphic site content is licensed under the [Creative Commons - 0 License (CC0)](http://creativecommons.org/publicdomain/zero/1.0/) -- [fork @ github](https://github.com/ngs-docs/angus). Presentations (PPT/PDF) and PDFs are the property of their respective owners and are under the terms indicated within the presentation.

[comments powered by Disqus](http://disqus.com/)

* [index](https://angus.readthedocs.io/en/2016/genindex.html)
* [next](https://angus.readthedocs.io/en/2016/day2.html) |
* [previous](https://angus.readthedocs.io/en/2016/amazon/terminate-instance.html) |
* [angus 5.0 documentation](https://angus.readthedocs.io/en/2016/index.html) »

* [Day 1 - Getting started with Amazon](https://angus.readthedocs.io/en/2016/day1.html) »

© Copyright 2010 onwards, C. Titus Brown et al.. Created using [Sphinx](http://sphinx-doc.org/) 1.3.5.

  v: 2016