Programming for Bioinformatics
BIOL 8803 B
August 31st, 2015

Commands for today:

`wget` – get a remote file

`curl` – capture a URL, in this case a file

`ftp` – File transfer protocol interface

`scp` – Secure copy, copy a file to a different computer

`cut` – Get certain columns from a file

`paste` – Merge files together by columns

`join` – Join files by a common column

`touch` – change the attributes of a file

File Transfer Exercises

1.) Downloading UCSC Genome Browser files with `wget`

Toy exercise – the objective here is only to learn how to use wget. You need not to wait for each file to finish downloading, you can quit the download if you figured you are using the right command.

a.) What is an ENCODE repository? Download a file (pick a small one of your choice) from the ENCODE repository on the UCSC genome browser with `wget` and understand what data it contains

An ENCODE (Encyclopaedia of DNA Elements) repository is the storehouse of the analysis data of the functional elements of the human genome including elements that act at the protein and RNA levels conducted by various research groups around the world funded by the National Human Genome Research Institute.

Downloading a file from the ENCODE Genome Segmentation (Genome Segmentations from Dnase, FAIRE, Histone and TFBS Signals) :

wget
hgdownload.cse.ucsc.edu/goldenPath/hg19/encodeDCC/wgEncodeAwgSegmentation/wgEncodeAwgSegmentationChromhmmGm12878.bed.gz

This file has the results of chromosome segmentation on GM12878 human cell type obtained from the ENCODE Project.

b.) The database folder located in hg19 in goldenPath is an important folder. We are going to fetch four files `knownGene.txt.gz`, `knownGene.sql`, `kgXref.txt.gz`. and `kgXref.sql`. Copy their link address, paste in a file. Now get `wget` to download the files by reading the URL from the file you just created. In other words, this time the input URL comes from a file, and there are multiple of them. **Hold on to these files, we will use these later on.**

Input File 'wgetInput.txt' contains the following URLS:

ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/knownGene.txt.gz
ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/knownGene.sql
ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/kgXref.txt.gz
ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/kgXref.sql

Command for downloading the files using wget for the input file is :
wget -i wgetinput.txt

c.) Download all of the Pol2b binding data available in one line with `wget`. Hint: regex.

wget -r -nd -A "*Pol2b*" 'ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/'

-r means recursive retrieval of files

-nd means to download the files in the current folder of the machine

-A means to download only the files present in the folder mentioned that contain the pattern Pol2b

After execution of the command, it downloaded 24 files.

d) Test if the URL
http://www.ncbi.nlm.nih.gov/SNP/snp_ref.cgi?searchType=adhoc_search&type=rs&rs=rs12345 is correct using `wget`. Hint: Spiderman.

wget --spider
http://www.ncbi.nlm.nih.gov/SNP/snp_ref.cgi?searchType=adhoc_search&type=rs&rs=rs12345

Yes file exists.

2.) Downloading UCSC Genome Browser files with `curl`

a.) Repeat the last question's part a) but this time using `curl`

curl -o output.txt
hgdownload.cse.ucsc.edu/goldenPath/hg19/encodeDCC/wgEncodeAwgSegmentation/wgEncodeAwgSegmentationChromhmmGm12878.bed.gz

-o option is to store the output in the file output.txt. If it is not mentioned, then curl displays the output on the standard output.

b.) Download two files with `curl` (this time give the URL on the command line and not from a file)

The following command will download two files from the command line. -O option is used to indicate the file being saved in the current directory with the same name as in the remote server.

curl -O ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/HInv.sql -O ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/database/HInv.txt.gz

c.) What are the differences between `wget` and `curl`?

I) Without the -o or -O option, curl prints the output to the standard output, whereas wget will create an 'index.html.1' file in the current directory and store the output.

II) Curl provides APIs that can be used by programmers inside their own code. Wget is only a command-line tool without any API.

III) Curl supports a lot more protocols which wget doesn't support like SFTP, TFTP,TELNET,SCP,SMTP,IMAP.

IV) Wget supports recursive download whereas curl does not.

3.) Downloading lots of files with `ftp`

a.) Go to the UCSC Genome Browser download site (ftp mirror) http://hgdownload.cse.ucsc.edu/downloads.html. Look around

This contains links for downloading sequence and annotation data of genome assemblies provided in the UCSC Genome Browser of 69 vertebrates (complete annotation sets), 15 insects, 6 nematode species, 3 yeast species, ebola virus and some more few species.

b.) ftp into `hgdownload.cse.ucsc.edu` (user name is `anonymous`)

ftp hgdownload.cse.ucsc.edu

username: anonymous

password : my email-address

c.) Use ftp to download multiple files, perhaps the ENCODE GIS-PET RNA clusters for the hg18 assembly of the human genome?

To download the clusters, I first went to the folder.

cd goldenPath/hg18/encodeDCC/wgEncodeGisPet

mget *Clusters* - this will download all the GIS-PET cluster files.

4.) Write one-liners to perform the following actions with wget/curl:

a.) Download the file taxdb.tar.gz from ftp.ncbi.nih.gov/blast/db/

wget ftp://ftp.ncbi.nih.gov/blast/db/taxdb.tar.gz

b.) Download all files that start with "blast" from ftp.ncbi.nih.gov/blast/documents/

wget -nd -A "blast*" 'ftp://ftp.ncbi.nih.gov/blast/documents/'

c.) Download all "ppt" files from ftp.ncbi.nih.gov/blast/demo/

wget -r -nd -A ppt 'ftp://ftp.ncbi.nih.gov/blast/demo/'

5.) Use `wget`, `curl` and/or `ftp` to download files from the NCBI

a.) Download a Genbank sequence file of interest – e.g. NM_006565.3, using the eutils. Google and figure out how you can do this. It's easy.

wget -O nm6565.txt "http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nucleotide&id=NM_006565.3&rettype=gb&retmode=text"

This downloads the GenBank sequence for the accesion number NM_006565.3 where I used 'efetch' to retrieve the sequence, return type(gb) and return mode(text) for GenBank flat file.

b.) Download multiple sequences of interest using a list of accessions (3-10 accessions for your favorite genes or proteins), similarly to the previous question.

wget -O listSeq.txt http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nucleotide&id=NM_000019,NM_000017,NM_000015,NM_000006&rettype=gb&retmode=text

This downloads genes with accession numbers NM_000019, NM_000017, NM_000015 and NM_000006.

c)Lets `ftp` this time to the NCBI server. Figure out the address for the server (it's on NCBI). The login credentials are `anonymous` and password is your email address. Navigate to genbank/genomes/Bacteria/Neisseria_meningitidis_FAM18_uid255 and get me the gff and ptt files. What are these two formats?

ftp ftp.ncbi.nlm.nih.gov

Here I ftp into the NCBI server with the required credentials.

Navigated to the folder as :
cd /genbank/genomes/Bacteria/Neisseria_meningitidis_FAM18_uid255

Downloaded the required files as :
mget *.gff *.ptt

The meaning of the file formats is :
gff – genomic feature file
ptt – protein table fil

6.) E-utils

The documentation on how to use E-utils can be found here:
http://www.ncbi.nlm.nih.gov/books/NBK25500/)

Downloading a random genome sequence for *Neisseria meningitidis*. To do so, your approach will be the following:

a.) Retrieve the Genome ID for *N. meningitidis*

http://eutils.ncbi.nlm.nih.gov/entrez/eutils/esearch.fcgi?db=genome&term=Neisseria+meningitidis

It gives the genome ID of 172.

b.) Retrieve the Nucleotide ID linked to the Genome ID and limit the search to only RefSeq Genome Sequences.

http://eutils.ncbi.nlm.nih.gov/entrez/eutils/elink.fcgi?dbfrom=genome&db=nucleotide&id=172&term=srcdb+refseq[prop]&cmd=neighbor

This gives us many nucleotide IDs of RefSeq Genome sequences of Neisseria. One ID is 896408116.

c)Download the genome sequence.

http://eutils.ncbi.nlm.nih.gov/entrez/eutils/efetch.fcgi?db=nucleotide&id=896408116&rettype=fasta&retmode=text

Downloaded in the fasta format.

7.) More utilitarian usage with scp

a.) Copy a file to a remote server (such as biocluster or whatever you have access on)

scp has the format of : scp <source> <destination>. So the above task is accomplished as follws:

scp a.txt tsom3@thebeast.biology.gatech.edu

b.) Copy a directory

scp -r test tsom3@thebeast.biology.gatech.edu

-r is used to copy directories recursively.

c) Copy them both back to the machine you are on

scp -r tsom3@thebeast.biology.gatech.edu:/test/ .

scp -r tsom3@thebeast.biology.gatech.edu:a.txt .

8.) `touch`ing things around

a.) Create a test file by the name of efg.txt using `touch`

```
touch efg.txt
```

b.) Open it up in `emacs` and write something in it

emacs efg.txt

c)Save it and close it

ctrl-x ctrl-s

ctrl-x ctrl-c

d) `touch` the file again, did anything change?

Yes, the time of access changed to current time.

e.) List all the attributes of abc.txt (created in 2a) and pay attention to time and date of access.

ls -l abc.txt

-rw-rw-r-- 1 tannishtha tannishtha 40 Aug 23 00:56 abc.txt

Here we see that attributes of abc.txt file.

f.) `touch` abc.txt and list all the attributes again, what changes?

touch abc.txt

ls -l abc.txt

-rw-rw-r-- 1 tannishtha tannishtha 40 Sep 12 19:24 abc.txt

Now we see that the date and time got updated to the current date and time.

9.) Happy together!

a.) Create `1.txt` with the following content:

```
1 abc
2 lmn
3 pqr
```

b.) Create `2.txt` with the following content:

```
1 abc
3 lmn
9  opq
```

c)join the two files by **column 1**

join 1.txt 2.txt

1 abc abc

3 pqr lmn

d)join the two files by **column 2**

join -1 2 -2 2 1.txt 2.txt (-1 2 means 2nd column of first file and -2 2 means 2nd column of 2nd file, thus it means to join by 2nd column)

abc 1 1

lmn 2 3

e.) Repeat part (d) but this time also include all records from the **first** file. This is referred to as left outer join

join -a1 -1 2 -2 2 1.txt 2.txt ("-a1" includes records from first file).

abc 1 1

lmn 2 3

pqr 3

f.) Repeat part (d), this time also include all records from the **second** file. This is referred to as right outer join.

join -a2 -1 2 -2 2 1.txt 2.txt ("-a2" includes records from the second file)

abc 1 1

lmn 2 3

opq 9

g.) Repeat part (d) one last time and include all records from **both** files. This is referred to as full outer join

join -a1 -a2 -1 2 -2 2 1.txt 2.txt

abc 1 1

lmn 2 3

opq 9

pqr 3

10.) File handling/manipulation

a.) Generate two files by using the following commands

```
cat /dev/urandom | tr -dc 'ACGT' | fold -w 50 | head -50 > r1.fa
```

```
cat /dev/urandom | tr -dc 'acgt' | fold -w 50 | head -50 > r2.fa
```

b.) Display the first 5 characters in each of the first 5 lines of r1.fa

head -5 r1.fa | cut -c1-5

c)Combine (horizontally) the first 5 lines of r1.fa and r2.fa into a new file. That is, the resulting file's first 5 lines will be from r1.fa and the next 5 lines will be from r2.fa.

head -5 r1.fa | cat > output.txt

head -5 r2.fa | cat >> output.txt

d.) Combine (vertically) the last 5 lines of r1.fa and r2.fa into a new file. That is, the resulting file's first column will be last 5 lines of r1.fa and the second column will be last 5 lines will be from r2.fa.

paste r1.fa r2.fa | tail -5 > output1.txt

Here output is stored in output.txt file.