GEO Dataset Tools Documentation

Michael A. Gohde

November 7, 2017

1 Introduction

The GEO Dataset Tools provide a set of utilities and functions that should make it relatively straightforward to query the NIH's GEO database. Ultimately, the information gained from these tools should make it easier to download datasets and to collect information about how they were created.

Currently, the tools present here are in a state of fairly heavy development, so any discrepancies between that which is documented here and the actual operation of the tools is likely due to recent changes to how they were designed.

2 Querying the GEO Database for Entries

The first step in the process of finding specific datasets to work with is to actually determine which datasets exist. In the project directory, there exists a script named, 'query_geo.py'. Its purpose is to search the GEO dataset for various terms related to the protocols used for data collection and analysis. For example, it can be used to determine which entries in the GEO database were generated using GRO-Seq.

When the script is invoked, it queries the GEO database with the specified search term or, "gro-seq" if none is specified. If the query is successful, it will proceed to either print or write a file containing a set of database element IDs corresponding to elements matching the query. With this set of IDs, it will be possible to invoke another script to fetch all of the found elements from GEO. This process will be detailed in the next section.

2.1 Invoking query_geo.py

At the time of this writing, query_geo can be invoked in the following ways:

```
Usage: ./query_geo.py <args> query
Query the GEO database for numbers usable by fetch_groseq.py
If the -o switch is not specified, it is recommended that this program's output
be piped into either another command or a file.

<args> may be one of the following:
```

```
-h, --help      Prints this message.
-o=<filename>   Writes to an output file instead of stdout.
-v      Verbose output. All logging messages are written to stderr.

Examples:
./query_geo.py gro-seq >out.txt
./query_geo.py -o=out.txt pro-seq
```

Generally, the query parameter will correspond to a specific protocol, such as pro-seq, gro-seq, gro-cap, and 5'gro. In fact, the database fetching software detailed later in this document expects that this is the case. However, it should be possible to query the database for other features of interest, such as specific author names, species, etc.

It should also be noted that all this script works best when its output is redirected to a separate file. This makes it easier to view debugging information when the '-v' parameter is specified. Also, the other scripts in this project expect to be able to read a file instead of receiving GEO database IDs from the command line.

Below is an example session used to demonstrate a series of queries made to the GEO database:

```
user@computer ~/geo_dataset_tools $ ./query_geo.py -o=groseq.txt -v gro-seq
[Message] About to make query...
[Message] Query successful!
[Message] Found 1373 elements.
[Message] Done.

user@computer ~/geo_dataset_tools $ ./query_geo.py -o=proseq.txt -v pro-seq
[Message] About to make query...
[Message] Query successful!
[Message] Found 106 elements.
[Message] Done.

user@computer ~/geo_dataset_tools $ ./query_geo.py -o=5gro.txt "5'gro"
```

Note that the last command didn't print anything to the console. This is so because the '-v' parameter was not specified.

Contents of an the '5gro.txt' file generated above:

```
QUERY 5'gro
200068677
200090035
200083108
200045914
301678910
301678908
302396016
302396015
```

```
302193123
301119600
301119599
```

3 Generating a local metadata database

Once one or more sets of IDs has been obtained following the procedure outlined in the previous section, it's necessary to generate and maintain a local cache of information from the GEO database. This is done for a number of reasons:

1. It's significantly faster than repeatedly querying an online database.
2. Keeping a local cache reduces the load on the GEO database servers.
3. Having a local database following strict formatting rules will enable more applications to be developed in the future.
4. Additional data can be associated with each element, such as data series matrices, etc.
5. This enables consistency in implementation and usage.

In order to create this database, all that's necessary is to run the following command for all of the ID files fetched in the previous section:

```
user@computer ~/geo_dataset_tools $ ./make_groseq_database.py groseq.txt
    proseq.txt 5seq.txt grocap.txt db
```

In the above example, the local database creation tool will read through every ID defined in all of the files specified, then store appropriate metadata in the directory specified. It is possible to specify any number of input files (as long as there is at least one), and more queries can be integrated into the same database directory after it is populated for the first time. In fact, it should also be possible to merge the same query back into the database at a future date. Doing so will allow any changed elements to be added, while existing elements will be updated if changes have been made.

Full usage statement for the make_groseq_database script:

```
Usage: ./make_groseq_database.py idfile(s) dbdir
Fetches project summaries and adds appropriate metadata to a database of GEO
GRO-Seq data.
WARNING: This script may generate several thousand directories under dbdir.
```

4 Querying the local metadata database

Once enough metadata has been successfully fetched, it is possible to perform operations involving that metadata. In order to do so, it is necessary to invoke the 'query_groseq_database.py' script in the project directory.

The script's usage statement is listed below:

```
Usage: ./query_groseq_database.py [-s,-pt,-lq] dbdir command <args>
Query a GRO-Seq metadata database fetched with make_groseq_database.py
If -s is specified, then only series IDs will be reported on
If -pt=<comma separated list of protocols> is specified, then only IDs with a
     specific protocol will be reported on.
If -lq or --last-query is specified, then the program will attempt to read as
     arguments the results of the last query.

List of commands:
  listprotocols -- List all protocols in the current database.
  queryprotocol -- Print all elements matching a given protocol.
  protocoloverlap -- Print out the set of elements that overlap between different protoc
  listspecies -- Print a listing of all species defined in the database.
  findspecies <species name in quotes> -- Find all projects that match a given species.
  getsummary <list of id numbers or paper names> -- Retrieves a summary for a given data
  fetchmatrices <id> -- Downloads data matrices necessary to fetch data for a set.
  fetchspmats <species name> -- Fetches all matrices for a given species.
  fetchallmatrices -- Fetch as many matrix files as possible. This will be slow!
  getsralist <id or paper name> -- Retrieves all SRAs for a given element given that mat
  getreadytosra -- Retrieves a list of all projects with fetched matrix files.
  getreadytodownload -- Retrieves a list of all projects that can be downloaded immediat
  getbyyear <year> -- Retrieves a list of all projects with downloaded series matrices b
  getbycontrib <contributor name> -- Retrieves a list of all projects with downloaded se
  listyears -- List all years appearing in data matrices.
  listcontribs -- List all contributors appearing in data matrices.
       by the first contributor.
  download <id or paper name> <outputdir> -- Downloads data into the specified directory
```

4.1 An explanation of the '-s', '-pt' and '-lq' command line arguments

As documented, every command implemented in 'query_groseq_database' is designed to work with one of the following sets of elements:

1. Every element present in the database.
2. A specific set of elements explicitly listed by the user.
3. A specific element explicitly listed by the user.

The first case in particular is extremely limited in that it will often present quite a bit of unwanted data for every given query. For example, a query based on species name will include both collections of data and every entry for every data element in each collection.

The '-s' and '-pt' command line arguments reduce the size of the input set that query commands work on by allowing the user to specify specific additional attributes for what they would like to find. As such, the '-s' command limits all queries to just the set of defined

collections, rather than the elements making them up. The '-pt' command similarly allows users to search for data collected under specific protocols.

'-lq' works somewhat differently from '-s' and '-pt'. It is used to specify that the list of IDs or paper names normally required of certain commands like 'search' are instead to be found in either a special file created after every query or in a named file specified by the user. Instead of modifying the set of elements to be iterated over when doing a search, it appends the contents of the file to the arguments list passed to the program.

4.2 listprotocols

This command lists the set of protocols defined in the database. This is actually the list of queries fed into the make_groseq_database script. At the time of this writing, this command effectively just lists the contents of the database directory specified.

4.3 queryprotocol

This command fetches a list of all elements matching a given protocol. This list includes a set of ID numbers and human-readable titles provided by the GEO database.

4.4 protocoloverlap

This command causes query_groseq_database to attempt to find which elements are the same across protocols. Certain data series defined by GEO may have multiple protocols associated with them. Running protocoloverlap should make it easier to find out which datasets have which protocols associated with them. In the future, more commands like this one will be implemented.

4.5 listspecies

This command does as specified. It searches for and returns a list of all species defined in the dataset sorted by the frequency in which they were mentioned. At the time of this writing, it is noted that certain data series may specify multiple species. This situation still must be rectified.

4.6 findspecies

This command searches for all IDs matching the specified species. At the time of this writing, multiple species may be defined per data element. This situation will be resolved or otherwise noted through program behavior in the future.

4.7 getsummary

This command prints a detailed summary for every ID or paper name specified. Individual IDs are to be separated by spaces. Eventually, wildcard and similar match support should be implemented.

4.8 fetchmatrices

This command fetches a set of data series matrices for the appropriate ID if they are defined in that ID's metadata. Please note that this command requires that series matrices be defined for the given data element. While almost every element has an associated matrix, you can check for whether one can be fetched by checking for the "Data matrix URL defined?" field on running the getsummary command. If "YES" is printed next to the field, then matrices can be fetched. Otherwise, no data matrices have been defined for the current element and thus cannot be fetched.

One side effect of having fetched series matrices for a given element is that every future listing for that element by other commands will contain a "paper name." This is a string consisting of the last name of the first contributor to the element's project along with the date that the element was posted to GEO. Please note that publication years for papers and the dates at which their associated data was submitted to GEO may differ.

4.9 fetchspmats

This command was implemented to make it easier to fetch all data matrices for a given species. It is equivalent to running the 'findspecies' command, then running 'fetchmatrices' for every result returned.

4.10 fetchallmatrices

This command attempts to fetch every series matrix file associated with every element in the database. Please note that due to the intensive nature of this command, it may take a very long time to complete and lacks much formal validation.

4.11 getsralist

If data series matrices have been successfully fetched for a given element, then this command looks up the SRA ID numbers for every data file associated with that element.

4.12 getreadytosra

This command is a convenient way of listing every project for which series matrices have been fetched.

4.13 getreadytodownload

This command lists all elements on which 'getsralist' has been run.

4.14 download

This command downloads the set of SRA files found by the 'getsralist' command into the directory specified on the command line.

4.15 getbyyear

This command lists all elements posted to GEO in the specified year. Please note that the only way to determine the year of publication is to fetch the set of series matrix files associated with a given entry.

4.16 listyears

This command searches through all data elements for which series matrices have been downloaded to produce a set of publication years and their frequencies. This is useful when determining the set of publication years to pass to 'getbyyear'.

4.17 getbycontrib

This command works in much the same way as 'getbyyear', except it searches by the last name of the first contributor.

4.18 listcontribs

This command works in much the same way as 'listyears', except it produces a set of contributor names and their frequencies. Please note that it only considers the name of the first contributor for any given project, though it may, in the future, search through all contributors.

4.19 getaccession

This command searches through the database for a given GEO accession number or numbers and returns a list of all matching database IDs.

4.20 listsras

This command lists all of the SRA numbers associated with the IDs provided. It assumes that getsralist has been run for all of the IDs provided.

4.21 examples

This subsection shows a number of examples that should demonstrate how 'query_groseq_database' is used in practice. For all of the examples listed, the database is stored in a directory named 'db'.

The following is an example of a simple query listing all species in the database. Please note that the '-s' flag has been specified, so only GEO series entries are searched for.

```
user@computer-$ ./query_groseq_database.py -s db listspecies
List of available species:
125 Homo sapiens
62 Mus musculus
```

```
24 Drosophila melanogaster
10 Caenorhabditis elegans
4 Rattus norvegicus
3 Pan troglodytes
3 Macaca mulatta
2 Saccharomyces cerevisiae
2 Arabidopsis thaliana
1 Zea mays
1 Tetrahymena thermophila
1 Sus scrofa
1 Schizosaccharomyces pombe
1 Plasmodium falciparum
1 Canis lupus familiaris

Total number of elements: 241
```

The following is like the above query, but only for gro-seq data:

```
user@computer-$ ./query_groseq_database.py -s -pt="gro-seq" db listspecies
List of available species:
99 Homo sapiens
57 Mus musculus
18 Drosophila melanogaster
7 Caenorhabditis elegans
2 Rattus norvegicus
1 Zea mays
1 Tetrahymena thermophila
1 Sus scrofa
1 Saccharomyces cerevisiae
1 Plasmodium falciparum
1 Pan troglodytes
1 Macaca mulatta
1 Arabidopsis thaliana

Total number of elements: 191
```

The following is a listing of all entries for Rattus norvegicus. Note once again that all non-series entries have been filtered out:

```
user@computer-$ ./query_groseq_database.py -s db findspecies "Rattus norvegicus"
Found 4 elements that match "Rattus norvegicus" given protocol(s) 5'gro,pro-seq,gro-cap,
List of paths:

[pro-seq] 200085337: Natural Selection has Shaped Coding and Non-coding Transcription in

[pro-seq] "Mishmar2016" 200085747: Initiation of mtDNA transcription is followed by paus
```

```
[gro-seq] 200085747: Initiation of mtDNA transcription is followed by pausing, and diver

[gro-seq] 200058009: Required Enhancer: Matrin-3 Structure Interactions for Homeodomain
```

The above query generated a hidden file named '.lastquery'. This file contains the list of IDs found in a search query. Note that the set of IDs shown matches the set mentioned in the previous example:

```
200085337
200085747
200085747
200058009
```

This is an example using the '.lastquery' file listed before. It generates a listing of summaries for all of the elements specified within the file. It is also possible to specify additional paper names or element IDs for consideration:

```
user@computer-$ ./query_groseq_database.py -lq -s db getsummary
----Summary for 200085337----
Title: Natural Selection has Shaped Coding and Non-coding Transcription in Primate CD4+
Posted: 2016/12/29
Accession nr: GSE85337
Species: Pan troglodytes; Rattus norvegicus; Mus musculus; Macaca mulatta; Homo sapiens
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE85nnn/GSE85
Summary (begins on next line):
Transcriptional regulatory changes have been shown to contribute to phenotypic differenc

Data matrix URL defined? YES
Ready to fetch data? NO (run fetchmatrices 200085337)
Data fetched? NO
Protocol: pro-seq

----Summary for 200085747----
Title: Initiation of mtDNA transcription is followed by pausing, and diverge across huma
Posted: 2016/09/30
Accession nr: GSE85747
Species: Macaca mulatta; Homo sapiens; Caenorhabditis elegans; Drosophila melanogaster;
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE85nnn/GSE85
Summary (begins on next line):
We analyzed nascent mtDNA-encoded RNA transcripts from GRO-seq and PRO-seq experiments i

Data matrix URL defined? YES
Ready to fetch data? YES
Data fetched? NO
Protocol: pro-seq
```

```
----Summary for 200085747----
Title: Initiation of mtDNA transcription is followed by pausing, and diverge across huma
Posted: 2016/09/30
Accession nr: GSE85747
Species: Macaca mulatta; Homo sapiens; Caenorhabditis elegans; Drosophila melanogaster;
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE85nnn/GSE85
Summary (begins on next line):
We analyzed nascent mtDNA-encoded RNA transcripts from GRO-seq and PRO-seq experiments i

Data matrix URL defined? YES
Ready to fetch data? YES
Data fetched? NO
Protocol: pro-seq

----Summary for 200058009----
Title: Required Enhancer: Matrin-3 Structure Interactions for Homeodomain Transcription
Posted: 2014/08/03
Accession nr: GSE58009
Species: Rattus norvegicus
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE58nnn/GSE58
Summary (begins on next line):
Study of the POU-homeodomain transcription factor, has revealed that, binding of Pit1-oc

Data matrix URL defined? YES
Ready to fetch data? NO (run fetchmatrices 200058009)
Data fetched? NO
Protocol: gro-seq
```

The following is an example of a query for which the user wishes to store results in a specific file:

```
user@computer-$ ./query_groseq_database.py -qf=query.txt -s db getbyyear 2015
[gro-seq] "Yu2015" 200071369: Panoramix enforces piRNA-dependent co-transcriptional sile

[gro-seq] "Fuda2015" 200058955: GAGA Factor maintains promoters in nucleosome-free confo

[gro-seq] "Shilatifard2015" 200070408: PAF1, a molecular regulator of promoter-proximal

[gro-seq] "Fuda2015" 200058956: GAGA Factor maintains promoters in nucleosome-free confo

[gro-seq] "Fuda2015" 200058957: GAGA Factor maintains promoters in nucleosome-free confo
```

Once again, summaries are fetched given the query file specified:

```
user@computer-$ ./query_groseq_database.py -lq=query.txt -s db getsummary
```

```
----Summary for 200071369----
Title: Panoramix enforces piRNA-dependent co-transcriptional silencing (GRO-Seq)
Posted: 2015/10/15
Accession nr: GSE71369
Species: Drosophila melanogaster
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE71nnn/GSE71
Summary (begins on next line):
The Piwi-interacting RNA (piRNA) pathway is a small RNA-based innate immune system that

Data matrix URL defined? YES
Ready to fetch data? YES
Data fetched? NO
Protocol: gro-seq

----Summary for 200058955----
Title: GAGA Factor maintains promoters in nucleosome-free conformation and allows promot
Posted: 2015/04/04
Accession nr: GSE58955
Species: Drosophila melanogaster
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE58nnn/GSE58
Summary (begins on next line):
Promoter-proximal pausing of RNA polymerase II (Pol II) is a widespread in higher eukary

Data matrix URL defined? YES
Ready to fetch data? YES
Data fetched? NO
Protocol: gro-seq

----Summary for 200070408----
Title: PAF1, a molecular regulator of promoter-proximal pausing by RNA Polymerase II
Posted: 2015/08/13
Accession nr: GSE70408
Species: Homo sapiens; Drosophila melanogaster
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE70nnn/GSE70
Summary (begins on next line):
The control of promoter-proximal pausing and the release of RNA polymerase II (RNA Pol I

Data matrix URL defined? YES
Ready to fetch data? YES
Data fetched? NO
Protocol: gro-seq

----Summary for 200058956----
Title: GAGA Factor maintains promoters in nucleosome-free conformation and allows promot
Posted: 2015/04/04
```

```
Accession nr: GSE58956
Species: Drosophila melanogaster
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE58nnn/GSE58
Summary (begins on next line):
Promoter-proximal pausing of RNA polymerase II (Pol II) is a widespread in higher eukary

Data matrix URL defined? YES
Ready to fetch data? YES
Data fetched? NO
Protocol: gro-seq

----Summary for 200058957----
Title: GAGA Factor maintains promoters in nucleosome-free conformation and allows promot
Posted: 2015/04/04
Accession nr: GSE58957
Species: Drosophila melanogaster
Entry Type: GSEMatrix URL/FTP Link: ftp://ftp.ncbi.nlm.nih.gov/geo/series/GSE58nnn/GSE58
Summary (begins on next line):
This SuperSeries is composed of the SubSeries listed below.

Data matrix URL defined? YES
Ready to fetch data? YES
Data fetched? NO
Protocol: gro-seq
```

The following uses lastquery to fetch every pro-seq entry for Drosophila melanogaster:

```
user@computer-$ ./query_groseq_database.py -pt=pro-seq db findspecies "Drosophila melano
Found 5 elements that match "Drosophila melanogaster" given protocol(s) pro-seq
List of paths:
<cut>

user@computer-$ ./query_groseq_database.py -lq db fetchmatrices
Done with 200077607.
<cut>
Done with 200085747.
<cut>
Done with 200042397.
<cut>
Done with 200042117.
<cut>
Done with 200081649.
```