pysradb: A python package to query next-generation sequencing metadata and data from NCBI Sequence Read Archive

Authors

Computational Biology and Bioinformatics
University Of Southern California, Los Angeles, CA 900089, USA.

email:

Abstract
NCBI's Sequence Read Archive (SRA) is the primary archive of next-generation sequencing datasets. SRA makes metadata and raw sequencing data available to the research community to encourage reproducibility, and to provide avenues for testing novel hypotheses on publicly available data. However, methods to programmatically access this data are limited and not intuitive. We introduce a Python package `pysradb` that provides a collection of command line methods to query and download metadata and data from SRA. It utilizes the curated metadata database available through the SRAdb project. We demonstrate the utility of pysradb for different use cases.

Keywords
bioinformatics, SRA, NGS, NCBI, metadata, GEO, python

Introduction

Several projects have made efforts to analyze and publish summaries of DNA- sequencing [1] and RNA-sequencing [2, 3] datasets. Obtaining metadata and raw data from NCBI's Sequencing Read Archive (SRA) [4] is often the first step towards re-analyzing public next-generation sequencing datasets to compare them to private data or test a novel hypothesis. NCBI's SRA toolkit [5] provides utility methods to download raw sequencing data, while the metadata can be obtained by querying the website or through the Entrez command line utility [6]. There are few gaps in this workflow: a) determining which runs to download

In order to make querying both metadata and data more precise and robust, SRAdb [7] project provides a frequently updated SQLite database containing all the metadata parsed from SRA. `SRAdb` tracks the five main data objects in SRA's metadata: submission, study, sample, experiment and run. These are mapped to five different relational database tables that are made available in the SQLite file. SRAdb's metadata semantics remain as they are in SRA with minor changes in the field names to improve usability of SQL queries. The accompanying package made available in the R programming language, also called `SRAdb` [8], provides a convenient framework to handle metadata query and raw data downloads by utilizing the SQLite dabase. Though powerful, SRAdb's interface still requires the end user to be familiar with the R programming language. `pysradb` package builds up on the principles of `SRAdb` and provides a simple and intuitive interface for querying metadata and downloading datasets from SRA through command line utility. It obviates the need for the user to be familiar with any programming language as far as querying and downloading sequencing datasets is concerned. Additionally, it provides utility functions that will further help the user perform more granular queries, that are often required with datasets at large scale.

Methods

Implementation

`pysradb` is implemented in Python (Python Software Foundation, https://www.python.org/) [9] and uses `pandas` [10] for data frame based operations. Since, downloading datasets can often take long time, `pysradb` displays progress for long haul tasks using `tqdm` [11].The metadata information is read in the form of a SQLite [12] database made available by SRAdb [7]. `pysradb` also supports accessing metadata information from Gene Expression Omnibus (GEO) [13, 14] through the SQLite database made available through the GEOmetadb project [15].

`pysradb` can be run on either Linux or Mac based operating systems. It is implemented in Python programming language, has minimal dependencies and can be easily installed using either `pip` or `conda` based package manager visa the `bioconda` [16] channel. It works both in Python 2 and Python 3 environments.

Operation

`pysradb` uses SQLite file produced and made available by SRAdb [7] project. The file itself can be downloaded using `pysradb` as:

```
$ pysradb srametadb
Downloading SRAmetadb.sqlite.gz:  2.15GB [01:22, 28.0MB/s]
Extracting data/SRAmetadb.sqlite.gz ...
Extracting SRAmetadb.sqlite.gz:  33.0GB [07:51, 75.2MB/s]
Done!
Metadata associated with data/SRAmetadb.sqlite:
name value
0 schema version 1.0
1 creation timestamp 2018-12-07 00:39:29
```

SRAmetadb.sqlite file is required for all other operations supported by `pysradb`.

Use Cases

The primary use case of `pysradb` is in automated download of an entire SRA project. NCBI's sratoolkit [5] allows downloading fetching raw data per sequencing run. Each SRA project (SRP) consists of one or multiple experiments (SRX) which are sequenced as one or multiple runs (SRR). `sratoolkit` allows downloading

Getting a list of GEO experiments for a GEO study

Any GEO study (GSE) will involve a collection of experiments (GSM). We can obtain an entire list of experiments corresponding to the study using the `gse-to-gsm` subcommand from `pysradb`:

```
$ pysradb gse-to-gsm GSE41637 | head
```

```
study_alias    experiment_alias
 GSE41637       GSM1020640_1
 GSE41637       GSM1020641_1
 GSE41637       GSM1020642_1
 GSE41637       GSM1020643_1
 GSE41637       GSM1020644_1
 GSE41637       GSM1020645_1
 GSE41637       GSM1020646_1
 GSE41637       GSM1020647_1
 GSE41637       GSM1020648_1
```

However, just a list of GSM id is not useful if one is performing any downstream analysis which essentially requires more detailed information about the metadata associated with each experiment. This relevant metadata associated with each sample can be obtained by provinding `gse-to-gsm` additional flags:

```
$ pysradb gse-to-gsm -desc GSE41637 | head
```

```
study_alias    experiment_alias    sample_attribute
GSE41637       GSM1020640_1        source_name: mouse_brain || strain: DBA/2J || tissue: brain
GSE41637       GSM1020641_1        source_name: mouse_colon || strain: DBA/2J || tissue: colon
GSE41637       GSM1020642_1        source_name: mouse_heart || strain: DBA/2J || tissue: heart
GSE41637       GSM1020643_1        source_name: mouse_kidney || strain: DBA/2J || tissue: kidney
GSE41637       GSM1020644_1        source_name: mouse_liver || strain: DBA/2J || tissue: liver
GSE41637       GSM1020645_1        source_name: mouse_lung || strain: DBA/2J || tissue: lung
GSE41637       GSM1020646_1        source_name: mouse_skm || strain: DBA/2J || tissue: skeletal muscle
GSE41637       GSM1020647_1        source_name: mouse_spleen || strain: DBA/2J || tissue: spleen
GSE41637       GSM1020648_1        source_name: mouse_testes || strain: DBA/2J || tissue: testes
```

The metadata information can then be parsed from the `sample_attribute` column. To obtain a more strucutred metadata, we can use an additional flag `-expand`:

```
$ pysradb gse-to-gsm -desc -expand GSE41637 | head
```

```
study_alias    experiment_alias    source_name     strain    tissue
GSE41637       GSM1020640_1        mouse_brain     dba/2j    brain
GSE41637       GSM1020641_1        mouse_colon     dba/2j    colon
GSE41637       GSM1020642_1        mouse_heart     dba/2j    heart
GSE41637       GSM1020643_1        mouse_kidney    dba/2j    kidney
GSE41637       GSM1020644_1        mouse_liver     dba/2j    liver
GSE41637       GSM1020645_1        mouse_lung      dba/2j    lung
GSE41637       GSM1020646_1        mouse_skm       dba/2j    skeletal muscle
```

Getting SRP from GSE

```
$ pysradb gse-to-srp -desc -expand GSE100007
```

study_alias	study_accession	cell_type	fraction	molecule
GSE100007	SRP109126	human embryonic stem cells (hesc)	cytoplasm	cytoplasmic rna
GSE100007	SRP109126	human embryonic stem cells (hesc)	high polysome (5-8+ ribosomes)	polysomal rna
GSE100007	SRP109126	human embryonic stem cells (hesc)	low polysome (2-4 ribosomes)	polysomal rna
GSE100007	SRP109126	human embryonic stem cells (hesc)	monosome (80s)	monosomal rna
GSE100007	SRP109126	human embryonic stem cells (hesc)	nucleus	nuclear rna
GSE100007	SRP109126	human embryonic stem cells (hesc)	ribosome protected footprints	ribosome protected foo
GSE100007	SRP109126	neural cultures (14 days)	cytoplasm	cytoplasmic rna
GSE100007	SRP109126	neural cultures (14 days)	high polysome (5-8+ ribosomes)	polysomal rna
GSE100007	SRP109126	neural cultures (14 days)	low polysome (2-4 ribosomes)	polysomal rna
GSE100007	SRP109126	neural cultures (14 days)	monosome (80s)	monosomal rna
GSE100007	SRP109126	neural cultures (14 days)	nucleus	nuclear rna
GSE100007	SRP109126	neural cultures (14 days)	ribosome protected footprints	ribosome protected foo
GSE100007	SRP109126	neural cultures (50 days)	cytoplasm	cytoplasmic rna
GSE100007	SRP109126	neural cultures (50 days)	high polysome (5-8+ ribosomes)	polysomal rna
GSE100007	SRP109126	neural cultures (50 days)	low polysome (2-4 ribosomes)	polysomal rna
GSE100007	SRP109126	neural cultures (50 days)	monosome (80s)	monosomal rna
GSE100007	SRP109126	neural cultures (50 days)	nucleus	nuclear rna
GSE100007	SRP109126	neural progenitor cells (npc)	cytoplasm	cytoplasmic rna
GSE100007	SRP109126	neural progenitor cells (npc)	high polysome (5-8+ ribosomes)	polysomal rna
GSE100007	SRP109126	neural progenitor cells (npc)	low polysome (2-4 ribosomes)	polysomal rna
GSE100007	SRP109126	neural progenitor cells (npc)	monosome (80s)	monosomal rna
GSE100007	SRP109126	neural progenitor cells (npc)	nucleus	nuclear rna
GSE100007	SRP109126	neural progenitor cells (npc)	ribosome protected footprints	ribosome protected foo

Getting SRPs from GSE

```
$ pysradb gse-to-srp GSE24355 GSE25842
```

study_alias	study_accession
GSE24355	SRP003870
GSE25842	SRP005378

Getting project's metadata

```
$ pysradb metadata SRP026005 -assay -desc -expand | head
```

study_accession	experiment_accession	sample_accession	run_accession	library_strategy	embryonic_stage	source_nam
SRP026005	SRX305245	SRS444476	SRR900108	RNA-Seq	e9.5	neural cres
SRP026005	SRX305246	SRS444467	SRR900109	RNA-Seq	e9.5	neural cres
SRP026005	SRX305247	SRS444468	SRR900110	RNA-Seq	e9.5	neural cres
SRP026005	SRX305247	SRS444468	SRR900111	RNA-Seq	e9.5	neural cres
SRP026005	SRX305248	SRS444470	SRR900112	RNA-Seq	e9.5	neural cres
SRP026005	SRX305249	SRS444471	SRR900113	RNA-Seq	e9.5	neural cres
SRP026005	SRX305250	SRS444472	SRR900114	RNA-Seq	e9.5	neural cres
SRP026005	SRX305250	SRS444472	SRR900115	RNA-Seq	e9.5	neural cres
SRP026005	SRX305251	SRS444473	SRR900116	RNA-Seq	e9.5	neural cres

Getting assay summary statistics

```
$ pysradb metadata SRP000941 -assay | tr -s ' ' | cut -f5 -d ' ' | tail -n +2 | sort
| uniq -c
```

999	Bisulfite-Seq
768	ChIP-Seq
121	OTHER
353	RNA-Seq
28	WGS

Downloading data

```
$ pysradb download -p SRP003870 -p SRP005378
```

The simplest use case of 'pysradb' is when you apriopri know the SRA project ID (SRP) and would simply want to fetch the metadata associated with it. This is generally reflected in the 'SraRunTable.txt' that you get from NCBI's website. Example: https://www.ncbi.nlm.nih.gov/Traces/study/?acc=SRP098789.

```
pysradb metadata SRP098789
```

Once you have fetched the metadata and made sure, this is the project you were looking for, you would want to download everything at once. NCBI follows this hiererachy: 'SRP => SRX => SRR'. Each 'SRP' (project) has multiple 'SRX' (experiments) and each 'SRX' in turn has multiple 'SRR' (runs) inside it. We want to mimick this hiereachy in our downloads. The reason to do that is simple: in most cases you care about 'SRX' the most, and would want to "merge" your SRRs in one way or the other. Having this hierearchy ensures your downstream code can handle such cases easily, without worrying about which runs (SRR) need to be merged.

```
$ pysradb download -p SRP063852
```

Often, you need to process only a smaller set of samples from a project (SRP). Consider the project SRP000941 which has data spanning four assays. But, we might be only interested in analyzing the 'RNA-seq' samples and would just want to download that subset. This can be done simply using the following command:

```
$ pysradb metadata SRP000941 -assay | grep 'study|RNA-Seq' | pysradb download
```

Search

Another common operation that we do on SRA is search, plain text search. If we want to look up for all projects where 'ribosome profiling' appears somewhere in the description, we can use:

```
$ pysradb search '"ribosome profiling"'
```

Data availability

Please add details of where any datasets that are mentioned in the paper, and that have not have not previously been formally published, can be found. If previously published datasets are mentioned, these should be cited in the references, as per usual scholarly conventions.

Software availability

Software and source code available from: https://github.com/saketkc/pysradb
Documentation available at: https://saketkc.github.io/pysradb
Archived source code at time of publication:
Software license: BSD-3-Clause

Author Contributions

Competing interests

No competing interests were disclosed.

Grant information

The author declared that no grants were involved in supporting this work.

Acknowledgments

References

[1] Daniel G MacArthur, Suganthi Balasubramanian, Adam Frankish, Ni Huang, James Morris, Klaudia Walter, Luke Jostins, Lukas Habegger, Joseph K Pickrell, Stephen B Montgomery, et al. A systematic survey of loss-of-function variants in human protein-coding genes. *Science*, 335(6070):823–828, 2012.

[2] Alexander Lachmann, Denis Torre, Alexandra B Keenan, Kathleen M Jagodnik, Hoyjin J Lee, Lily Wang, Moshe C Silverstein, and Avi Ma'ayan. Massive mining of publicly available rna-seq data from human and mouse. *Nature communications*, 9(1):1366, 2018.

[3] Leonardo Collado-Torres, Abhinav Nellore, Kai Kammers, Shannon E Ellis, Margaret A Taub, Kasper D Hansen, Andrew E Jaffe, Ben Langmead, and Jeffrey T Leek. Reproducible rna-seq analysis using recount2. *Nature biotechnology*, 35(4):319, 2017.

[4] Rasko Leinonen, Hideaki Sugawara, Martin Shumway, and International Nucleotide Sequence Database Collaboration. The sequence read archive. *Nucleic acids research*, 39(suppl_1):D19–D21, 2010.

[5] SRA Toolkit Development Team. Sra toolkit. `https://ncbi.github.io/sra-tools/`. [Online; accessed 10-December-2018].

[6] Jonathan Kans. Entrez direct: E-utilities on the unix command line. 2018.

[7] Yuelin Zhu, Robert M Stephens, Paul S Meltzer, and Sean R Davis. Sradb: query and use public next-generation sequencing data from within r. *BMC bioinformatics*, 14(1):19, 2013.

[8] Jack Zhu and Sean Davis. Bioconductor:sradb, December 2018. URL `https://doi.org/10.18129/B9.bioc.SRAdb`.

[9] Guido van Rossum and Fred L. Drake. *The Python Language Reference Manual*. Network Theory Ltd., 2011. ISBN 1906966141, 9781906966140.

[10] Wes McKinney. Data structures for statistical computing in python. In Stéfan van der Walt and Jarrod Millman, editors, *Proceedings of the 9th Python in Science Conference*, pages 51 – 56, 2010.

[11] Casper da Costa-Luis, Stephen L., Hadrien Mary, noamraph, Mikhail Korobov, Ivan Ivanov, Marcel Bargull, James, Guangshuo Chen, Matthew D. Pagel, Staffan Malmgren, Socialery, Jack McCracken, Fabian Dill, Daniel Panteleit, Alex Rothberg, Yaroslav Halchenko, Tomas Ostasevicius, Shirish Pokharel, ReadmeCritic, Peter VandeHaar, Kuang che Wu, jcea, Hugo, Ford Hurley, Edward Betts, David Bau, Arun Persaud, Alexander, and Adnan Umer. tqdm/tqdm: tqdm v4.20.0 stable, April 2018. URL `https://doi.org/10.5281/zenodo.1211527`.

[12] Sqlite home page. `https://sqlite.org/`. [Online; accessed 10-December-2018].

[13] Ron Edgar, Michael Domrachev, and Alex E Lash. Gene expression omnibus: Ncbi gene expression and hybridization array data repository. *Nucleic acids research*, 30(1):207–210, 2002.

[14] Tanya Barrett, Stephen E Wilhite, Pierre Ledoux, Carlos Evangelista, Irene F Kim, Maxim Tomashevsky, Kimberly A Marshall, Katherine H Phillippy, Patti M Sherman, Michelle Holko, et al. Ncbi geo: archive for functional genomics data sets—update. *Nucleic acids research*, 41(D1):D991–D995, 2012.

[15] Yuelin Zhu, Sean Davis, Robert Stephens, Paul S Meltzer, and Yidong Chen. Geometadb: powerful alternative search engine for the gene expression omnibus. *Bioinformatics*, 24(23):2798–2800, 2008.

[16] Björn Grüning, Ryan Dale, Andreas Sjödin, Brad A Chapman, Jillian Rowe, Christopher H Tomkins-Tinch, Renan Valieris, Johannes Köster, and Team Bioconda. Bioconda: sustainable and comprehensive software distribution for the life sciences. *Nature methods*, 15(7):475, 2018.