genomepy: Genes and Genomes at your fingertips

This manuscript (permalink) was automatically generated from vanheeringen-lab/genomepy_manuscript@d8b0f8f on July 8, 2022.

Authors

- **Siebren Frölich**
  0000-0001-6925-8446 · siebrenf
  Department of Molecular Developmental Biology, Radboud University

- **Maarten van der Sande**
  0000-0001-7803-1526 · Maarten-vd-Sande · MaartenvdSande
  Department of Molecular Developmental Biology, Radboud University

- **Simon van Heeringen**
  0000-0002-0411-3219 · simonvh · svheeringen
  Department of Molecular Developmental Biology, Radboud University

Abstract

Analyzing functional genomics data, including ATAC-, ChIP- and RNA-sequencing, requires genomic data such as a genome assembly and gene annotations. These resources can generally be retrieved from multiple organizations, at multiple versions, and multiple processing methods. Most bioinformatic workflows require the user to supply this genomic data manually, which can be a tedious and error-prone process.

Here we present genomepy, which can search, download, and preprocess the right genomic data for your analysis. Genomepy can search genomic data on GENCODE, Ensembl, UCSC and NCBI, and compare available gene annotations, to enable an informed decision. The selected genome and gene annotation can be downloaded (from anywhere) and preprocessed with sensible, yet controllable, defaults. Additional supporting data can be automatically generated, such as aligner indexes, genome metadata and blacklists. These functionalities are available on command line interface, aimed at ease of use and integration in automated pipelines, with extended functionality on the Python application programming interface.

Introduction

Data analysis is increasingly important in biological research. Whether you are analyzing gene expression in two samples or protein binding motifs in genomic atlases, you will need external information such as a reference genome or a gene annotation. For these types of data, there are three major providers: Ensembl [1], UCSC [2] and NCBI [3], and many model-system specific providers, such as GENCODE [4], ZFIN [5], FlyBase [6], WormBase [7], Xenbase [8] and more. Providers have different approaches to compiling genome assemblies and gene annotations, which effect formats, format compliance, naming, data quality, available versions and release cycle. These differences significantly impact compatibility with research [9], tools and (data based on) other genomic data.

You could try to find genomic data yourself, but there are many options with no clear metric for the “best” one. Ensembl, UCSC and NCBI each have FTP archives, web portals, and REST APIs, which you can use to search their individual databases. Alternatively, there are several tools that can be used to access some of these databases programmatically, such as ncbi-genome-download [10] and ucsc-genomes-downloader [11]. However, none of these can search, compare or download from all major genome providers data. Furthermore, downloading and processing genomic data manually can be tedious, error-prone, and poorly reproducible. Although the latter could be remedied by a data management tool, such as iGenomes [12], refGenie [13] or Go Get Data [14], data managers still require the user to add new data manually.

We have developed genomepy to 1) find genomic data on major providers, 2) compare gene annotations, 3) select the genomic data best suited to your analysis and 4) provide a suite of functions to peruse and manipulate the data. Selected data can be downloaded from anywhere, and is processed automatically. To ensure reproducibility, data sources and processing steps are documented, and can be enhanced further by using a data manager. Genomic data can be loaded into genomepy, which utilizes and extends on packages including pyfaidx [15], pandas [16] and MyGene.info [17] to rapidly work with gene and genome sequences and metadata. Similarly, genomepy has been incorporated into other packages, such as pybedtools [18] and CellOracle [19]. Genomepy can be used on command line and though the (fully documented) Python API, for a one-time analysis or integration in pipelines and workflow managers.

Features of genomepy

The key features of genomepy are 1) providing an overview of available assemblies with the `search` function, 2) download and processing of a selected assembly, with the `install` function and 3) utilizing assembly data using the Python API.

The `search` function queries the databases of GENCODE, Ensembl, UCSC and NCBI (caching the metadata), for text, taxonomy identifiers or assembly accession identifiers. The input type is automatically recognized and used to find assemblies that have the text in the genome names or various description fields, matches the taxonomy identifier or (partially) matches the assembly accession. The output of the `search` function is a table with rows of metadata for each assembly found. The metadata contains the assembly name and accession, taxonomy identifier, and indicates whether a gene annotation can be downloaded (or which of the four UCSC annotations) (see fig. 1a). Snippets of available gene annotation(s) can be inspected with the `annotation` function (fig. 1b).

```
$ genomepy search GRCh38
name           provider   accession          tax_id  annotation   species
        other_info
                                                       n r e k     <- UCSC
        options (see help)
GRCh38         GENCODE    GCA_000001405.15     9606       ✓        Homo sapiens
        GENCODE annotation + UCSC genome
GRCh38.p13     Ensembl    GCA_000001405.28     9606       ✓        Homo sapiens
        2014-01-Ensembl/2021-08
hg38           UCSC       GCA_000001405.15     9606    ✓ ✓ ✗ ✓     Homo sapiens
        Dec. 2013 (GRCh38/hg38)
GRCh38         NCBI       GCF_000001405.26     9606       ✓        Homo sapiens
        Genome Reference Consortium
 ^
 Use name for genomepy install
```

```
$ genomepy annotation GRCh38.p13
12:00:00 | INFO | Ensembl
1       ensembl_havana  gene    1211340 1214153 .       -       .
        gene_id "ENSG00000186827"; gene_version "11"; gene_name "TNFRSF4";
        gene_source "ensembl_havana"; gene_biotype "protein_coding";
12:00:00 | INFO | NCBI
NC_000001.11    genomepy        transcript      11874   14409   .       +
        .       gene_id "DDX11L1"; transcript_id "NR_046018.2";  gene_name
        "DDX11L1";
```

```
$ genomepy install --annotation GRCh38.p13
$ ls -1 ~/.local/share/genomes/GRCh38.p13
GRCh38.p13.fa
GRCh38.p13.fa.fai
GRCh38.p13.fa.sizes
GRCh38.p13.gaps.bed
GRCh38.p13.annotation.bed
GRCh38.p13.annotation.gtf
assembly_report.txt
README.txt
index/
```

An assembly name can be passed to the `install` function (fig. 1c). The genome FASTA file is downloaded with the desired sequence masking level [20,21] and alternate sequences (softmasked and none by default, respectively). Alternate sequences to reflect biological diversity and are often contained in reference assemblies. During sequence alignment however, similar reference sequences result in multiple alignment, leading to loss of data (as discussed in [22]). Additional filters may be passed to either include or exclude contigs (chromosomes, scaffolds, etc.) by name or regex pattern. Once processed, a genome index is generated using pyfaidx [15], as well as contig sizes and contig gap sizes.

Gene annotations come in a variety of recognized formats (GFF3, GTF, BED12). The `install` function will download the most descriptive format available, to output the commonly used GTF and BED12 formats. Contig names of the genome and gene annotation sometimes mismatch, which makes them incompatible with tools such as splice-aware aligners. Therefore, genomepy will attempt to match the contig names of the gene annotations to those used in the genome FASTA.

The `install` function can be extended with postprocessing steps via plugins. The options can be inspected and toggled with the `plugin` function. Briefly, the blacklist plugin downloads blacklists by the Kundaje lab [23] for the supported genomes. Other plugins support the generation of aligner indexes, including DNA aligner indexes for Bowtie2 [24], BWA [25], GMAP [26] or Minimap2 [27], and splice-aware aligners such as STAR [28] and HISAT2 [29].

Assemblies not present on the major providers can be processed similarly by supplying the URLs or file paths to the `install` function. For data provenance and reproducibility, a README file is generated during the installation process with time, source files, processing steps, and filtered contigs.

These features are available on both the command line interface and Python API. Additional features are available on the Python API, focussed around two classes. The `Genome` class can be used to extract exact or random sequences from the FASTA, filter the FASTA and list the contigs, contig sizes and contig gaps. The `Annotation` class can be used to browse and filter the BED12 or GTF files as pandas dataframes [16], map gene identifiers to other types using mygene.info [17], map chromosome names to naming schemes of other major providers, and create a dictionary of any two GTF columns or attribute fields (to easily convert gene identifiers to gene names for instance).

Conclusion

Obtaining suitable genomic data is a principal step in any genomics project. With genomepy, finding available assemblies becomes trivial. A genome, with the desired sequence masking, level of biological diversity, and contigs can be obtained with a single command. Gene annotations in GTF and BED12 format, and matching the genome, can similarly be obtained, with further options available in the Python API. Whatever install options you choose are logged, for reproducibly, allowing you to start your analysis with confidence.

Code availability

Genomepy can be installed using Bioconda, Pip, or directly used in workflows with this Docker image or snakemake wrapper. Code and documentation are available on github and github-pages, respectively.

References

1. https://academic.oup.com/nar/advance-article/doi/10.1093/nar/gkz966/5613682

2. **The Human Genome Browser at UCSC**
WJames Kent, Charles W Sugnet, Terrence S Furey, Krishna M Roskin, Tom H Pringle, Alan M Zahler, and David Haussler
Genome Research (2002-06-01) https://genome.cshlp.org/content/12/6/996
DOI: 10.1101/gr.229102

3. **The UCSC Table Browser data retrieval tool**
Donna Karolchik, Angela S Hinrichs, Terrence S Furey, Krishna M Roskin, Charles W Sugnet, David Haussler, WJames Kent
Nucleic acids research (2004-01-01) https://www.ncbi.nlm.nih.gov/pmc/articles/PMC308837/
DOI: 10.1093/nar/gkh103 · PMID: 14681465 · PMCID: PMC308837

4. https://academic.oup.com/nar/article/49/D1/D916/6018430

5. https://academic.oup.com/nar/article/47/D1/D867/5165261

6. https://academic.oup.com/nar/article/47/D1/D759/5144957

7. https://academic.oup.com/nar/advance-article/doi/10.1093/nar/gkz920/5603222

8. https://academic.oup.com/nar/article/46/D1/D861/4559118

9. **A comprehensive evaluation of ensembl, RefSeq, and UCSC annotations in the context of RNA-seq read mapping and gene quantification**
Shanrong Zhao, Baohong Zhang
BMC Genomics (2015-02-18) https://doi.org/10.1186/s12864-015-1308-8
DOI: 10.1186/s12864-015-1308-8

10. **GitHub - kblin/ncbi-genome-download: Scripts to download genomes from the NCBI FTP servers**
GitHub
https://github.com/kblin/ncbi-genome-download

11. **ucsc-genomes-downloader: Python package to quickly download genomes from the UCSC.**
Luca Cappelletti
https://github.com/LucaCappelletti94/ucsc_genomes_downloader

12. **iGenomes** https://support.illumina.com/sequencing/sequencing_software/igenome.html

13. https://academic.oup.com/gigascience/article/doi/10.1093/gigascience/giz149/5717403

14. **Go Get Data (GGD): simple, reproducible access to scientific data**
Michael J Cormier, Jonathan R Belyeu, Brent S Pedersen, Joseph Brown, Johannes Koster, Aaron R Quinlan
(2020-09-11) https://www.biorxiv.org/content/10.1101/2020.09.10.291377v2

15. **Efficient "pythonic" access to FASTA files using pyfaidx**
Matthew D Shirley, Zhaorong Ma, Brent S Pedersen, Sarah J Wheelan
PeerJ PrePrints (2015-04-08) https://peerj.com/preprints/970

16. **pandas-dev/pandas: Pandas 1.4.3**
Jeff Reback, jbrockmendel, Wes McKinney, Joris Van den Bossche, Matthew Roeschke, Tom Augspurger, Simon Hawkins, Phillip Cloud, gfyoung, Sinhrks, … Thomas Li
Zenodo (2022-06-23) https://zenodo.org/record/6702671

17. **High-performance web services for querying gene and variant annotation**
Jiwen Xin, Adam Mark, Cyrus Afrasiabi, Ginger Tsueng, Moritz Juchler, Nikhil Gopal, Gregory S Stupp, Timothy E Putman, Benjamin J Ainscough, Obi L Griffith, … Chunlei Wu
Genome Biology (2016-05-06) https://doi.org/10.1186/s13059-016-0953-9
DOI: 10.1186/s13059-016-0953-9

18. https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btr539

19. **CellOracle: Dissecting cell identity via network inference and in silico gene perturbation**
Kenji Kamimoto, Christy M Hoffmann, Samantha A Morris
(2020-04-21) https://www.biorxiv.org/content/10.1101/2020.02.17.947416v3

20. **RepeatMasker Home Page** http://repeatmasker.org/

21. **A Fast and Symmetric DUST Implementation to Mask Low-Complexity DNA Sequences**
Aleksandr Morgulis, EMichael Gertz, Alejandro A Schäffer, Richa Agarwala
Journal of Computational Biology (2006-06) https://doi.org/fqs85g
DOI: 10.1089/cmb.2006.13.1028 · PMID: 16796549

22. **Extending reference assembly models**
Deanna M Church, Valerie A Schneider, Karyn Meltz Steinberg, Michael C Schatz, Aaron R Quinlan, Chen-Shan Chin, Paul A Kitts, Bronwen Aken, Gabor T Marth, Michael M Hoffman, … Paul Flicek
Genome Biology (2015-01-24) https://doi.org/10.1186/s13059-015-0587-3
DOI: 10.1186/s13059-015-0587-3

23. **The ENCODE Blacklist: Identification of Problematic Regions of the Genome**
Haley M Amemiya, Anshul Kundaje, Alan P Boyle
Scientific Reports (2019-06-27) https://www.nature.com/articles/s41598-019-45839-z
DOI: 10.1038/s41598-019-45839-z

24. **Fast gapped-read alignment with Bowtie 2**
Ben Langmead, Steven L Salzberg
Nature Methods (2012-04) https://www.nature.com/articles/nmeth.1923
DOI: 10.1038/nmeth.1923

25. https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/btp324

26. https://academic.oup.com/HTTPHandlers/Sigma/LoginHandler.ashx?error=login_required&state=8ad838c3-98fb-4d9f-ab24-fd44b62291earedirecturl%3Dhttpszazjzjacademiczwoupzwcomzjbioinformaticszjarticlezylookupzjdoizj10zw1093zjbioinformaticszjbti310

27. https://academic.oup.com/bioinformatics/article/34/18/3094/4994778

28. https://academic.oup.com/bioinformatics/article-lookup/doi/10.1093/bioinformatics/bts635

29. **HISAT: a fast spliced aligner with low memory requirements**
Daehwan Kim, Ben Langmead, Steven L Salzberg
Nature Methods (2015-04) https://www.nature.com/articles/nmeth.3317
DOI: 10.1038/nmeth.3317