Extract genome region

Release 0.0.3

May 24, 2016

Contents

Contents:

CHAPTER 1

Overview

1.1 Extract genome region

docs	
tests	
package	

Given a CSV file of variable information defining the regions of interest, return a file that contains a fasta-formatted representation of these regions.

- Free software: BSD license

1.1.1 Usage

```
$ extract_genome_region --help
Usage: extract_genome_region [OPTIONS] REGIONS IN_FASTA OUT_FASTA

  Given a CSV file of variable information defining the regions of interest
  along with input and output fasta file paths, write a file that contains a
  fasta-formatted representation of these regions.

  Structure of the `regions` CSV file:

    record_name   The name you want the seq to have in the new fasta.
       scaffold   The name of the seq record in the source fasta (chromosome, scaffold, contig, etc)
          start   The first bp of the seq feature you want in the new fasta.
           stop   The last bp of the seq feature you want in the new fasta.
       left_bfr   How many "extra" bp with coords smaller than `start` you want (0 for none).
      right_bfr   How many "extra" bp with coords larger than `stop` you want (0 for none).

  Naming Strategies:

            csv   use only the contents of the `record_name` field in the csv file (>CPR23).
      seq_range   use only the `scaffold` name and sequence range (>scaffold1:230-679).
  csv_seq_range   use both the contents of the `record_name` field and the `scaffold`.
                  name and sequence range (>CPR23 scaffold1:230-679).

Options:
  -n, --naming [csv|seq_range|csv_seq_range]
                                  Options regarding how each new fasta record
                                  will be named. See main help-text for
                                  explainations of options. [default='csv']
  --help                          Show this message and exit.
```

1.1.2 Installation

```
$ conda install -c bioconda -c gusdunn extract_genome_region
```

Or

```
$ pip install extract_genome_region
```

1.1.3 Documentation

https://extract-genome-region.readthedocs.org/

1.1.4 Development

To run the all tests run:

```
tox
```

CHAPTER 2

Installation

2.1 With Conda

2.1.1 Add package channels to your configuration

Note: Our goal here is to make sure you have a couple anaconda.org channels in your `.condarc` file. If you know you have these, you can skip this section. The channels are: bioconda, r, gusdunn.

Lets see what your conda installation looks like to see what needs to be tweaked in order for the installation scripts to "see" the package channels that we will need. To do this, issue the following command at the terminal:

```
$ conda info
```

Here is mine:

```
$ conda info
Using Anaconda Cloud api site https://api.anaconda.org
Current conda install:

             platform : linux-64
        conda version : 4.0.6
  conda-build version : 1.20.2
       python version : 3.5.1.final.0
     requests version : 2.10.0
     root environment : /home/gus/.anaconda  (writable)
  default environment : /home/gus/.anaconda/envs/jupyter
     envs directories : /home/gus/.anaconda/envs
        package cache : /home/gus/.anaconda/pkgs
         channel URLs : https://conda.anaconda.org/bioconda/linux-64/
                        https://conda.anaconda.org/bioconda/noarch/
                        https://conda.anaconda.org/gusdunn/linux-64/
                        https://conda.anaconda.org/gusdunn/noarch/
                        https://conda.anaconda.org/t/<TOKEN>/r/linux-64/
                        https://conda.anaconda.org/t/<TOKEN>/r/noarch/
                        https://conda.anaconda.org/t/<TOKEN>/pandas/linux-64/
                        https://conda.anaconda.org/t/<TOKEN>/pandas/noarch/
                        https://repo.continuum.io/pkgs/free/linux-64/
                        https://repo.continuum.io/pkgs/free/noarch/
                        https://repo.continuum.io/pkgs/pro/linux-64/
                        https://repo.continuum.io/pkgs/pro/noarch/
                        https://conda.anaconda.org/davidbgonzalez/linux-64/
                        https://conda.anaconda.org/davidbgonzalez/noarch/
          config file : /home/gus/.condarc
    is foreign system : False
```

Right now, we want to see what is in your "config file". The second to last line tells me that mine is located at `/home/gus/.condarc`. If you have a brand new installation, you might not have one yet. If the report says `None`, we can create one and add the channels easily:

```
$ conda config --add channels gusdunn --add channels r --add channels pandas --add channels bioconda
```

Note: Even if you **do** have one, you can run the same command to add the channels. If any channel exists already, it will be skipped.

2.1.2 Install extract_genome_region

Now you will want to activate the conda environment where you want to install `extract_genome_region`. You do that by running the following (substituting the name of your environment for `ENVNAME`):

```
$ source activate ENVNAME
```

Next lets run the install:

```
$ conda install extract_genome_region
```

2.2 With pip

Note: I recommend that you use `conda` rather than `pip`, but `pip` should also work.

```
$ pip install git+https://github.com/xguse/extract-genome-region
```

2.3 Confirming success

Let's make sure it worked by calling the program's help text. You should get something similar to this:

```
$ extract_genome_region --help
Usage: extract_genome_region [OPTIONS] REGIONS IN_FASTA OUT_FASTA

  Given a CSV file of variable information defining the regions of interest
  along with input and output fasta file paths, write a file that contains a
  fasta-formatted representation of these regions.

  Structure of the `regions` CSV file:

    record_name   The name you want the seq to have in the new fasta.
       scaffold   The name of the seq record in the source fasta (chromosome, scaffold, contig, etc)
          start   The first bp of the seq feature you want in the new fasta.
           stop   The last bp of the seq feature you want in the new fasta.
       left_bfr   How many "extra" bp with coords smaller than `start` you want (0 for none).
      right_bfr   How many "extra" bp with coords larger than `stop` you want (0 for none).

  Naming Strategies:

            csv   use only the contents of the `record_name` field in the csv file (>CPR23).
      seq_range   use only the `scaffold` name and sequence range (>scaffold1:230-679).
  csv_seq_range   use both the contents of the `record_name` field and the `scaffold`.
                  name and sequence range (>CPR23_scaffold1:230-679).

Options:
  -n, --naming [csv|seq_range|csv_seq_range]
                                  Options regarding how each new fasta record
                                  will be named. See main help-text for
                                  explainations of options. [default='csv']
  --help                          Show this message and exit.
```

CHAPTER 3

Usage

```
$ extract_genome_region --help
Usage: extract_genome_region [OPTIONS] REGIONS IN_FASTA OUT_FASTA

  Given a CSV file of variable information defining the regions of interest
  along with input and output fasta file paths, write a file that contains a
  fasta-formatted representation of these regions.

  Structure of the `regions` CSV file:

    record_name   The name you want the seq to have in the new fasta.
       scaffold   The name of the seq record in the source fasta (chromosome, scaffold, contig, etc).
          start   The first bp of the seq feature you want in the new fasta.
           stop   The last bp of the seq feature you want in the new fasta.
       left_bfr   How many "extra" bp with coords smaller than `start` you want (0 for none).
      right_bfr   How many "extra" bp with coords larger than `stop` you want (0 for none).

  Naming Strategies:

            csv   use only the contents of the `record_name` field in the csv file (>CPR23).
      seq_range   use only the `scaffold` name and sequence range (>scaffold1:230-679).
  csv_seq_range   use both the contents of the `record_name` field and the `scaffold`.
                  name and sequence range (>CPR23 scaffold1:230-679).

Options:
  -n, --naming [csv|seq_range|csv_seq_range]
                                  Options regarding how each new fasta record
                                  will be named. See main help-text for
                                  explainations of options. [default='csv']
  --help                          Show this message and exit.
```

CHAPTER 4

Reference

4.1 extract_genome_region package

4.1.1 Module contents

Given a CSV file of variable information defining the regions of interest along with input and output fasta file paths, write a file that contains a fasta-formatted representation of these regions.

`extract_genome_region.__main__.`**`gen_coords`**(*records*)

Given records as a `namedtuple`, yield coordinate information as a `namedtuple`.

Parameters **`records`** (*`namedtuple`*) – each row info from the "regions" CSV file.

Yields *namedtuple* – the actual coordinates for slicing the fasta sequence (accounting for any buffers) for a single row in the "regions" CSV file.

Note: The coordinates in each yielded `namedtuple` will assume slicing indexing of standard python strings (zero-based).

`extract_genome_region.__main__.`**`gen_faidx_objs`**(*fasta*, *coords*, *naming_strategy=None*)

Given the pyfaidx fasta obj and the coords generator, yield each sequence slice as `pyfaidx.Sequence` objs.

Parameters

- **`fasta`** (*`faidx.Fasta`*) – faidx fasta object.
- **`coords`** (*`generator`*) – of row information from "regions" CSV file.
- **`naming_strategy`** (*`str`*) – [csv|seq_range|csv_seq_range] how to name each record. If `None`, use coord.record_name as string.

Yields *generator* – of faidx sequence objects (`faidx.Sequence`) for each row in the "regions" CSV file.

`extract_genome_region.__main__.`**`gen_out_rec_strings`**(*faidx_objs*)

Yield the fasta formated record: ready for writing out.

Parameters **`faidx_objs`** (*`generator`*) – of `faidx.Sequence` objects representing the described region of each row in "regions" CSV file.

Yields *generator* – of formated `str` objects representing the fasta record of the described region of each row in "regions" CSV file.

`extract_genome_region.__main__.`**`gen_records`**(*path*)

Given the csv *path*, yield each record as a *namedtuple*.

Parameters **`path`** (*`str`*) – location of the "regions" CSV file.

Yields *namedtuple* – each row info from the "regions" CSV file.

CHAPTER 5

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

5.1 Bug reports

When reporting a bug please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.2 Documentation improvements

Extract genome region could always use more documentation, whether as part of the official Extract genome region docs, in docstrings, or even on the web in blog posts, articles, and such.

5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/xguse/extract-genome-region/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.4 Development

To set up *extract-genome-region* for local development:

1. Fork extract-genome-region on GitHub.
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/extract-genome-region.git
```

3. Create a branch for local development:

```
git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. Commit your changes and push your branch to GitHub:

```
git add .
git commit -m "Your detailed description of your changes."
git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

5.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`) [1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

5.4.2 Tips

To run a subset of tests:

```
tox -e envname -- py.test -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.

It will be slower though ...

CHAPTER 6

Authors

- Gus Dunn - https://github.com/xguse

CHAPTER 7

Changelog

7.1 0.0.3 (2016-05-24)

- altered meaning of ‘csv_seq_range’ to allow splitting fasta key on whitespace to ignore scaffold:range info

7.2 0.0.2 (2016-05-24)

- expanded the help text
- built the docs

7.3 0.0.1 (2016-05-23)

- First release on anaconda.org/gusdunn.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index