microbio Documentation

Release 0.1

Gregor Sturm

April 27, 2016

Contents

a minimalist Bioinformatics framework

For most of my use cases, extensive frameworks such as Biopython or scikit-bio are complete overkill. These frameworks are handling everything as an object (e.g. Sequences) which is certainly helpful in some cases, but often it's just more straightforward to save the sequences as plain python strings.

This framework aims at providing a bunch of modules for bioinformatics that are lightweight and super-easy to use. At the moment it is limited to some file reading and writing routines, but who knows what comes in the future ;)

CHAPTER 1

Installation

```
pip install microbio
```

CHAPTER 2

Contributing

I will extend this project as it serves my projects. If you have useful code to add, feel free to send a pull request.

CHAPTER 3

Documentation

http://microbio.readthedocs.org/

CHAPTER 4

Contents

4.1 File Formats (Input/Output)

4.1.1 Fasta

Reading fasta

is as easy as

```
r = FastaReader(file)
for header, seq in r.get_entries():
    print(header)
    print(seq)
```

file can either be a path or a file object.

class formats.fasta.**FastaReader**(*file*)
Read fasta files into tuples (header, seq).

close()
Close file handle

get_entries()
Get the next Entry from the fasta file.

Returns Generator, which yields (header, sequence) tuples

Writing fasta

```
w = FastaWriter(file, split=60)
header = "some_random_nucleotides"
seq = "ACTGACATT"
w.write_entry(header, seq)
w.close()
```

again, file can be a path of a file object. `split` speficies, after how many characters a sequence will be wrapped in multiple lines. Default is 80.

class formats.fasta.**FastaWriter**(*file*, *split=80*)
Write fasta files from tuples (header, seq)

close()
Close file handle

write_entry(*header*, *sequence*)
Write Entry to File

Parameters

- **header** – >sequence_header (without >)
- **sequence** – ACTGATT...

4.1.2 Fast5

Fast5 is a sequence format generated by [Oxford Nanopore](http://nanoporetech.com) devices. This class is designed to read formats up to the sequencing kit SQK-MAP006). Compatibility with more recent formats is not guaranteed.

As it serves for rather exotic use-cases, this file will most likely drop out of future versions of this framework.

Example

```
f5 = Fast5File('lambda_burnin_ch101_file2_strand.fast5')
seq = f5.get_seq('template')
```

class formats.fast5.**Fast5File**(*path*)

static events2seq(*events*)
turn events into a nt sequence based on the 'move' column

Parameters **events** – list of events (dictionaries)

get_attrs(*strand*)
Get Attributes for template/complement strand

Parameters **strand** – either "template" or "complement"

Returns {shift: foo, drift: foo, scale: foo} or None if strand not in File

Return type dict

get_corrected_events(*strand*)
Get events for template/complement strand and apply the shift/scale/drift corrections. Unfortunately, these corrections are not documented exactly anywhere. Some information is on https://wiki.nanoporetech.com/display/BP/1D+Basecalling+overview.

Parameters **strand** – either "template" or "complement"

Returns generator yielding one event-dict at a time or None if strand not in File

get_events(*strand*)
Get events for template/complement strand

Parameters **strand** – either "template" or "complement"

Returns generator yielding one event-dict at a time or None if strand not in File

get_id()

Returns unique identifier for f5 file.

get_seq(*strand*)
get the nt-sequence, based on the kmers and 'move'. Evaluation is done in a lazy fashion. If the function was called once, the sequence can be accessed in constant time.

Parameters **strand** (*str*) – either template or complement

Returns (str) nucleotide sequence

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

f

C

E

F

G

W