cortexpy Documentation

Release 0.46.4

Warren Kretzschmar and Kiran Garimella

Apr 25, 2019

Contents

CHAPTER 1

Overview of Cortexpy

tests	
package	
docs	

Cortexpy is a Python package for sequence analysis using linked and colored De Bruijn graphs such as the ones created by Cortex and Mccortex. This project aims to mirror many of the features contained in CortexJDK.

Cortexpy also comes with a command-line tool for basic inspection and manipulation of Cortex graphs with and without links.

1.1 Audience

The audience of cortexpy is researchers working with colored De Bruijn graphs and link information in Cortex and Mccortex format.

1.2 Free software

Cortexpy is free software; you can redistribute it and/or modify it under the terms of the Apache License version 2.0. Contributions are welcome. Please join us on GitHub.

1.3 Installation

```
pip install cortexpy
```

1.4 Documentation

For more information, please see cortexpy documentation.

1.5 Citing cortexpy

If you use cortexpy in your work, please consider citing:

> Akhter, Shirin, Warren W. Kretzschmar, Veronika Nordal, Nicolas Delhomme, Nathaniel R. Street, Ove Nilsson, Olof Emanuelsson, and Jens F. Sundström. "Integrative analysis of three RNA sequencing methods identifies mutually exclusive exons of MADS-box isoforms during early bud development in *Picea abies*." *Frontiers in Plant Science* 9 (2018). https://doi.org/10.3389/fpls.2018.01625

1.6 Bugs

This code is maintained by Warren Kretzschmar <warrenk@kth.se>. For bugs, please raise a GitHub issue.

1.7 Development

1. Install conda.
2. Download development and testing tools:

   ```
   conda env create -f environment.yml -n my-dev-environment
   ```

3. Activate development environment:

   ```
   conda activate my-dev-environment
   ```

All remaining commands in the development section need to be run in an activated conda dev environment.

1.7.1 Tests

```
make test
```

1.7.2 Deploy new cortexpy version to pypi

Requires access credentials for pypi.

```
make deploy
```

1.7.3 Building the docs

The documentation is automatically built by read-the-docs on push to master. To build the documentation manually:

```
# install sphinx dependencies
pip install docs/requirements.txt

make docs
```

1.7.4 Updating the dev environment

This section is experimental because it does not work on travis-CI yet.

```
# Create a new env from the high-level requirements file
conda env create -f environment.yml -n another-dev-env

# activate the new environment
conda activate another-dev-env

# save new env to environment.lock.yml
make lock
```

CHAPTER 2

Tutorial

The cortexpy package consists of a python API and a command-line tool for working with Cortex graphs. Below, we start by looking at how to use the python API to perform an example workflow.

2.1 Using the python API to filter Cortex graphs

2.1.1 Building Cortex files

Let's start by by creating two Cortex files to work with. At present, cortexpy does not provide a way to easily create a Cortex file, so we will instead use Mccortex. Mccortex can be compiled from source or installed using bioconda.

Let's start by creating two FASTA files from which to create two Cortex files:

```
echo -e '>1\nAAAAA' > file1.fasta
echo -e '>1\nCCCCC' > file2.fasta
```

We now have two FASTA files each containing a single sequence. We can now build a Cortex graph from each file. We choose to use a kmer-size of 5:

```
mccortex 5 build --sort -k 5 --sample file1 -1 file1.fasta file1.ctx
mccortex 5 build --sort -k 5 --sample file2 -1 file2.fasta file2.ctx
```

We now have two cortex files: `file1.ctx` and `file2.ctx`. As the Cortex format represents colored De Bruijn graphs, we could have stored the information from the two FASTA files in a single graph as two separate colors. However, we are creating two files in order to demonstrate the cortexpy API later on.

We can check what kmers are stored in each graph using the **`cortexpy`** command-line tool:

```
> cortexpy view graph file1.ctx
AAAAA 1 ........

> cortexpy view graph file2.ctx
CCCCC 1 ........
```

This output tells us that each graph consists of a single kmer with coverage 1.

2.1.2 Inspecting Cortex graphs in Python

Cortexpy offers many ways to inspect Cortex files. Much of that functionality is available through the `RandomAccess` class. Let us start by loading a Cortex file inside python:

```
>>> from cortexpy.graph.parser.random_access import RandomAccess
>>> # make sure to open the cortex graph in binary mode
>>> ra = RandomAccess(open('file1.ctx', 'rb'))
```

We can now interrogate the `ra` object. Let's see what the header size of the Cortex file is:

```
>>> ra.header.kmer_size
5
```

Let's check if the kmer `AAAAA` exists in the graph and retrieve it:

```
>>> 'AAAAA' in ra
True
>>> ra['AAAAA']
Kmer(_kmer_data=KmerData(_data=b'\x00\x00\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x00
↪', kmer_size=5, num_colors=1, _kmer='AAAAA', _coverage=None, _edges=None), num_
↪colors=1, kmer_size=5, _revcomp=None)
```

We can see that the returned kmer object contains information on the kmer size (5) and the number of colors stored in the kmer (1).

Now let's put it all together and search both graphs that we created while *Building Cortex files* for our kmer of interest, `AAAAA`:

Listing 1: search.py

```
from cortexpy.graph.parser.random_access import RandomAccess

for graph in ['file1.ctx', 'file2.ctx']:
    # make sure to open the cortex graph in binary mode
    with open(graph, 'rb') as fh:
        ra = RandomAccess(fh)

        # let's see if our favorite kmer is in the graph
        if 'AAAAA' in ra:
            print(f'AAAAA exists in {graph}!')
```

This is what we see if we run this code from the command line:

```
> python3 search.py
AAAAA exists in file1.ctx!
```

CHAPTER 3

API reference

3.1 Random access of Cortex graphs

This module contains classes for inspecting Cortex graphs with random access to their kmers.

class cortexpy.graph.parser.random_access.**RandomAccess**(*graph_handle*, *kmer_cache_size=None*)

Provide fast k-mer access to Cortex graph in log(n) time (n = number of kmers in graph)

__getitem__(*lexlo_string*)

Return kmer associated with kmer string

No check is performed to make sure that the input string is a lexicographically-lowest kmer string. Use *get_kmer_for_string()* in order to convert a kmer string to its lexlo form before retrieving it from the cortex object.

__iter__()

Iterate over kmer strings in graph in order stored in graph

get_kmer_for_string(*string*)

Will compute the revcomp of kmer string before getting a kmer

items()

Iterate over kmer strings and kmers in graph in order stored in graph

values()

Iterate over kmers in cortex graph

3.2 Cortex graph headers

This module contains classes for parsing and representing a Cortex file header

class cortexpy.graph.parser.header.**Header**(*version=6*, *kmer_size=1*, *kmer_container_size=None*, *num_colors=None*, *mean_read_lengths=None*, *total_sequences=None*, *sample_names=None*, *error_rates=None*, *color_info_blocks=NOTHING*)

Cortex header object

This object allows access to header information contained in a cortex file

classmethod from_stream(*stream*)

Extract a cortex header from a file handle

3.3 Cortex kmers

This module provides classes and functions for working with Cortex kmers.

class cortexpy.graph.parser.kmer.**Kmer**(*kmer_data*, *num_colors*, *kmer_size*, *revcomp=None*)

Represents a Cortex kmer

This class wraps a kmer data object with attributes and methods for inspecting and manipulating the underlying kmer data object.

increment_color_coverage(*color*)

Increment the coverage of a color by one

class cortexpy.graph.parser.kmer.**StringKmerConverter**(*kmer_size*)

Converts kmer strings to various binary representations

to_uints(*kmer_string*)

Converts kmer_string to big-endian uint64 array

cortexpy.graph.parser.kmer.**connect_kmers**(*first*, *second*, *color*, *identical_kmer_check=True*)

Connect two kmers

cortexpy.graph.parser.kmer.**disconnect_kmers**(*first*, *second*, *colors*)

Disconnect two kmers

cortexpy.graph.parser.kmer.**find_all_neighbors**(*first*, *second*)

Return kmers and letters to get from first kmer to second

3.4 Utility functions

This module contains utility functions that are used inside cortexpy. These functions may also be useful outside of cortexpy.

cortexpy.utils.**kmerize_contig**(*contig*, *kmer_size*)

Return generator of kmers in contig

The returned kmers are not lexicographically lowest.

```
>>> list(kmerize_contig('ATTT', 3))
['ATT', 'TTT']
```

cortexpy.utils.**kmerize_fasta**(*fasta*, *kmer_size*)

Return generator to all kmers in fasta

cortexpy.utils.**lexlo**

Return lexicographically lowest version of a kmer string and its reverse complement

The reverse complement of a kmer string is generated and the lexicographically-lowest kmer string is returned.

```
>>> lexlo('AAA')
'AAA'
```

```
>>> lexlo('TTT')
'AAA'
```

CHAPTER 4

License

Cortexpy is distributed under the Apache Lincense version 2.0:

```
                                 Apache License
                           Version 2.0, January 2004
                        http://www.apache.org/licenses/

TERMS AND CONDITIONS FOR USE, REPRODUCTION, AND DISTRIBUTION

1. Definitions.

   "License" shall mean the terms and conditions for use, reproduction,
   and distribution as defined by Sections 1 through 9 of this document.

   "Licensor" shall mean the copyright owner or entity authorized by
   the copyright owner that is granting the License.

   "Legal Entity" shall mean the union of the acting entity and all
   other entities that control, are controlled by, or are under common
   control with that entity. For the purposes of this definition,
   "control" means (i) the power, direct or indirect, to cause the
   direction or management of such entity, whether by contract or
   otherwise, or (ii) ownership of fifty percent (50%) or more of the
   outstanding shares, or (iii) beneficial ownership of such entity.

   "You" (or "Your") shall mean an individual or Legal Entity
   exercising permissions granted by this License.

   "Source" form shall mean the preferred form for making modifications,
   including but not limited to software source code, documentation
   source, and configuration files.

   "Object" form shall mean any form resulting from mechanical
   transformation or translation of a Source form, including but
   not limited to compiled object code, generated documentation,
```

(continues on next page)

(continued from previous page)

```
   and conversions to other media types.

   "Work" shall mean the work of authorship, whether in Source or
   Object form, made available under the License, as indicated by a
   copyright notice that is included in or attached to the work
   (an example is provided in the Appendix below).

   "Derivative Works" shall mean any work, whether in Source or Object
   form, that is based on (or derived from) the Work and for which the
   editorial revisions, annotations, elaborations, or other modifications
   represent, as a whole, an original work of authorship. For the purposes
   of this License, Derivative Works shall not include works that remain
   separable from, or merely link (or bind by name) to the interfaces of,
   the Work and Derivative Works thereof.

   "Contribution" shall mean any work of authorship, including
   the original version of the Work and any modifications or additions
   to that Work or Derivative Works thereof, that is intentionally
   submitted to Licensor for inclusion in the Work by the copyright owner
   or by an individual or Legal Entity authorized to submit on behalf of
   the copyright owner. For the purposes of this definition, "submitted"
   means any form of electronic, verbal, or written communication sent
   to the Licensor or its representatives, including but not limited to
   communication on electronic mailing lists, source code control systems,
   and issue tracking systems that are managed by, or on behalf of, the
   Licensor for the purpose of discussing and improving the Work, but
   excluding communication that is conspicuously marked or otherwise
   designated in writing by the copyright owner as "Not a Contribution."

   "Contributor" shall mean Licensor and any individual or Legal Entity
   on behalf of whom a Contribution has been received by Licensor and
   subsequently incorporated within the Work.

2. Grant of Copyright License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   copyright license to reproduce, prepare Derivative Works of,
   publicly display, publicly perform, sublicense, and distribute the
   Work and such Derivative Works in Source or Object form.

3. Grant of Patent License. Subject to the terms and conditions of
   this License, each Contributor hereby grants to You a perpetual,
   worldwide, non-exclusive, no-charge, royalty-free, irrevocable
   (except as stated in this section) patent license to make, have made,
   use, offer to sell, sell, import, and otherwise transfer the Work,
   where such license applies only to those patent claims licensable
   by such Contributor that are necessarily infringed by their
   Contribution(s) alone or by combination of their Contribution(s)
   with the Work to which such Contribution(s) was submitted. If You
   institute patent litigation against any entity (including a
   cross-claim or counterclaim in a lawsuit) alleging that the Work
   or a Contribution incorporated within the Work constitutes direct
   or contributory patent infringement, then any patent licenses
   granted to You under this License for that Work shall terminate
   as of the date such litigation is filed.

4. Redistribution. You may reproduce and distribute copies of the
```

(continues on next page)

(continued from previous page)

```
   Work or Derivative Works thereof in any medium, with or without
   modifications, and in Source or Object form, provided that You
   meet the following conditions:

   (a) You must give any other recipients of the Work or
       Derivative Works a copy of this License; and

   (b) You must cause any modified files to carry prominent notices
       stating that You changed the files; and

   (c) You must retain, in the Source form of any Derivative Works
       that You distribute, all copyright, patent, trademark, and
       attribution notices from the Source form of the Work,
       excluding those notices that do not pertain to any part of
       the Derivative Works; and

   (d) If the Work includes a "NOTICE" text file as part of its
       distribution, then any Derivative Works that You distribute must
       include a readable copy of the attribution notices contained
       within such NOTICE file, excluding those notices that do not
       pertain to any part of the Derivative Works, in at least one
       of the following places: within a NOTICE text file distributed
       as part of the Derivative Works; within the Source form or
       documentation, if provided along with the Derivative Works; or,
       within a display generated by the Derivative Works, if and
       wherever such third-party notices normally appear. The contents
       of the NOTICE file are for informational purposes only and
       do not modify the License. You may add Your own attribution
       notices within Derivative Works that You distribute, alongside
       or as an addendum to the NOTICE text from the Work, provided
       that such additional attribution notices cannot be construed
       as modifying the License.

   You may add Your own copyright statement to Your modifications and
   may provide additional or different license terms and conditions
   for use, reproduction, or distribution of Your modifications, or
   for any such Derivative Works as a whole, provided Your use,
   reproduction, and distribution of the Work otherwise complies with
   the conditions stated in this License.

5. Submission of Contributions. Unless You explicitly state otherwise,
   any Contribution intentionally submitted for inclusion in the Work
   by You to the Licensor shall be under the terms and conditions of
   this License, without any additional terms or conditions.
   Notwithstanding the above, nothing herein shall supersede or modify
   the terms of any separate license agreement you may have executed
   with Licensor regarding such Contributions.

6. Trademarks. This License does not grant permission to use the trade
   names, trademarks, service marks, or product names of the Licensor,
   except as required for reasonable and customary use in describing the
   origin of the Work and reproducing the content of the NOTICE file.

7. Disclaimer of Warranty. Unless required by applicable law or
   agreed to in writing, Licensor provides the Work (and each
   Contributor provides its Contributions) on an "AS IS" BASIS,
   WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or
```

(continues on next page)

(continued from previous page)

```
   implied, including, without limitation, any warranties or conditions
   of TITLE, NON-INFRINGEMENT, MERCHANTABILITY, or FITNESS FOR A
   PARTICULAR PURPOSE. You are solely responsible for determining the
   appropriateness of using or redistributing the Work and assume any
   risks associated with Your exercise of permissions under this License.

8. Limitation of Liability. In no event and under no legal theory,
   whether in tort (including negligence), contract, or otherwise,
   unless required by applicable law (such as deliberate and grossly
   negligent acts) or agreed to in writing, shall any Contributor be
   liable to You for damages, including any direct, indirect, special,
   incidental, or consequential damages of any character arising as a
   result of this License or out of the use or inability to use the
   Work (including but not limited to damages for loss of goodwill,
   work stoppage, computer failure or malfunction, or any and all
   other commercial damages or losses), even if such Contributor
   has been advised of the possibility of such damages.

9. Accepting Warranty or Additional Liability. While redistributing
   the Work or Derivative Works thereof, You may choose to offer,
   and charge a fee for, acceptance of support, warranty, indemnity,
   or other liability obligations and/or rights consistent with this
   License. However, in accepting such obligations, You may act only
   on Your own behalf and on Your sole responsibility, not on behalf
   of any other Contributor, and only if You agree to indemnify,
   defend, and hold each Contributor harmless for any liability
   incurred by, or claims asserted against, such Contributor by reason
   of your accepting any such warranty or additional liability.

END OF TERMS AND CONDITIONS

APPENDIX: How to apply the Apache License to your work.

   To apply the Apache License to your work, attach the following
   boilerplate notice, with the fields enclosed by brackets "[]"
   replaced with your own identifying information. (Don't include
   the brackets!)  The text should be enclosed in the appropriate
   comment syntax for the file format. We also recommend that a
   file or class name and description of purpose be included on the
   same "printed page" as the copyright notice for easier
   identification within third-party archives.

Copyright [yyyy] [name of copyright owner]

Licensed under the Apache License, Version 2.0 (the "License");
you may not use this file except in compliance with the License.
You may obtain a copy of the License at

    http://www.apache.org/licenses/LICENSE-2.0

Unless required by applicable law or agreed to in writing, software
distributed under the License is distributed on an "AS IS" BASIS,
WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
See the License for the specific language governing permissions and
limitations under the License.
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search

Python Module Index

C

Index

Symbols

C

D

F

G

H

I

K

L

R

S

T

V