NPLinker Community Meeting

2024-09-10

netherlands eScience center

Agenda

16:00	Short Introduction
16:05	Announcements and upcoming events
16:10	NPLinker development
16:25	Monthly plan
16:30	Q&A
17:00	.

netherlands eScience center

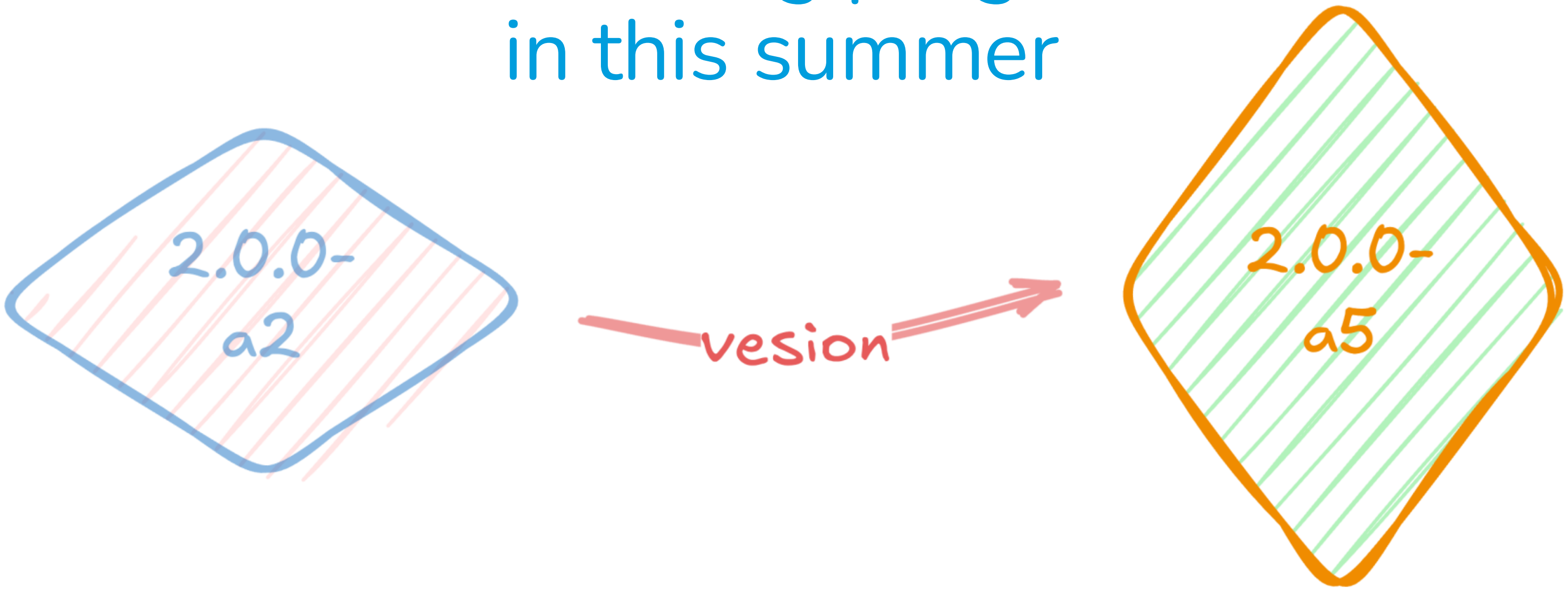
Announcements and upcoming events?

NPLinker Development

New features & changes

Contributors: Cunliang Geng, Giulia Crocioni, Arjan Draisma

Refactoring progress in this summer



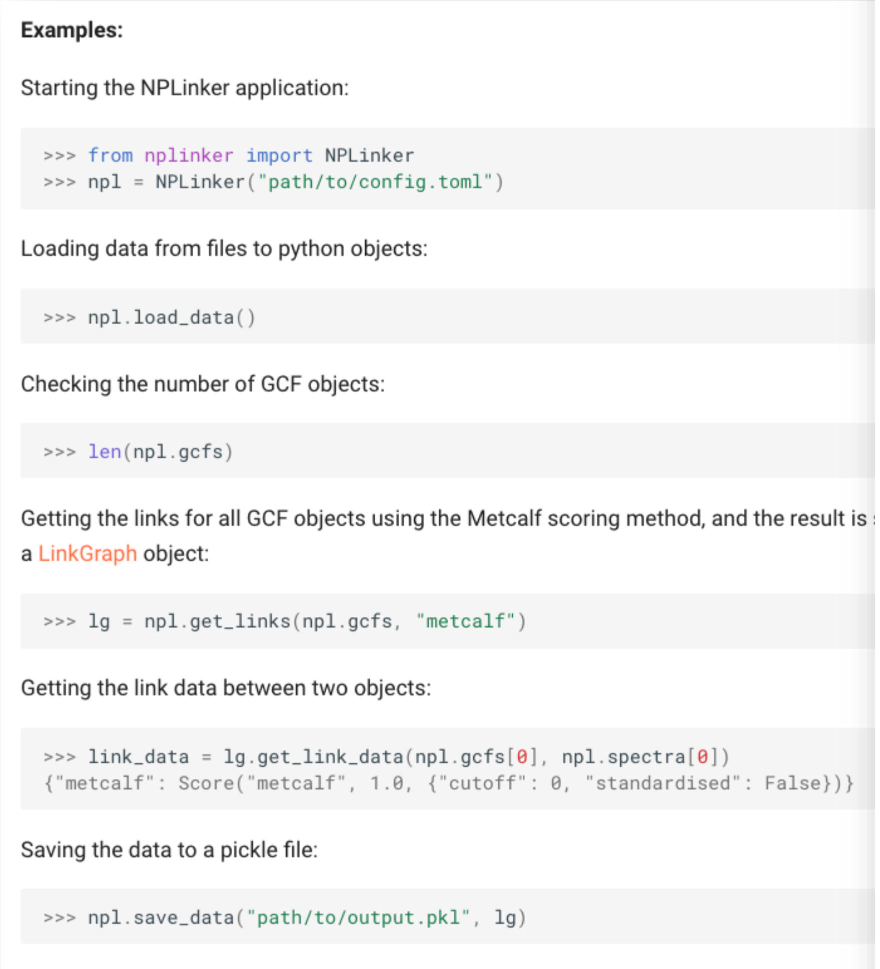
Added

- The check of available disk space when downloading data. You will get a warning if having <50GB
- Warning with details when it fails to download some antiSMASH data
- OpenSSF scorecard for code quality and security openssf scorecard 6

Improved

- The detection of tabular file delimiter
- docstrings: added code examples, link to specific function/class, and more explanations

Improved docstrings -> doc



Examples:

Starting the NPLinker application:

```
>>> from nplinker import NPLinker
>>> npl = NPLinker("path/to/config.toml")
```

Loading data from files to python objects:

```
>>> npl.load_data()
```

Checking the number of GCF objects:

```
>>> len(npl.gcfs)
```

Getting the links for all GCF objects using the Metcalf scoring method, and the result is a LinkGraph object:

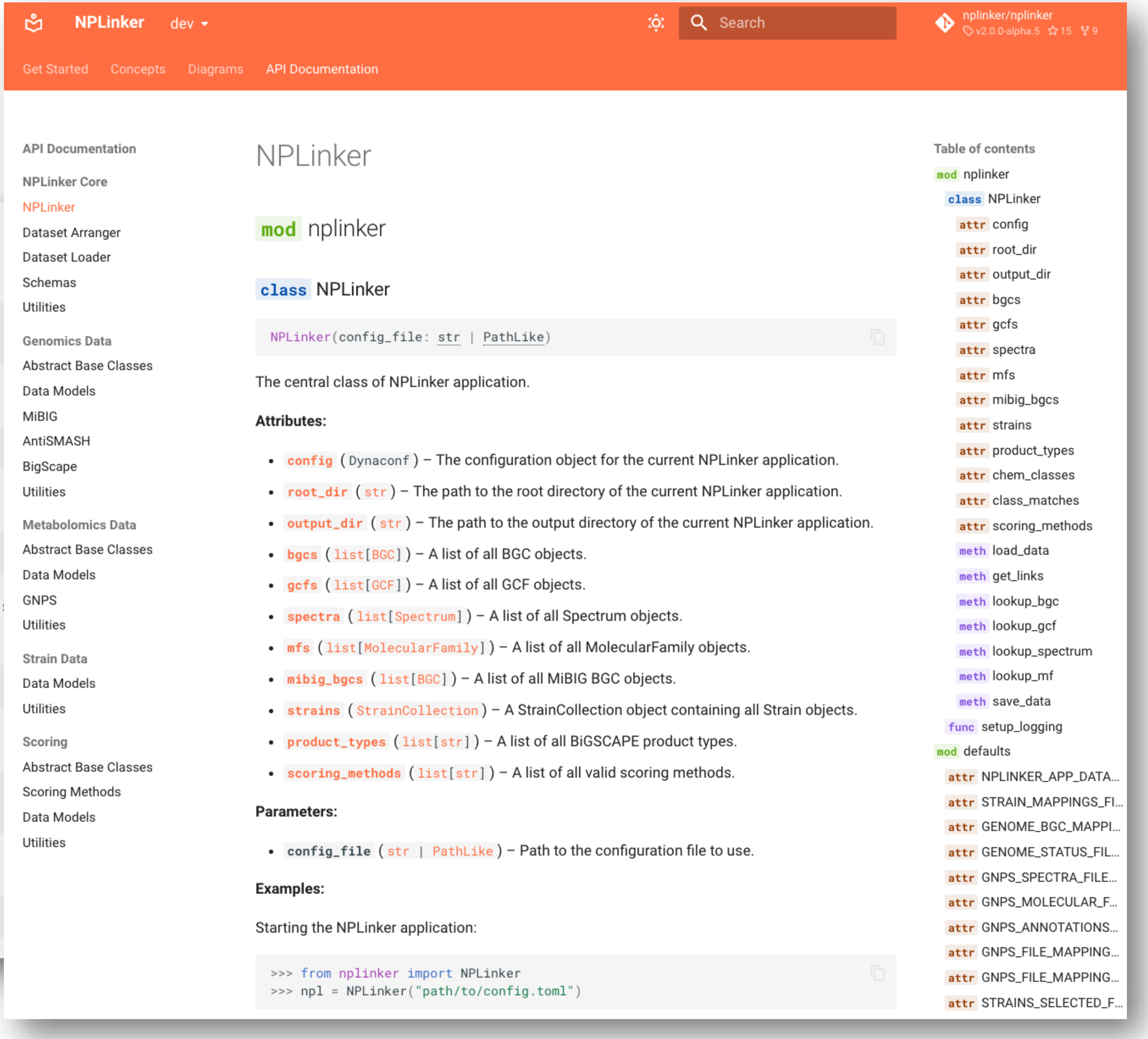
```
>>> lg = npl.get_links(npl.gcfs, "metcalf")
```

Getting the link data between two objects:

```
>>> link_data = lg.get_link_data(npl.gcfs[0], npl.spectra[0])
{"metcalf": Score("metcalf", 1.0, {"cutoff": 0, "standardised": False})}
```

Saving the data to a pickle file:

```
>>> npl.save_data("path/to/output.pkl", lg)
```



NPLinker

mod nplinker

class NPLinker

```
NPLinker(config_file: str | PathLike)
```

The central class of NPLinker application.

Attributes:

- `config` (`Dynaconf`) – The configuration object for the current NPLinker application.
- `root_dir` (`str`) – The path to the root directory of the current NPLinker application.
- `output_dir` (`str`) – The path to the output directory of the current NPLinker application.
- `bgcs` (`list[BGC]`) – A list of all BGC objects.
- `gcfs` (`list[GCF]`) – A list of all GCF objects.
- `spectra` (`list[Spectrum]`) – A list of all Spectrum objects.
- `mfs` (`list[MolecularFamily]`) – A list of all MolecularFamily objects.
- `mibig_bgcs` (`list[BGC]`) – A list of all MiBIG BGC objects.
- `strains` (`StrainCollection`) – A StrainCollection object containing all Strain objects.
- `product_types` (`list[str]`) – A list of all BiGSCAPE product types.
- `scoring_methods` (`list[str]`) – A list of all valid scoring methods.

Parameters:

- `config_file` (`str | PathLike`) – Path to the configuration file to use.

Examples:

Starting the NPLinker application:

```
>>> from nplinker import NPLinker
>>> npl = NPLinker("path/to/config.toml")
```

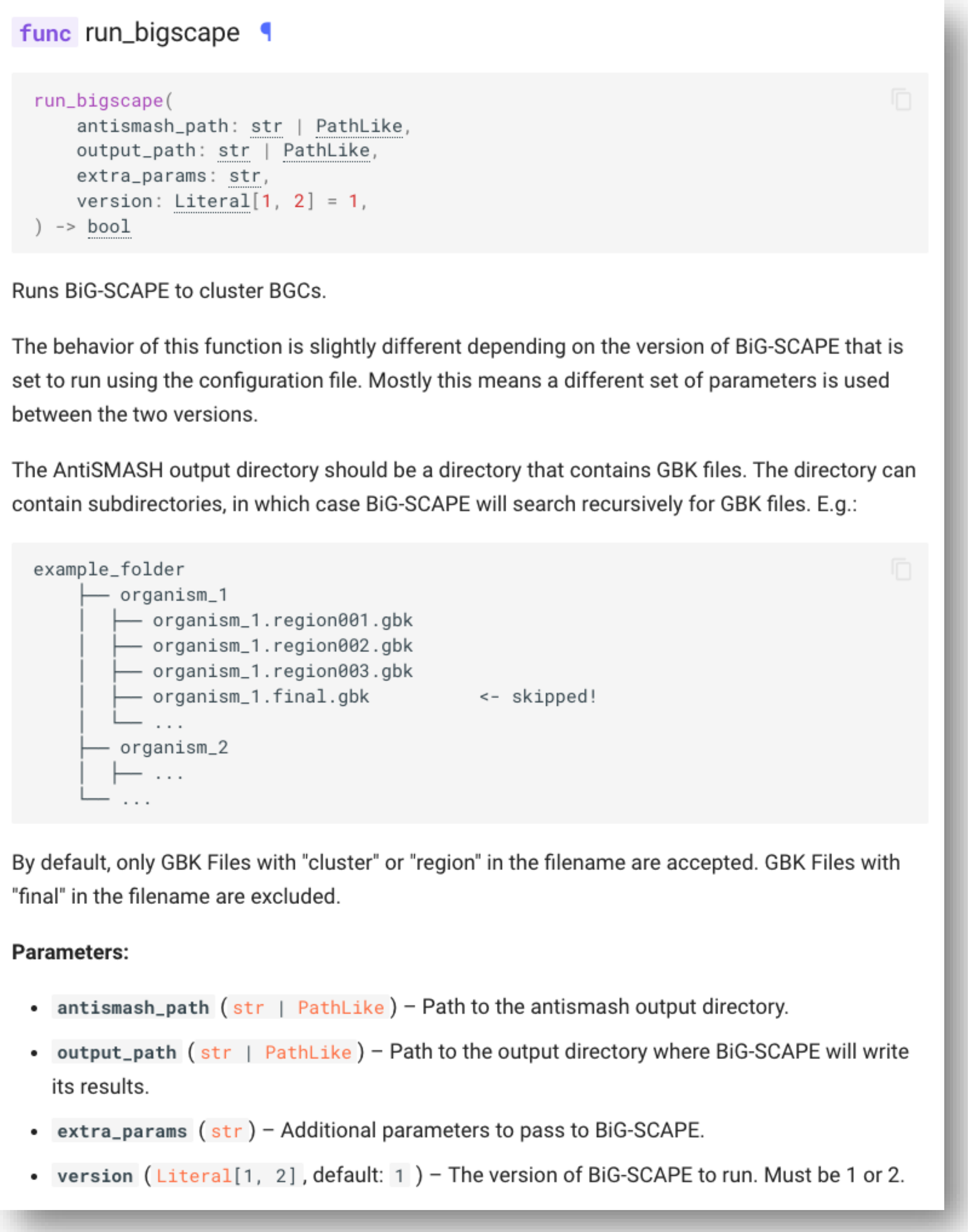
[PR 266]

netherlands eScience center

Integrated BigScape V2

Now you can run BigScape V1 and V2 in NPLinker

[PR 251 by Arjan]



func run_bigscape ¶

```
run_bigscape(
    antismash_path: str | PathLike,
    output_path: str | PathLike,
    extra_params: str,
    version: Literal[1, 2] = 1,
) -> bool
```

Runs BiG-SCAPE to cluster BGCs.

The behavior of this function is slightly different depending on the version of BiG-SCAPE that is set to run using the configuration file. Mostly this means a different set of parameters is used between the two versions.

The AntiSMASH output directory should be a directory that contains GBK files. The directory can contain subdirectories, in which case BiG-SCAPE will search recursively for GBK files. E.g.:

```
example_folder
    ├── organism_1
    │  ├── organism_1.region001.gbk
    │  ├── organism_1.region002.gbk
    │  ├── organism_1.region003.gbk
    │  ├── organism_1.final.gbk          <- skipped!
    │  └── ...
    ├── organism_2
    │  ├── ...
    └── ...
```

By default, only GBK Files with "cluster" or "region" in the filename are accepted. GBK Files with "final" in the filename are excluded.

Parameters:

- `antismash_path` (`str | PathLike`) – Path to the antismash output directory.
- `output_path` (`str | PathLike`) – Path to the output directory where BiG-SCAPE will write its results.
- `extra_params` (`str`) – Additional parameters to pass to BiG-SCAPE.
- `version` (`Literal[1, 2]`, default: `1`) – The version of BiG-SCAPE to run. Must be 1 or 2.

netherlands eScience center

NPLinker Webapp Development

Contributors: Giulia Crocioni, Cunliang Geng

Demo

netherlands eScience center

Monthly plan

What will happen before next meeting

netherlands eScience center

Need your tests and feedback on the refactored NPLinker

Questions? Ideas? Feedback?