Snk: A Snakemake CLI and Workflow Management System

Wytamma Wirth ⓞ [1], **Simon Mutch** ⓞ [2], **and Robert Turnbull** ⓞ [2]

1 Peter Doherty Institute for Infection and Immunity, University of Melbourne, Australia **2** Melbourne Data Analytics Platform, University of Melbourne, Melbourne 3010, Australia

Summary

Snk (pronounced "snek") is a workflow management tool designed to simplify the use of Snakemake workflows by dynamically generating command-line interfaces (CLIs). Snk allows complex Snakemake workflows to be used as modular components in larger systems that can be executed and managed from the command line with minimal overhead. This enables researchers and developers to integrate and manage sophisticated workflows seamlessly. Snk significantly improves the interoperability and accessibility of Snakemake workflows, making it easier to use and share computational pipelines in various research fields.

Statement of need

The integration of bioinformatic analyses into comprehensive pipelines (aka workflows) has revolutionised the field by improving the robustness and reproducibility of analyses. One of the most popular workflow frameworks is Snakemake (Mölder et al., 2021). Snakemake is a user-friendly and adaptable make-style workflow framework with a powerful specification language built atop of the Python programming language. Despite its success, Snakemake workflows are often developed for specific research analysis rather than as general-purpose reusable tools. That is, Snakemake workflows are typically built for the reproducibility of a single analysis but not necessarily built for flexibility.

To improve their utility, Snakemake workflows developers often encapsulate workflows within CLI tools by producing wrapper code to abstract the workflow execution, sometimes called workflows-as-applications or workflows packages (Roach et al., 2022). These wrappers serve as intermediaries between the end-user (via the CLI) and the workflow execution, enabling developers to tailor the Snakemake experience to specific use cases. For example, the pangolin CLI tool wraps a snakemake workflow for SARS-CoV-2 lineage assignment (O'Toole et al., 2022). Initiatives like Snaketool have simplified the development of Snakemake-based CLIs by offering a template for developers (Roach et al., 2022). Nonetheless, the onus remains on the developer to create and maintain the CLI wrappers for their workflow.

Here we present Snk, a Snakemake workflow management system that allows users to install Snakemake workflows as dynamically generated Command Line Interfaces. Thus users can create a CLI for their (or others') Snakemake workflows with minimal to no code changes required. The Snk-generated CLIs follow best practices and include several features out of the box that improve user experience. The CLIs can be configured at install time or via a snk.yaml configuration file. Snk is readily available for installation via PyPI and Conda, using the commands `pip install snk` and `conda install snk`, respectively.

Snk has two distinct major functions; managing the installation of workflows, and dynamical generating CLIs from Snakemake configuration files. To install a workflow as a CLI, users can

specify the file path, URL, or GitHub name (username/repo) of a workflow. Snk copies (clones) workflows into a managed directory structure, creates a CLI entry point, and optionally creates an isolated virtual environment for each workflow. Workflows can be installed from specific commits, tags, or branches, ensuring reproducibility. The advent of Snk allows users to utilise the Snakemake workflow catalog (https://snakemake.github.io/snakemake-workflow-catalog) as a searchable package index of Snk-installable Snakemake tools. The snk install command is flexible and can be used to install diverse workflows using installation options. For example, the dna-seq-gatk-variant-calling workflow (release tag v2.1.1) can be installed as a CLI named `variant-calling` with Snakemake v8.10.8 and Pandas and NumPy dependencies using the following command:

```
snk install \
    snakemake-workflows/dna-seq-gatk-variant-calling \
    --name variant-calling \
    --snakemake 8.10.8 \
    -d pandas==1.5.3 \
    -d numpy==1.26.4 \
    -t v2.1.1
```

The workflow will then be accessible via the `variant-calling` CLI in the terminal (Figure 1). Additionally, the snk command can be used to list and uninstall workflows installed with Snk. The complete documentation for managing workflows can be found at https://snk.wytamma.com/managing_workflows.

Figure 1: The `variant-calling` CLI generated by Snk.

The core functionality of Snk is the dynamic creation of CLIs. Internally snk uses the `Snk-CLI` sister package to generate the CLI. By default key values pairs of the Snakemake configfile are mapped to CLI option. For example, `samples: samples.tsv` in the configfile will generate a `--samples` option in the CLI with the default value `samples.tsv` (Figure 2). The CLI generated by snk is highly customisable and can be configured via a snk.yaml file placed in the workflow directory. The snk.yaml file can configure many aspects of CLI including subcommands, ASCII art, help messages, resource files, default values, and much more. Complete documentation for the Snk config file can be found at https://snk.wytamma.com/snk_config_file.

```
❯ variant-calling run --help

Usage: variant-calling run [OPTIONS]

Run the workflow.
All unrecognized arguments are passed onto Snakemake.
┌─ Options ──────────────────────────────────────────────────────────────────┐
  --config                       FILE      Path to snakemake config file. Overrides existing workflow configuration.
                                           [default: None]
  --resource        -r           PATH      Additional resources to copy from workflow directory at run time.
  --profile         -p           TEXT      Name of profile to use for configuring Snakemake. [default: None]
  --force           -f                     Force the execution of workflow regardless of already created output.
  --dry             -n                     Do not execute anything, and display what would be done.
  --lock            -l                     Lock the working directory.
  --dag             -d           PATH      Save directed acyclic graph to file. Must end in .pdf, .png or .svg
                                           [default: None]
  --cores           -c           INTEGER   Set the number of cores to use. If None will use all cores. [default: None]
  --no-conda                               Do not use conda environments.
  --keep-resources                         Keep resources after pipeline completes.
  --keep-snakemake                         Keep .snakemake folder after pipeline completes.
  --verbose         -v                     Run workflow in verbose mode.
  --help-snakemake  -hs                    Print the snakemake help and exit.
  --help            -h                     Show this message and exit.
└────────────────────────────────────────────────────────────────────────────┘
┌─ Workflow Configuration ───────────────────────────────────────────────────┐
  --samples                                                  TEXT     [default: config/samples.tsv]
  --units                                                    TEXT     [default: config/units.tsv]
  --ref-species                                              TEXT     [default: homo_sapiens]
  --ref-release                                              INTEGER  [default: 98]
  --ref-build                                                TEXT     [default: GRCh38]
  --filtering-vqsr              --no-filtering-vqsr                   [default: no-filtering-vqsr]
  --filtering-hard-snvs                                      TEXT     [default: QD < 2.0 || FS > 60.0 ||
                                                                      MQ < 40.0 || MQRankSum < -12.5 ||
                                                                      ReadPosRankSum < -8.0]
  --filtering-hard-indels                                    TEXT     [default: QD < 2.0 || FS > 200.0
                                                                      || ReadPosRankSum < -20.0]
  --processing-remove-duplicates --no-processing-remove-duplicates   [default:
                                                                      processing-remove-duplicates]
  --params-gatk-haplotypecaller                              TEXT
  --params-gatk-baserecalibrator                             TEXT
  --params-gatk-genotypegvcfs                                TEXT
  --params-gatk-variantrecalibrator                          TEXT
  --params-picard-markduplicates                             TEXT     [default: REMOVE_DUPLICATES=true]
  --params-trimmomatic-pe-trimmer                            TEXT     [default: LEADING:3, TRAILING:3,
                                                                      SLIDINGWINDOW:4:15, MINLEN:36]
  --params-trimmomatic-se-trimmer                            TEXT     [default: LEADING:3, TRAILING:3,
                                                                      SLIDINGWINDOW:4:15, MINLEN:36]
  --params-vep-plugins                                       TEXT     [default: LoFtool]
  --params-vep-extra                                         TEXT
└────────────────────────────────────────────────────────────────────────────┘
```

Figure 2: The run command of the `variant-calling` CLI dynamically generated from the Snakemake configfile. Several standard options are provided in the Options section, e.g., `--dry` (equivalent to Snakemakes `--dry-run`), `--dag` to create a DAG plot of the workflow, and `--cores` witch defaults to all. The Workflow Configuration section contains the options dynamically generated from the configfile. Snk-CLI automatically infers the defaults and types of the options and creates flags for boolean options.

Developers can also directly use the Snk-CLI package to generate CLIs for their Snakemake workflows. By using the CLI class from Snk-CLI workflow, developers can build a fully featured workflow package without having to write a Snakemake wrapper. We provide a guide for using Snk-CLI to build self-contained workflow packages at https://snk.wytamma.com/workflow_ packages. The Snk-CLI package is available via PYPI and can be installed using the command `pip install snk-cli`.

Snk is a powerful tool that simplifies the use of Snakemake workflows by dynamically generating CLIs. Snk is open-source software released under the MIT license. Snk documentation, source code, and issue tracker are available at https://github.com/Wytamma/snk. We welcome contributions and feedback from the community to improve Snk and make it a valuable tool for the Snakemake community and reproducible research at large.

Acknowledgements

Mölder, F., Jablonski, K. Ph., Letcher, B., Hall, M. B., Tomkins-Tinch, Christopher H, Sochat, V., Forster, J., Lee, S., Twardziok, S. O., Kanitz, A., Wilm, A., Holtgrewe, M., Rahmann, S., Nahnsen, S., & Köster, J. (2021). Sustainable data analysis with snakemake [version 2; peer review: 2 approved]. *F1000Research*, *10*(33). https://doi.org/10.12688/f1000research.29032.2

O'Toole, Á., Pybus, O. G., Abram, M. E., Kelly, E. J., & Rambaut, A. (2022). Pango lineage designation and assignment using SARS-CoV-2 spike gene nucleotide sequences. *BMC Genomics*, *23*(1), 121. https://doi.org/10.1186/s12864-022-08358-2

Roach, M. J., Pierce-Ward, N. T., Suchecki, R., Mallawaarachchi, V., Papudeshi, B., Handley, S. A., Brown, C. T., Watson-Haigh, N. S., & Edwards, R. A. (2022). Ten simple rules and a template for creating workflows-as-applications. *PLOS Computational Biology*, *18*(12), e1010705. https://doi.org/10.1371/journal.pcbi.1010705