bgrrl Documentation

Release 0.6

Christian Schudoma

Apr 04, 2019

Contents

bgrrl ("bgrrl", **B**acterial **G**enome **R**econstruction (aka *assembly*) and **R**ecognition (aka *annotation*) **Pipe**line) is a Python3-based workflow system for large-scale bacterial genomics studies. bgrrl is used by the Earlham Institute (EI) Core Bioinformatics group to deliver bacterial genome assembly and annotation projects as part of EI's national capability. bgrrl is a customisable, Snakemake-driven python application wrapping established and state-of-the-art bioinformatics software tools such as unicycler and prokka. The pipeline consists of individual modules for preprocessing, assembly, annotation, quality assurance, and project-specific finalisation and packaging. Individual steps and settings of the pipeline, such as choice of assembly software (currently supported: unicycler, spades, velvet-optimizer) or trimming/preprocessing parameters can be customised according to user preference or project requirements. bgrrl offers two choices of genome annotation - de novo via prokka and reference-based via ratt. Report generation and quality assurance of the produced assemblies and annotations is performed via the independent qaa ("kaa", **Q**uality **A**ssemblies and **A**nnotations) workflow system, which currently wraps quast, qualimap, busco, and blobtools for quality assessment, which is then collated into a multiqc report.

bgrrl has been designed to deal with large numbers of samples of varying size and quality. To ensure pipeline stability at runtime, we perform a set of pre-assembly checks on read quality, kmer-distribution and GC-content, and survey assemblies with tadpole to determine the assemble-ability of a sample ("survey-stage"). The pipeline is easy to operate and runs with minimal user interaction but also allows for revising the outcomes of each stage if required. The produced assemblies and annotations are complemented with comprehensive reports for each stage.

So far, bgrrl has been successfully applied to over 8000 samples distributed over five projects with different project requirements including a large part of the sequencing data generated by the 12K Salmonella Project between the University of Liverpool and Earlham Institute.

CHAPTER 1

Contents

1.1 Installation

1.1.1 Requirements

bgrrl is a Python3 application, so Python3.5+ is necessary.

Python dependencies

- snakemake >= 4.4.0
- drmaa
- sphinx
- qaa ()

Third party software

- bbmap
- fastqc
- kat
- unicycler/spades
- velvet-optimizer
- prokka
- emboss
- ratt

For qaa:

- quast
- blobtools
- qualimap
- busco
- multiqc ()
- picardtools
- samtools
- bwa and/or bowtie2

We provide Singularity recipes (one for bgrrl and one for qaa) that cover all the requirements.

If you plan to rely on your own installations, please make sure you have Python3-compatible versions, e.g. of quast and blobtools.

Other resources

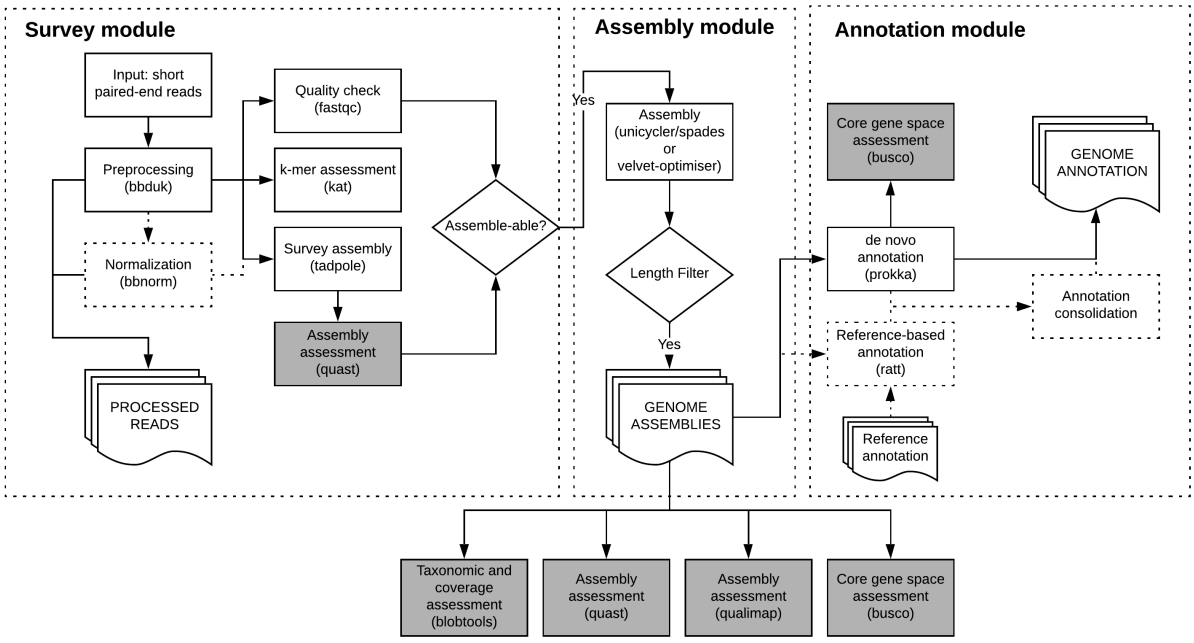
- bacteria.odb9 busco database

  http://busco.ezlab.org/v2/datasets/bacteria_odb9.tar.gz

- a comprehensive nucleotide sequence database (including taxonomy ids) for blobtools

  e.g. NCBI nt

1.2 Usage



1.2.1 Preparation of sample data

bgrrl takes as input a set of paired-end libraries, which need to be passed to the pipeline in form of a samplesheet. This samplesheet needs to be comma-separated and can be either generated manually (s. below for column format) or with the included `create_samplesheet` script. The script takes as argument a directory containing the fastq files (currently with mandatory `.fastq.gz` suffix) and writes the samplesheet to stdout.

Example usage

```
create_samplesheet <read_directory> > samplesheet.csv
```

For manual generation of the samplesheet, please create a comma-separated file following the column order below. Please note that items have to be present unless noted otherwise.

1. Sample ID
2. Sample Name (can be the same as Sample ID; the intention is to allow a more understandable sample reference in the future)
3. Full path to R1 file
4. Full path to R2 file
5. LEAVE EMPTY (intended use: Full path to single-end file)
6. LEAVE EMPTY (intended use: NCBI taxonomy id)
7. LEAVE EMPTY (intended use: Taxonomy name)
8. LEAVE EMPTY (intended use: FastQC report for raw R1)
9. LEAVE EMPTY (intended use: FastQC report for raw R2)
10. LEAVE EMPTY (intended use: FastQC report for raw single-end file)

Please note that despite columns 5-10 not being used, bgrrl still expects to see 10 columns at present.

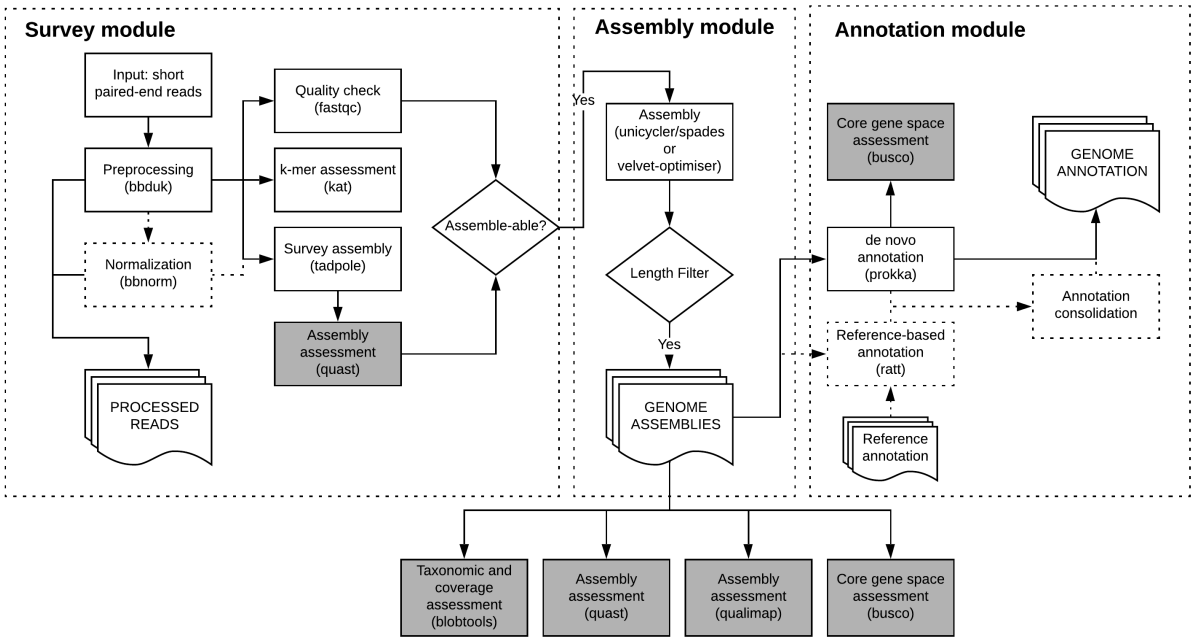
1.2.2 Preparation of the bgrrl run

bgrrl runs are driven by two configuration files, which need to be adapted to your computing environment. Templates are included in the `bgrrl/etc` directory in the bgrrl source directory. To copy these files to the current run you can use the `bginit` command. `bginit -o <outdir>` will automatically copy the files into the folder `<outdir>/config`. The copies can then be edited.

bgrrl_config.yaml

hpc_config.json

1.3 General Run Parameters



bgrrl consists of three main modules: `survey`, `assemble`, and `annotate`. Each module, or stage, has individual command line options. This section describes common command line options.

Each bgrrl run requires at the very least the following three command line parameters:

- `input_sheet`
- `--config`
- `--hpc_config`

1.3.1 Command line arguments

The command `bgrrl -h` or `bgrrl <stage> -h` (or `--help` instead of `-h`) will display a list of command line options.

The first block of command line options is stage-specific and will be discussed in the description of each stage.

bgrrl options:

- `input_sheet`

  This is the path to a samplesheet, a comma-separated file, containing location and meta-information for each sample/library (REQUIRED!).

- `-o OUTPUT_DIR, --output-dir OUTPUT_DIR`

  Output will be written to the specified directory [Analysis].

- `--project-prefix PROJECT_PREFIX`

  Reports and data packages will be prefixed by this string []

- `--config CONFIG`

  Path to configuration file. This file specifies software configurations, resource locations and general project metadata (REQUIRED!).

- `--report-only`

  With this option, bgrrl Only (re-)runs reporting modules. No snakemake pipelines are called.

- `-f, --force`

  Force overwriting existing output directory, causes pipeline to be restarted. (CURRENTLY DISABLED)

- `--enterobase-groups ENTEROBASE_GROUPS`

  Comma-separated list of Enterobase microbial organisms. The set of assemblies is tested against organism-specific criteria and assemblies are packaged according to their species. [NEEDS REWORDING!]. By default, the enterobase mode is disabled.

HPC Options:

Controls for how jobs should behave across the HPC resources.

- `--partition PARTITION`

  Will run all child jobs on this partition/queue, this setting overrides anything specified in the `--hpc_config` file.

- `--scheduler SCHEDULER`

  The job scheduler to use. LSF, PBS and SLURM are currently supported. If running without a scheduler type NONE here. Assumes SLURM by default.

- `--no_drmaa`

  Use this flag if DRMAA is not available.

- `-N MAX_NODES, --max_nodes MAX_NODES`

  Maximum number of nodes to use concurrently

- `-c MAX_CORES, --max_cores MAX_CORES`

  Maximum number of cores to use concurrently

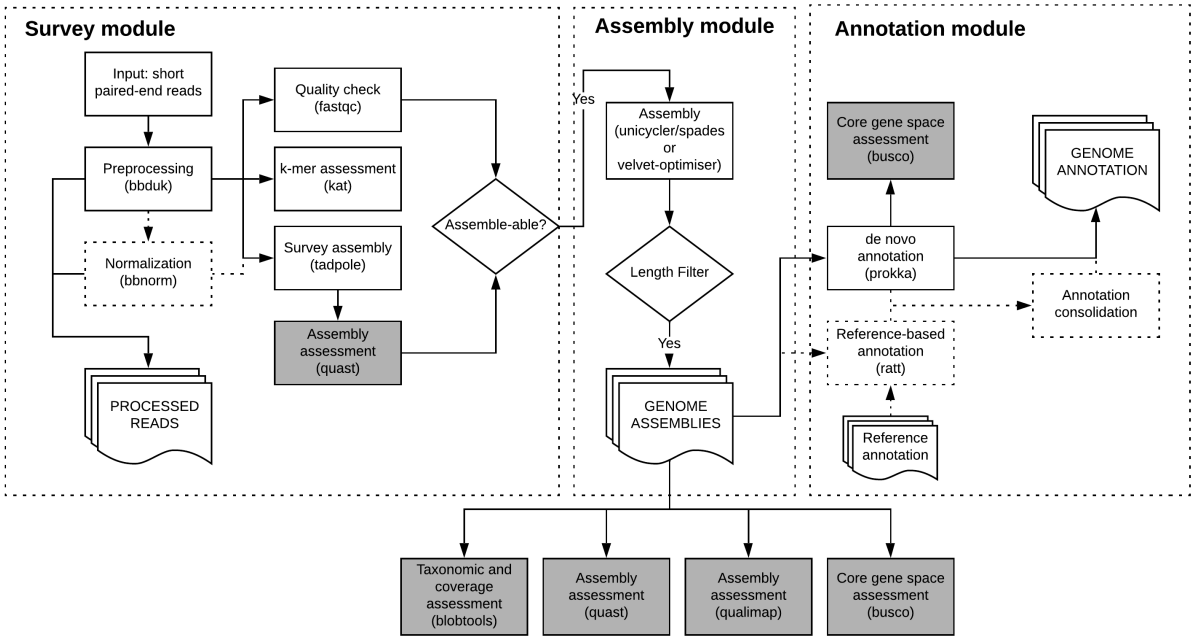
- `--hpc_config HPC_CONFIG`

  Configuration file for the HPC. Can be used to override what resources and partitions each job uses. (REQUIRED!)

- `--unlock`

  If the snakemake pipeline is not running because it is reporting that the directory is locked, then you can unlock it using this option. Please make sure that there are no other snakemake jobs are running in this directory before using this option!

1.4 The survey module



The survey module has two main functions. 1) Preprocessing of the library data and 2) assessment of the assemble-ability of each sample/library.

In detail, the steps performed by the survey module are:

1. Preprocessing with `bbduk`

   Each paired-end library is preprocessed with bbduk with the following operations:

   - k-mer based adapter trimming against `bbduk`'s default adapter list
   - gentle quality trimming on either side, keeping only bases with at least phred=3
   - length filtering (100bp)
   - quality filtering (maq=20)
   - tpe/tbo (s. `bbduk` documentation)

   These operations can be adjusted via the `bgrrl_config.yaml`.

2. Normalization with `bbnorm` (optional, switch off with `--no-normalization`)

   Libraries are normalized to the range of 2x-100x.

   The range can be adjusted via the `bgrrl_config.yaml`.

3. Read quality assessment with `fastqc`

4. Read sequence feature assessment with `kat`

5. Survey assembly with `tadpole`

   Each library is subjected to a simple and quick assembly with `tadpole`. This determines if the library can be assembled or not. Library assembly statistics are then calculated with `quast` via `qaa`.

6. Filtering by assemble-ability

   Libraries that could be assembled with `tadpole` and that contain at least 1000 reads and have an assembly size above 1Mbp (or user-specified, s. below) will be automatically passed on to the assembly stage. Libraries that fail these checks will be filtered out. However, the user can manually add them to the assembly samplesheet (s. below).

7. Assembly samplesheet generation

   Samples that passed the filtering stage will be written to a new samplesheet (`samplesheet.qc_pass.csv`) in the `<outdir>/reports/samplesheets` directory. This samplesheet can then be used as input for the assembly stage.

8. Read packaging

   Preprocessed reads will be automatically packaged into the `<outdir>/Data_Package` directory unless suppressed by the `--no-packaging` option.

1.4.1 Command line arguments

The command `bgrrl -h` or `bgrrl <stage> -h` (or `--help` instead of `-h`) will display a list of command line options.

Remember each bgrrl run requires at the very least the following three command line parameters:

- `input_sheet`
- `--config`
- `--hpc_config`

survey options:

- `--no-normalization`

  Disable read normalization. [False]

- `--no-packaging`

  Disable automatic packaging. [False]

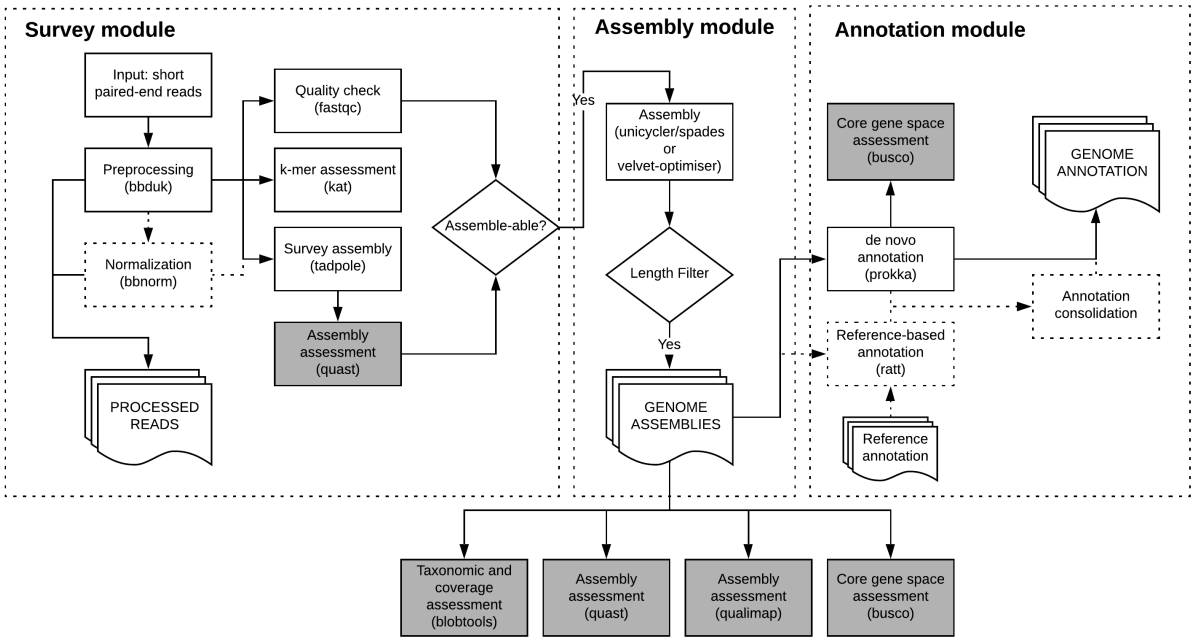
- `--full-qaa-analysis`

  Perform full qaa-analysis on survey assemblies. [False]

- `--minimum-survey-assembly-size MINIMUM_SURVEY_ASSEMBLY_SIZE`

  Minimum size (in bp) for tadpole assembly to pass survey stage [1Mbp] Setting this option to a smaller size allows plasmid-specific libraries to be processed.

1.5 The assemble module



The assemble module performs automatically optimized assemblies of the provided library data.

1.5.1 Sample sheet preparation

If the user has previously run the `survey` module, the resulting samplesheet `samplesheet.qc_pass.tsv` can be used directly to drive the assemble module.

Otherwise, the user should prepare a comma-separated samplesheet following the column order below.

1. Sample ID
2. Sample Name (can be the same as Sample ID; the intention is to allow a more understandable sample reference in the future)
3. Full path to R1 file
4. Full path to R2 file
5. LEAVE EMPTY (intended use: Full path to single-end file)
6. LEAVE EMPTY (intended use: NCBI taxonomy id)
7. LEAVE EMPTY (intended use: Taxonomy name)
8. LEAVE EMPTY (intended use: FastQC report for raw R1)
9. LEAVE EMPTY (intended use: FastQC report for raw R2)
10. LEAVE EMPTY (intended use: FastQC report for raw single-end file)
11. Full path to preprocessed, non-normalized R1 file
12. Full path to preprocessed, non-normalized R2 file
13. LEAVE EMPTY (intended use: Full path to preprocessed, non-normalized single-end file)

14. Full path to preprocessed, normalized R1 file
15. Full path to preprocessed, normalized R2 file
16. LEAVE EMPTY (intended use: Full path to preprocessed, normalized single-end file)

1.5.2 Command line arguments

The command `bgrrl -h` or `bgrrl <stage> -h` (or `--help` instead of `-h`) will display a list of command line options.

Remember each bgrrl run requires at the very least the following three command line parameters:

- `input_sheet`
- `--config`
- `--hpc_config`

assemble options:

- `--assembler {unicycler,velvet}`

  Assembly software to use for genome assembly. [unicycler]

- `--contig-minlen CONTIG_MINLEN`

  Minimum length [bp] of contigs retained in filtering step [0].

- `--no-normalization`

  Use non-normalized reads in assemble module [False]

- `--run-annotation`

  Run annotation on assembly. If set, de novo annotation with prokka will be run. Additionally, you may enable annotation transfer by specifying a path to a reference annotation with `--ratt-reference`. [False]

- `--custom-prokka-proteins CUSTOM_PROKKA_PROTEINS`

  If you have a custom protein database that you would like prokka to use (`prokka`'s `--proteins` option), then specify the path to it here. [n/a]

- `--ratt-reference RATT_REFERENCE`

  Path to reference data for ratt annotation transfer

- `--is-final-step`

  If set, analysis packaging will take place after the assembly stage. Otherwise, assume that an annotation stage will follow, which will then take care of analysis packaging. [False]

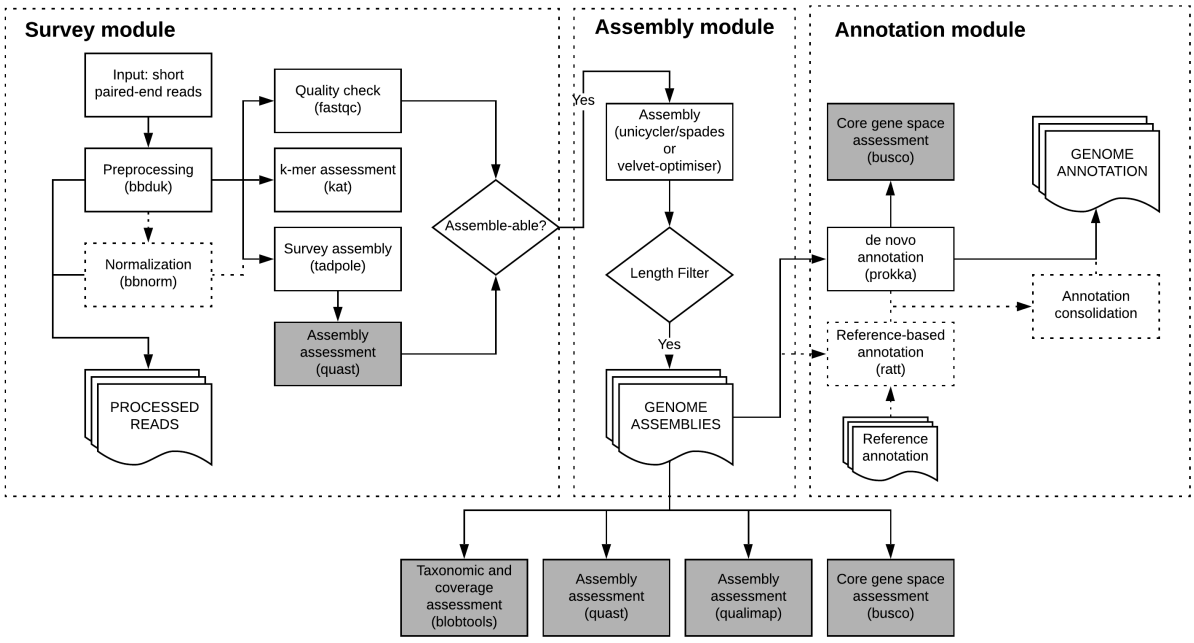
- `--no-packaging`

  Disable automatic packaging. [False]

- `--prokka-package-style {by_sample,all_in_one}`

  Should the prokka annotation be packaged into one directory per sample (by_sample) or into one single directory (all_in_one)? [by_sample]

1.6 The annotate module



The annotate performs de novo, and optionally reference-based, genome annotation on a set of assemblies.

1.6.1 Sample sheet preparation

If the user has previously run the `assemble` module without annotation, the resulting samplesheet `samplesheet.asm_pass.tsv` can be used directly to drive the annotate module.

Otherwise, the user should prepare a comma-separated samplesheet following the column order below.

1. Sample ID
2. Sample Name (can be the same as Sample ID; the intention is to allow a more understandable sample reference in the future)
3. Full path to assembly file

TODO: CHECK

1.6.2 Command line arguments

The command `bgrrl -h` or `bgrrl <stage> -h` (or `--help` instead of `-h`) will display a list of command line options.

Remember each bgrrl run requires at the very least the following three command line parameters:

- `input_sheet`
- `--config`
- `--hpc_config`

annotate options:

- `--custom-prokka-proteins CUSTOM_PROKKA_PROTEINS`

  If you have a custom protein database that you would like prokka to use (`prokka`'s `--proteins` option), then specify the path to it here. [n/a]

- `--ratt-reference RATT_REFERENCE`

  Path to reference data for ratt annotation transfer

- `--no-packaging`

  Disable automatic packaging. [False]

- `--prokka-package-style {by_sample,all_in_one}`

  Should the prokka annotation be packaged into one directory per sample (by_sample) or into one single directory (all_in_one)? [by_sample]