

Changelog

3.12.2

- Updated documentation

3.12.1

No user-facing changes

3.12.0

- Update to Scala 3.3.1

3.11.0

- Machine-parseable condition barcode summary file

3.10.0

- More efficient and memory-safe sampling technique for unexpected sequence reporting

3.9.0

- Use sampling technique for generating unexpected sequence reports

3.8.0

- Replace kantan.csv with scala-csv

3.7.2

- Parse barcodes from read IDs in demultiplexed mode

3.7.1

- Adjust handling of command-line arguments in demultiplexed FASTQ file case

3.7.0

- Support for processing demultiplexed FASTQ files

3.6.4

No user-facing changes

3.6.3

- Documentation for processing demultiplexed FASTQ files

3.6.2

- Additional validation of FASTQ records to detect badly concatenated files
- Documentation for read ID check policy

3.6.1

- Add a specialized template barcode policy to optimize a common use case, where a template matches 2 barcode regions, each of which has a short prefix, separated by a fixed length run of arbitrary barcodes.

3.6.0

- Support reading multiple reads files sequentially as part of a single run

3.5.0

Released as open source under a BSD 3-Clause license

3.4.4

No user-facing changes

3.4.3

Improvements

- Report empty conditions as “Unlabeled Sample Barcodes”

3.4.2

Improvements

- Add support for different read ID checking policies, including one specific to Illumina which ignores divergent material after the first space in the ID line. ## Bugfixes
- Corrected fixed barcode prefix policies to allow the ‘FIXED@’ syntax, per the documentation; previous versions required the syntax ‘FIXED:’. For backwards compatibility, the former syntax is still supported but in future versions this may be removed.

3.4.1

No user-facing changes

3.4.0

Improvements

- Added support for reading row barcodes from paired-end sequencing data

3.3.4

Improvements

- Improved error message in the case where reads are too short

3.3.3

No user-facing changes

3.3.2

No user-facing changes

3.3.1

Improvements

- Advanced setting `--always-count-col-barcodes` (disabled by default). When enabled, PoolQ counts column barcodes from reads even if a row barcode is not found, which changes the per column barcode metrics `Matched Sample Barcode` and `% Match`, reported in the quality file.

3.3.0

Improvements

- Support UMI barcodes

3.2.11

Improvements

- Allow empty barcode IDs in row and column reference files

3.2.10

No user-facing changes

3.2.9

Improvements

- Updated manual

3.2.8

No user-facing changes

3.2.7

No user-facing changes

3.2.6

No user-facing changes

3.2.5

No user-facing changes

3.2.4

No user-facing changes

3.2.3

Bugfixes

- Correctly skip writing correlation file when there is only a single construct barcode or a single experimental condition

3.2.1

Improvements

- Added whitespace to run info file for readability

3.2.0

Improvements

- Add `Normalized Match` column to quality file
- Change quality file header from `Read counts for sample barcodes without associated conditions:` to `Read counts for most common sample barcodes without associated conditions:`
- Change terminology from `scores` to `counts`
- Log version and run configuration in run info file

- Add changelog

3.1.1

Bugfixes

- Correctly skip writing correlation file when there is only a single experimental condition

3.1.0

No user-facing changes

3.0.5

Bugfixes

- Fixed path to PoolQ jar in `Makefile` and scripts distributed with PoolQ
- ### Improvements
- Added regression tests for barcode counts and log-normalized counts files

3.0.4

Bugfixes

- Fixed delimiter in `Construct IDs` field of log-normalized counts files

3.0.3

No user-facing changes

3.0.2

No user-facing changes

3.0.1

Improvements

- Made 5' search limit optional when specifying a row barcode search template

3.0.0

- Complete re-write of PoolQ