PoolQ

Mark Tomko, John Sullivan, Shuba Gopal. 2011.02 - 2013.02

This documentation was last updated for PoolQ version 1.0.0 (02/25/2013).

PoolQ is a counter for indexed samples from next-gen sequencing of pooled DNA.

Background

The Broad Institute RNAi Platform uses Solexa sequencing to tally the results of pooled RNAi screens. We perform the following steps to generate one or more FASTQ reads files for the pooled screen:

1. We start with multiple genomic DNA samples (conditions) - gathered either from different pools, or from the same pool at different time points.
2. We apply PCR to the genomic DNA with primers that are designed to attach to the genomic DNA within the vector sequence at a fixed distance from the start of the hairpin sequence. The primer contains a fixed-length DNA barcode that is unique to the genomic DNA sample. In this way, all the amplification products will contain the inserted barcode.
3. We mix all the samples, normalized to equalize barcode representation, and run them in a single Solexa sequencing lane.
4. The Solexa sequencing process generates a FASTQ file for each sequencing lane.

PoolQ comes into play after the FASTQ file has been generated. A single PoolQ run processes one FASTQ file. It attempts to parse out the barcode and the hairpin sequence from each read. It maps barcodes to conditions and hairpins to clone IDs, and generates a matrix with the conditions as columns, and the clones as rows.

PoolQ Inputs

PoolQ requires 3 input files to run: the (FASTQ or SAM/BAM) file containing the reads (the reads file); a file mapping barcodes to conditions (the conditions file); and a file mapping hairpin sequences to hairpin IDs (the reference file). PoolQ can also take an optional input file (the platform reference file) describing known hairpins that are not expected to be present in the sequencing reads. There are also a variety of optional settings that specify the details of how PoolQ processes the reads.

The Reads File

The *reads file* is a either standard FASTQ file, or a BAM file, containing the reads. It may or may not be compressed with gzip. If the file is a FASTQ file, then the file extension should be `.fastq` or `.txt` (case-insensitive). If it is a BAM file, then the file extension should be `.bam` (case-insensitive). Files that do not follow these naming conventions can still be used, but you must specify the file type explicitly.

If the file is compressed with gzip, then the file must have an extra `.gz` extension after the `.fastq`, `.txt`, or `.bam` extension.

A barcode and a hairpin are parsed out of each read. The length of the barcode is inferred from the contents of the conditions file. The length of the hairpin is inferred from the contents of the hairpin file. By default, the barcode starts at the first base of the read, and the hairpin starts at the 17th base of the read, but both of these defaults can be overridden with an optional flag.

Every read in the file must have the same length. The read length must be long enough to contain both a the barcode and the hairpin.

The Conditions File

The conditions file maps barcodes to conditions:

- It is a file with two columns
- The first column contains the barcodes, and the second column contains the condition descriptions
- The columns can be separated by either commas or tabs
- The file may not include any column headers or any extra columns
- Barcodes must contain only A, T, C or G
- A barcode cannot occur more than once in the file
- Every barcode in the conditions file must have the same length
- You can have multiple barcodes mapping to the same condition, but be aware that if you do so, the reads for those barcodes will be counted together, resulting in a single column in the scores file for that condition

The Reference File

The *reference file* maps from hairpins to hairpin IDs:

- It is a file with two columns
- The first column contains the hairpin sequences, and the second column contains the hairpin IDs
- The columns can be separated by either commas or tabs
- The file may not include any column headers or any extra columns
- Hairpin sequences must be at least 20 bases in length, and must contain only A, T, C or G
- A hairpin ID cannot occur more than once in the file
- Every hairpin in the reference file must have the same length
- You can have multiple hairpin IDs mapping to the same hairpin sequence; the scores file will report the scores for the hairpin sequence alongside a comma-separated list of associated hairpin IDs.

The Platform Reference File

The *platform reference file* is an optional input file whose format is identical to that of the reference file. It consists of a master list of known hairpin sequences and their hairpin IDs. This file is used to provide hairpin IDs for hairpins encountered during the PoolQ run that were not expected to occur. A hairpin is expected to occur only if it is present in the reference file.

Include Non PF Reads

This is an optional input flag that, when present, indicates that PoolQ should include reads from a BAM file that fail the purity filter quality control check. This flag has no effect on the PoolQ results for FASTQ files, since the annotation is not present in FASTQ files. For more information on this flag, please see the documentation for Picard and SAMtools.

Minimum Hairpin Length

This is an optional input flag that, when present, specifies the minimum allowable length for a hairpin.

Barcode Start Index

This is an optional input flag that, when present, specifies the index of the first base of the barcode within the read. This is a 0-based index, which means that the first base of the read has index 0, the second base of the read has index 1, etc.

Hairpin Start Index

This is an optional input flag that, when present, specifies the index of the first base of the hairpin within the read. This is a 0-based index, which means that the first base of the read has index 0, the second base of the read has index 1, etc.

Unexpected Sequence Threshold

This is an optional input flag that, when present, specifies the minimum number of reads per 10,000,000 that a hairpin sequence must appear before it is included in the unexpected sequence file. The default is 5000 reads per 10,000,000, or 0.05%.

Exact Match

This is an optional input flag that, when present, indicates that fuzzy matching of hairpins found in reads to the hairpins in the reference file should be disabled. Only hairpins that exactly match a hairpin in the reference file will be counted. The default behavior (when this flag is not present) is to allow single base mismatches when matching hairpins to the reference file.

Include Ambiguous

This is an optional input flag that, when present, controls the handling of hairpins encountered in reads that fuzzy-match to more than one hairpin in the reference file. The default behavior is to discard ambiguously matching reads, so they will not be included in the scores file. If this behavior is not desired, specify this flag and all ambiguous reads will be counted for every possible matching hairpin. As a consequence of this behavior, the sum of the scores for a given column may add up to more than the number of reads for a particular condition.

Reads File Type

This is an optional input flag that, when present, specifies how PoolQ should treat the reads file type. By default, PoolQ will attempt to guess whether the reads file is a FASTQ, BAM, or text file based on the file name. If the filename is misleading, you can specify the file type explicitly using this flag. Valid values include BAM, FASTQ, and RAW (for plain text).

Skip Short Reads

This is an optional input flag that, when present, specifies that PoolQ should simply ignore reads that are too short to contain a barcode and hairpin. By default, PoolQ considers reads files with short hairpins to be badly formed and exits. By specifying this flag, you indicate that PoolQ should simply skip these short reads; a count of the number of skipped short reads will be available in the quality file.

PoolQ Outputs

PoolQ generates output files representing the matrix of read counts (or scores) for expected sequences, a report of read counts for unexpected sequences, and a report containing simple metrics used to help assess the overall quality of the sequencing data. There are two optional output files that contain alternative representations of the scores matrix. One contains the scores in log normalized form and the other contains read counts by barcode rather than by condition.

The Scores File

The *scores file* is a text file that contains a simple matrix of the read counts. The columns of the matrix represent the experimental conditions, and the rows of the matrix correspond the hairpin sequences. The individual values in each row are separated by tabs.

If you plan on loading the scores file into a spreadsheet application such as Excel, then we recommend using a file extension, such as `.txt`, that your spreadsheet application will recognize as being a text file. When opening the file in Excel, you will probably be prompted with a dialog asking you to describe the structure of the file. In the section about separator options, be sure that the checkbox for "Tab" is selected.

The Scores File in GCT Format

PoolQ can also produce the scores file in GCT format. This format is required for upload into GENE-E to perform a RIGER type analysis. Simply choose a '.gct' or '.GCT' file extension when selecting the name of the file.

The Quality Report

The *quality report* is a simple text file containing some extra information gathered during the PoolQ run. The information reported here is intended to help you assess the quality of your data, and spot problems such as an unacceptably high frequency of uncounted reads, or mistakes in barcode tracking. We currently report:

- The total number of reads
- The total number of reads that were successfully counted
- Out of the counted reads, the total number of single base mismatches to the hairpin
- Out of the counted reads, the percent that matched to both a known barcode and a known hairpin
- The average frequency of unknown barcode sequences
- The log-normalized frequency of unknown barcode sequences
- For each barcode mapped to a condition, we report:
    - the barcode
    - the condition
    - the total number of reads matching the barcode plus an expected hairpin
    - the total number of reads matching the barcode
    - the percent of the reads for the barcode that matched an expected hairpin
    - the log normalized number of matches
- For each barcode not mapped to a condition, we report the barcode and the total number of reads
- For hairpins mapping to multiple hairpin IDs, we report the hairpins and the hairpin IDs they map to. A hairpin can map to multiple hairpin IDs for two reasons:
- The hairpin may have occurred twice in the reference file
- If the hairpin sequences are truncated in the reads in the reads file, then two otherwise unique hairpins could end up being the same after truncation

The Barcode Scores File

The barcode scores file has a similar format to the scores file, except that the columns in the matrix represent the read counts for individual DNA barcodes rather than for experimental conditions. If, based on the quality file, a particular PCR appear to have been of low quality, it is possible to reaggregate scores by condition by excluding the scores from the barcode corresponding to the failed PCR. The barcode scores file is an optional output intended to provide support for loading PoolQ data into the RNAi Informatics database. However, it is available for any consumer of PoolQ data.

The Log Normalized Scores File

The log normalized scores file has the same format as the scores file, but every score is normalized according to the following procedure:

1. Take the raw read count for the hairpin ID and the condition
2. Divide by the total number of reads for that condition that matched a hairpin found in a reference file
3. Multiply by a constant factor of 1 million
4. Add one
5. Take the log base 2

The Unexpected Sequence File

The unexpected sequence file contains a report that describes briefly the collection of hairpin sequences found during the run. It is an optional output. The report contains two sections.

The first section represents a table whose rows correspond to unexpected hairpin sequences and whose columns indicate the number of times each hairpin sequence was found for each barcode. An additional column lists the hairpin IDs for these hairpin sequences, if the IDs are known. These hairpin IDs can be provided to the PoolQ tool via the platform reference file, described above.

The second section describes unexpected barcodes and the number of times an unexpected sequence appeared with each unexpected barcode. The unexpected barcodes are listed in descending order of the number of occurrences.

Running PoolQ

There are two different ways you can run PoolQ: using the Graphical User Interface (GUI), or using the Command Line Interface (CLI). But before you can run it, you need to download the zip file and unzip it.

Prerequisites

PoolQ is built for Java 6. To run PoolQ, you will need a JRE or JDK for version 6 or later. PoolQ has not been tested with Java 7, and the packaged jars may not run with Java 5. You can download an appropriate JRE or JDK from Oracle at:

> http://www.oracle.com/technetwork/java/javase/downloads/index.html

Downloading and Unzipping PoolQ

You can download PoolQ from an as yet undetermined location. The file you download is a ZIP file that you will need to unzip. In most cases, this is as simple as right-clicking on the zip file, and selecting something like "extract contents" from the popup menu. This will create a new folder on your computer named `poolq-1.0.0`, with the following contents:

- `poolq.jar`
- `poolq-cli.bat`
- `poolq-cli.sh`
- `poolq-gui.bat`
- `poolq-gui.sh`

Feel free to rename the folder, and to move it to wherever you want. Be aware, however, that the `.sh` and `.bat` files will only function properly if they can find the `poolq.jar` file in the same folder.

Recommended JVM Settings

The Java virtual machine (JVM) runs in "client mode" by default. This is a collection of settings optimized for applications run interactively by a user and that require very little memory (RAM). These default settings are not suitable for PoolQ, which is capable of processing very large files, which requires more memory than is available to the client mode JVM.

We recommend the following JVM settings be provided when running PoolQ:

- `-Xmx2048M`
- `-Xms2048M`
- `-server`
- `-XX:+UseStringCache`
- `-XX:+UseFastAccessorMethods`
- `-XX:+UseCompressedOops`
- `-XX:+UseGCOverheadLimit`
- `-XX:+UseParNewGC`
- `-XX:+UseConcMarkSweepGC`

This document contains a number of example command-lines for running PoolQ; however, we only list the full JVM options once, since typing the full command becomes unwieldy and the JVM options distract somewhat from the command-line arguments that are passed to PoolQ itself. You can copy and paste the full Java command from here:

```
java -Xmx2048M -Xms2048M -server -XX:+UseStringCache -XX:+UseFastAccessorMethods -
XX:+UseCompressedOops -XX:+UseGCOverheadLimit -XX:+UseParNewGC -XX:+UseConcMarkSweepGC
```

You will still need to add a classpath (`-cp`) argument as well as the name of the main class you wish to run (either `org.broadinstitute.rnai.poolq.gui.PoolQGui` or `org.broadinstitute.rnai.poolq.cli.PoolQCli`).

Running the PoolQ GUI

There are multiple ways you can run the PoolQ GUI. From simplest to most complex:

- Double-click on the `poolq.jar` file
- If you are on a Windows machine, double-click on the `poolq-gui.bat` file
- If you are on a UNIX-based machine such as Linux or Mac, open a terminal window, change directories to the `poolq-1.0.0` directory, and run

  ```
  ./poolq-gui.sh
  ```

- On any platform, open a terminal window, change directories to the `poolq-1.0.0` directory, and run:

  ```
  java -cp poolq.jar org.broadinstitute.rnai.poolq.gui.PoolQGui
  ```

If you successfully launched the PoolQ GUI, you should see a window prompting you to specify names and locations for 5 files: the 3 input files, the three optional flags, and the 3 output files discussed above. Once you select the run parameters, the "Perform Analysis" button will become enabled. Click that button to start the analysis.

Running the PoolQ CLI

You can run the PoolQ CLI from any Windows, Mac, or Linux machine, but it requires some understanding about how to launch programs from the command line on your given operating system. If this seems daunting to you, just use the PoolQ GUI. There is nothing you can do with the PoolQ CLI that you cannot do with the PoolQ GUI.

1. Open a terminal window for your operating system
2. Change directories to the `poolq-1.0.0` directory
3. On Windows, run:

   ```
   poolq-cli.bat
   ```

- Or, on a UNIX-based machine, run:

  ```
  ./poolq-cli.sh
  ```

- Or, on any machine, run:

  ```
  java -cp poolq.jar org.broadinstitute.rnai.poolq.cli.PoolQCli
  ```

If you successfully launched the PoolQ CLI, you should see a usage message that looks something like this:

```
  PoolQ version 1.0.0
usage: <program> <args>
    --barcode-scores <arg>                An optional output CSV file
                                          with the barcodes as columns,
                                          the hairpins as rows, and the
                                          read counts scores.
    --barcode-start <arg>                 The index of the start of a
                                          barcode within a read.
                                          Defaults to 0.
    --conditions <arg>                    An input file with two
                                          columns: the barcode, and the
                                          condition.
    --exact-match                         Use exact matches for hairpins
                                          in the reference file
    --hairpin-start <arg>                 The index of the start of a
                                          hairpin within a read.
                                          Defaults to 16.
    --help                                Prints usage message and
                                          exits.
    --include-ambiguous                   Score reads that match
                                          ambiguously to all matching
                                          hairpins
    --include-non-pf                      Include non-PF reads [SAM or
                                          BAM files only]
    --min-hairpin-length <arg>            The minimum allowable hairpin
                                          length. Defaults to 20.
    --norm-scores <arg>                   An optional output CSV file
                                          with the conditions as
```

```
                                        columns, the hairpins as rows,
                                        and the read counts as the log
                                        normalized scores.
    --platform-reference <arg>          An optional input file with
                                        two columns: a hairpin 21mer
                                        that is known to exist, and
                                        the associated hairpin ID.
    --quality <arg>                     An output text file containing
                                        a basic report of the quality
                                        control information gathered
                                        while processing the reads.
    --reads <arg>                       The file containing the
                                        sequencing reads in either
                                        FASTQ or BAM format. The file
                                        may be gzipped or not; if the
                                        file is gzipped, it should end
                                        with the .gz suffix.
    --reads-file-type <arg>             Override the reads file type.
                                        One of [ BAM, FASTQ, RAW ].
    --reference <arg>                   An input file with two
                                        columns: the hairpin 21mer
                                        that are contained in the
                                        reads, and the Hairpin IDs.
    --scores <arg>                      An output CSV file with the
                                        conditions as columns, the
                                        hairpins as rows, and the read
                                        counts as the scores.
    --skip-short-reads                  Skip reads too short to
                                        contain a barcode and hairpin.
                                        Defaults to false.
    --unexpected-sequence-threshold <arg>   The minimum number of reads
                                        per 10 million that need to
                                        contain an unexpected sequence
                                        before it is included in the
                                        unexpected sequence reference
                                        file. The default value is
                                        5000 or 0.05%.
    --unexpected-sequences <arg>        An optional output text file
                                        containing a report of
                                        sequences found in the reads
                                        but not mapped to hairpin IDs
                                        by the reference file. If a
                                        platform reference file is
                                        provided,any hairpins
                                        contained there will be
                                        identified by hairpin IDs in
                                        this file.
    --version                           Prints usage message and
                                        exits.
```

At this point, you are ready to run the PoolQ CLI for real, supplying file names and locations for the 3 file inputs and 2 or 3 file outputs. For example:

- On Windows, run:

  ```
  poolq-cli.bat --reads reads.txt --conditions conditions.txt --reference reference.txt
  --scores scores.txt --quality quality.txt
  ```

- Or, on a UNIX-based machine, run:

  ```
  ./poolq-cli.sh --reads reads.txt --conditions conditions.txt --reference reference.txt
  --scores scores.txt --quality quality.txt
  ```

- Or, on any machine, run:

  ```
  java -cp poolq.jar org.broadinstitute.rnai.poolq.cli.PoolQCli --reads reads.txt --
  conditions conditions.txt --reference reference.txt --scores scores.txt --quality
  quality.txt
  ```

The Scoring Algorithm

The *reads file* contains the sequencing reads. PoolQ supports any of the following formats:

- FASTQ (including Solexa/Illumina variant)
- SAM
- BAM
- Plain text (one read per line)

PoolQ currently ignores any read sequence content besides the barcode and the hairpin. In the future, we may check this sequence to help confirm the quality of the read.

Most often, the entire hairpin is included in the read. However, for files with very short read lengths the hairpin sequence may be truncated. In the case of truncated hairpin sequences, PoolQ will attempt to match based on the available nucleotide prefix.

If the read does not have a barcode that is an exact match to a barcode found in the conditions file, then the line is not counted, except in the section of the quality report devoted to counting reads for barcodes not found in the conditions file.

Counting Reads that Match a Barcode

The PoolQ scoring algorithm always attempts to match hairpins exactly to to one of the sequences provided in the reference file first. If an exact match is found, then only the exact match is counted.

If an exact match is not found, PoolQ will attempt to match to a known hairpin sequence allowing a single nucleotide mismatch. An N in the hairpin sequence is considered a single nucleotide mismatch. The exact match setting allows you to override the single nucleotide mismatch behavior and score only exact matches.

If a hairpin sequence is a single nucleotide mismatch to two or more different hairpins, PoolQ will discard the read by default. It is possible to override this behavior as well with the include ambiguous setting, in which case PoolQ scores the read for every hairpin sequence that is a single nucleotide mismatch.

If PoolQ matches a read to a barcode that is mapped to a condition, and a hairpin that is mapped to one or more hairpin IDs, then the counts are incremented for all of the matching condition/hairpin ID pairs.

Hairpins are counted as unexpected sequences if they are not successfully matched by the above procedure.

Contact Us

Your feedback of any kind is much appreciated. Please email us at rnaiinformatics@broadinstitute.org.