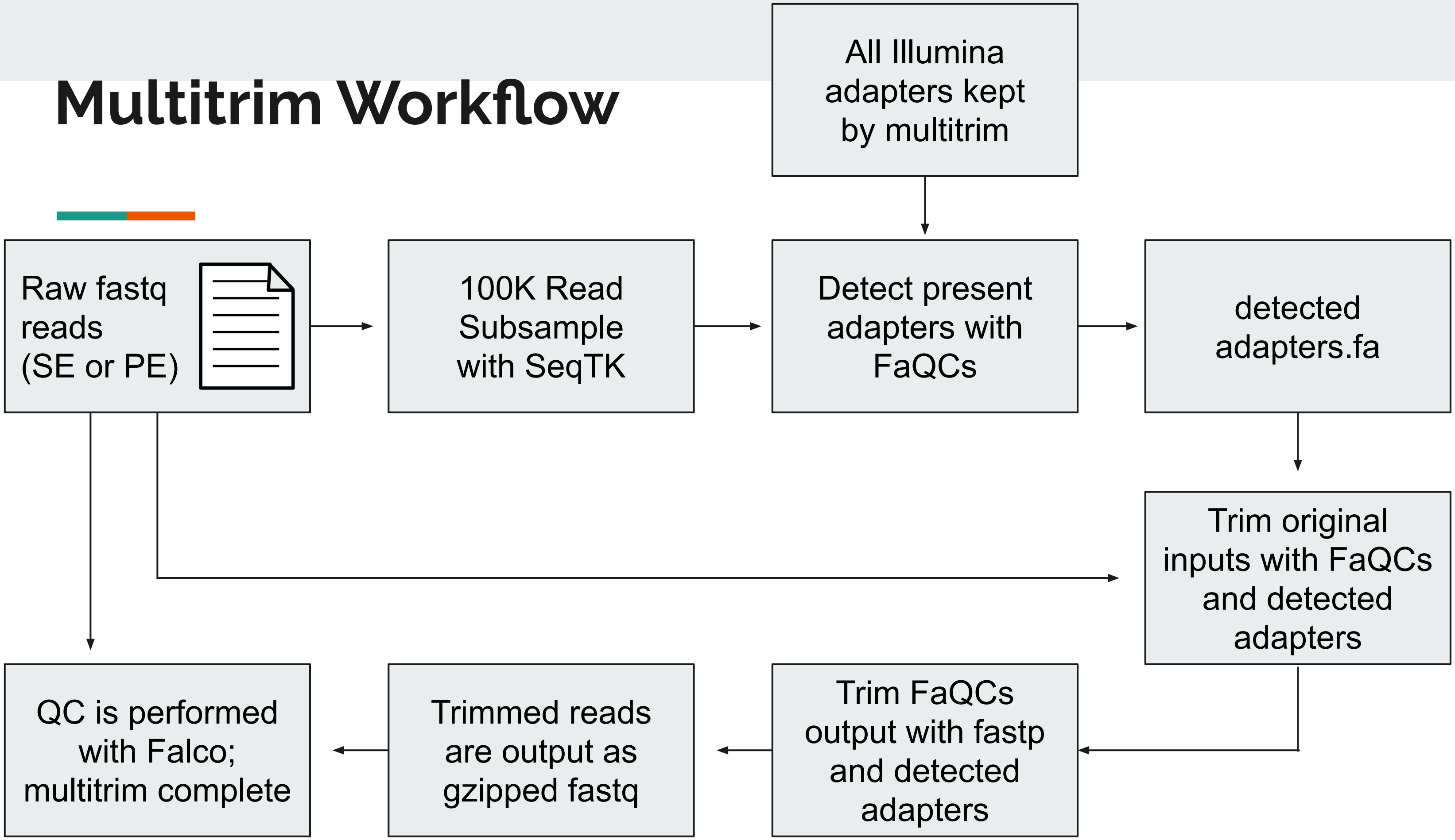
Multitrim User Manual

Kenji Gerhardt

Multitrim Workflow



All Illumina adapters kept by multitrim

Raw fastq reads (SE or PE)

100K Read Subsample with SeqTK

Detect present adapters with FaQCs

detected adapters.fa

Trim original inputs with FaQCs and detected adapters

Trim FaQCs output with fastp and detected adapters

Trimmed reads are output as gzipped fastq

QC is performed with Falco; multitrim complete

Requirements and Installation

Multitrim requires the python programming language and the following tools to be installed:

- FaQCs:
  - https://github.com/LANL-Bioinformatics/FaQCs
- Fastp:
  - https://github.com/OpenGene/fastp
- Falco:
  - https://github.com/smithlabcode/falco
- SeqTK:
  - https://github.com/lh3/seqtk

The easiest way to do all of this is to use Conda. Instructions for doing so are on the next few slides.

Installation of Miniconda

1. (Windows only): Get Ubuntu for Windows and install it.
2. Download the appropriate version of Miniconda:
3. Follow the installation instructions:
    a. Ubuntu for Windows/Linux
    b. MacOS

Installation of Multitrim

1. Acquire multitrim with git by running the following command in your terminal:
    a. `git clone https://github.com/KGerhardt/multitrim`
2. Create the multitrim environment by running the following command:
    a. `conda env create -f multitrim/multitrim.yml`
3. Activate the conda environment for multitrim with the following command:
    a. `conda activate multitrim`
4. You’re ready to run the multitrim script.
    a. You need to rerun (3) each time you open your terminal. Conda environments are not active by default.

Quality of Life (Optional)

- Make multitrim available as an alias by running the following command in your terminal:
  - alias multitrim=”python *path/to/multitrim_directory*/multitrim/multitrim.py”
- Make it permanent by adding it to your profile:
  - Ubuntu for Windows/Linux:
  - MacOS

If you don’t do this, you’ll need to remember where the multitrim.py script is located and use it as python /path/to/multitrim_directory/multitrim/multitrim.py [OPTS] each time. It’s much easier to create the alias once and then always have it available when you open your terminal.

Using Multitrim - Basic usage

Note: I will assume going forward that you've followed the optional instructions on slide 5, and can use multitrim by simply typing "multitrim" on the command line.

- Single end reads:
  - multitrim -u [unpaired_reads.fq]
- Paired end reads:
  - multitrim -1 [forward_reads.fq] -2 [reverse_reads.fq]

That's it.

Note: input Reads can be gzipped (they probably end with ".fq.gz"), and multitrim still works.

Using Multitrim - Controlling Outputs

- Multitrim has a standard set of naming conventions for its outputs:
  - (directory)/(prefix_)(file_identifier)_(base_name).(extension)
- There is no default prefix, and the default directory is the current working directory.
  - You can specify a directory with the -d option
  - You can specify a prefix with the -p option
- The naming convention for reads and their QC reports include information from the input file names.
  - This means you can run multitrim on multiple sets of reads, place the results in the same directory, and all critical information will be kept.
  - If you want to keep the fastp reports, the detected adapter files, and the subsample results for multiple reads in a single directory, use prefixes to make these names unique.

Using Multitrim - Trimming Options

Multitrim uses both FaQCs and fastp to trim. FaQCs performs an initial trim, and then fastp trims those results. The following settings for these tools are available to a user of multitrim:

- FaQCs:
    - --score [INT]; sets FaQCs score parameter. Default 27.
- Fastp:
    - --advanced; applies poly-G tail trim and low-complexity read filters. Read fastp documentation for more detail
    - --window [INT]; sets the width of the sliding window for fastp trimming. Default 3 bp.
    - --window_qual [INT]; sets the minimum avg. quality for bases in the window. Default 20.
    - --min_L [INT]; minimum read length post-trim for a read to be kept. Default 50.

Using Multitrim - Other Useful Options

- --threads [INT]
    - Uses the specified number of threads to perform trimming and QC. More threads = faster. Default 1.
- --max
    - Uses all of the cores available to your computer/HPC job. Overwrites --threads if both are supplied.
    - **You should always use this option.**
- --min_adapt_pres [float]
    - Adapters must be detected in the specified percent of reads during subsampling to be considered present. Default 0.1

Using Multitrim - Other Less-Useful Options

- --phred_fmt
  - Exists to resolve an issue with FaQCs in SE mode. Sets phred scoring offset. Default 33.
- --skip_faqcs
  - Skips the FaQCs trimming step. Generally results in over-trimmed reads.
- --skip_fastp
  - Skips the Fastp trimming step. Generally results in under-trimmed reads.
- The following allow you to supply the location of binaries for each tool, in case you cannot conda install
  - --falco
  - --seqtk
  - --faqcs
  - --fastp

Multitrim Outputs:

- Both SE and PE:
  - detected_adapters.fasta
  - post_trim_fastp.html
  - post_trim_fastp.json
  - Subsample_Adapter_Detection.stats.txt
  - Subsample_Adapter_Detection_qc_report.pdf

Note: all files here will begin with the user's chosen prefix, if one is supplied with -p

Note: File base names are created as such: base_name.extension(.gz) -> base_name

- SE only:
  - unpaired.pre_trim_qc_[file_base_name].html
  - unpaired.post_trim_qc_[file_base_name].html
  - unpaired.trimmed_[file_base_name].fq.gz
- PE only:
  - 1.pre_trim_qc_[forward_base_name].html
  - 1.post_trim_qc_[forward_base_name].html
  - 1.trimmed_[forward_base_name].fq.gz
  - 2.pre_trim_qc_[reverse_base_name].html
  - 2.post_trim_qc_[reverse_base_name].html
  - 2.trimmed_[reverse_base_name].fq.gz

Understanding the outputs:

- *pre_trim_QC*
    - Falco QC report on your inputs before any trimming
- *post_trim_QC*
    - Falco QC report on the final, fully trimmed outputs
- [1/2/unpaired].trimmed*.fq.gz
    - Your trimmed reads as a gzipped FASTQ file
- detected_adapters.fasta - FASTA file containing detected adapter sequences.
    - May be empty if no adapters are found.
- post_trim_fastp.[html/json]
    - graphical/text report from fastp trim.
- Subsample_Adapter_Detection.[pdf/stats.txt]
    - graphical/text reports from FaQCs on the subsampled 100K reads used to detect adapters.