# Bio-Rad ATAC-Seq Analysis Toolkit Tutorial

Bio-Rad Laboratories, Inc.

# Table of Contents

## Introduction

This document is intended to provide a step-by-step walkthrough for the Bio-Rad ATAC-Seq Analysis Toolkit using an example dataset. We display all commands necessary to perform secondary analysis of single-cell ATAC-Seq data, beginning with FASTQ files.

The Tools in the Toolkit are modular but are intended to be run in a mostly linear fashion, with one “Tool” accepting the output of an upstream Tool as its input. The modularity permits users to substitute their own Tools of choice at any point in the workflow. As long as the appropriate input for a given Tool is formatted properly, it will be processed regardless of its origin.

## Prerequisites

The Bio-Rad ATAC-Seq Analysis Toolkit is a command line Tool that is packaged into Docker containers. Each Tool is its own Docker container that accomplishes a specific step in the analysis workflow.

Therefore, the user simply needs to install Docker in order to get started with the Toolkit. Docker is available as a free Community Edition, which can be downloaded from `Docker` (https://www.docker.com/get-started). We assume that Docker is installed and running on your system. Advanced knowledge of Docker is not necessary to use the Toolkit, only a few fundamental commands are required to launch each container.

The user must also provide a genome index for alignment of reads. The aligner used in this workflow is `BWA` (http://bio-bwa.sourceforge.net/). Due to the size of genome indices, they are not provided. But we provide instructions for creating a genome index using the provided BWA Alignment Tool in the appendix of this document.

## System Requirements

**Minimum:** 8 CPU, 32 GB RAM, and 500 GB available disk space.

**Recommended:** 16+ CPU, 64 GB RAM, and 1+ TB available disk space.

## Docker Configuration

To ensure the best possible experience running Docker, we advise users to make all of their system resources available to Docker. In Docker > Preferences > Advanced, drag the CPU and RAM sliders to the maximum.

### Docker Login

To access the bioraddbg Dockerhub organization, ensure that you are logged in to your Dockerhub account by executing `docker login` at the command line interface.

## Tutorial Overview

This tutorial is presented from a UNIX interface. All commands are appropriate for use in any environment that supports UNIX commands. If the user is working from Windows 10, `git bash` (https://git-scm.com/downloads) is a good choice.

The commands written here can be copied to duplicate this analysis.

## Windows

A few extra steps are required to get Docker running properly if run in Windows.

### Windows Shared Drives

Windows users will likely need to share their hard drive(s) with Docker. This is done at Docker > Preferences > Shared Drives.

### Windows Aliases

Users may need to update their Docker alias in Windows if attempting to execute UNIX-like commands in a git bash terminal.

```
DOCKER_HOME=/c/'Program Files'/Docker/Docker/resources/bin
docker() { export MSYS_NO_PATHCONV=1; ("$DOCKER_HOME/docker.exe" "$@"); export
MSYS_NO_PATHCONV=0; }
```

## Toolkit Overview

The Toolkit is designed to process one sample at a time, where each sample is represented by some number of paired FASTQ files. These FASTQ files are the entry point to the workflow.

For this demonstration, we begin with a directory containing the raw FASTQ files from one sample. All analysis will proceed from this directory and the Toolkit will read from and write to it.

## Container Structure

Each container in the Toolkit follows the same general pattern. The sample directory is mounted as a Docker volume, then the container parses the input file structure to find the appropriate inputs, executes its function, then writes output to a unique directory.

1. Each container is launched via the `docker run` command.

2. The sample directory is mounted as a Docker volume by the `-v` argument. The mapping occurs with the following syntax: `local_file_directory/:/container_directory/`. For most cases, the `container_directory` is referred to as `/data/` and the local file directory is the location of the input FASTQ files on the local file system.

3. The container to be launched is then named with the following syntax: `bioraddbg/<container-name>`.

4. An input path from the perspective of the container is provided: `-i /data/`.

5. An output path from the perspective of the container is provided: `-o /data/<output_directory>`.

All together, the command to launch a container appears as:

```
docker run --rm -v \
  ~/my_fastq_files/:/data/ \
  bioraddbg/<container-name> \
  -i /data/input_directory \
  -o /data/output_directory
```

## File Handling

Our input to the Toolkit is simply a directory containing the raw FASTQ files for the analysis. The directory structure before the start of the analysis looks like this:

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
└── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
```

## FASTQ Quality Control

(bioraddbg/atac-seq-fastqc)

### Summary

This Docker container runs `FASTQC` (https://www.bioinformatics.babraham.ac.uk/projects/fastqc/) and generates HTML reports for each of the FASTQ files in the input directory.

### Inputs

The sample directory containing raw FASTQ files.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-fastqc \
  -i /data/ \
  -o /data/fastqc_results
```

### Output

An HTML report and .zip file containing the QC data are generated for each FASTQ file.

```
 ~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
└── fastqc_results
    ├── N714-Exp68-sample11_S1_L001_R1_001_fastqc.html
    ├── N714-Exp68-sample11_S1_L001_R1_001_fastqc.zip
    ├── N714-Exp68-sample11_S1_L001_R2_001_fastqc.html
    ├── N714-Exp68-sample11_S1_L001_R2_001_fastqc.zip
    ├── N714-Exp68-sample11_S1_L002_R1_001_fastqc.html
    ├── N714-Exp68-sample11_S1_L002_R1_001_fastqc.zip
    ├── N714-Exp68-sample11_S1_L002_R2_001_fastqc.html
    ├── N714-Exp68-sample11_S1_L002_R2_001_fastqc.zip
    ├── N714-Exp68-sample11_S1_L003_R1_001_fastqc.html
    ├── N714-Exp68-sample11_S1_L003_R1_001_fastqc.zip
    ├── N714-Exp68-sample11_S1_L003_R2_001_fastqc.html
    ├── N714-Exp68-sample11_S1_L003_R2_001_fastqc.zip
    ├── N714-Exp68-sample11_S1_L004_R1_001_fastqc.html
    ├── N714-Exp68-sample11_S1_L004_R1_001_fastqc.zip
    ├── N714-Exp68-sample11_S1_L004_R2_001_fastqc.html
    └── N714-Exp68-sample11_S1_L004_R2_001_fastqc.zip
```

## FASTQ Debarcoding

(bioraddbg/atac-seq-debarcode-bap)

### Summary

This Docker container runs `BAP` (https://github.com/caleblareau/bap) to “debarcode” FASTQ files. The barcodes are parsed out of the R1 reads and appended to the read name for all reads with valid barcodes. Reads that fail debarcoding are discarded from the analysis.

### Inputs

The sample directory containing the raw FASTQ files.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-debarcode-dbg \
  -i /data/ \
  -o /data/debarcoded_reads
```

### Output

The outputs of debarcoding are debarcoded FASTQ files. Also included is a summary report of the debarcoding process, indicating how many reads were correctly debarcoded. All reads that failed debarcoding will have been discarded.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── debarcoded_reads
│   ├── R1.fastq.gz
│   ├── R2.fastq.gz
│   ├── mismatches.csv.gz
│   └── parse.sumstats.log
└── fastqc_results
```

## Alignment

(bioraddbg/atac-seq-bwa)

### Summary

The alignment step takes debarcoded (and possibly trimmed) reads and aligns them to a reference genome. The Toolkit **does not** provide the reference genome. The mixed hg19-mm10 reference genome used here is available by contacting Bio-Rad Support (support@bio-rad.com). The path to this directory on our system is: `~/genomes/hg19-mm10` and its contents appear as:

```
hg19-mm10/
├── annotation
│    ├── TSS
│    │    ├── README.txt
│    │    ├── hg19-mm10.chromInfo.txt
│    │    ├── hg19-mm10.refGene.TSS.bed
│    │    ├── hg19-mm10.refGene.tss.2k.bed
│    │    └── hg19-mm10.refgene.txt
│    └── blacklist
│         └── hg19-mm10.full.blacklist.bed
├── bwa
│   ├── hg19-mm10.amb
│   ├── hg19-mm10.ann
│   ├── hg19-mm10.bwt
│   ├── hg19-mm10.pac
│   └── hg19-mm10.sa
└── fasta
    └── hg19-mm10.fa
```

### Inputs

This Tool requires the user to mount both the sample directory and the reference genome directory to the Docker container, which requires a second `-v` argument.

**Note:** The `-i` argument needs to be pointed at a directory containing the FASTQ files for alignment.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  -v ~/genomes/hg19-mm10/bwa/:/genome/ \
  bioraddbg/atac-seq-bwa \
  -i /data/debarcoded_reads/ \
  -o /data/alignments/ \
  -r /genome/
```

### Output

The output of the Alignment Tool is a .bam file and index with the bead barcode annotated with the XB:Z tag.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample04_R2_001.downSampled.fastq.gz
├── alignments/
│    ├── alignments.possorted.tagged.bam
│    └── alignments.possorted.tagged.bam.bai
├── debarcoded_reads/
├── fastqc_results
└── trimmed_reads/
```

## Alignment QC

(bioraddbg/atac-seq-alignment-qc)

### Summary

This Tool performs high-level alignment QC on the .bam file generated in the preceding step. It wraps around the `PICARD` (https://broadinstitute.github.io/picard/) CollectAlignmentSummaryMetrics function and writes a text file containing alignment summary metrics.

### Inputs

This Tool requires the user to mount both the sample directory and the reference genome directory to the Docker container, which requires a second `-v` argument.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  -v ~/genomes/hg19-mm10/fasta/:/genome/ \
  bioraddbg/atac-seq-alignment-qc \
  -i /data/alignments/ \
  -r /genome/hg19-mm10.fa \
  -o /data/alignment_qc
```

### Outputs

A single .txt file containing the alignment summary metrics is written to the output path.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── alignment_qc
│   ├── alignments.possorted.tagged.aligment_summary_qc.txt
│   └── db_output.csv
├── alignments/
├── debarcoded_reads/
├── fastqc_resualts
└── trimmed_reads/
```

## Bead Filtration

(bioraddbg/atac-seq-filter-beads)

### Summary

Due to the nature of droplet-based single-cell sequencing, a large proportion of beads are not captured in the same droplet as a cell. The bead filtration step identifies barcodes with DNA from cells and filters out beads from cell-free droplets.

In addition, the bead filtration step performs the necessary calculation of identical fragments observed across multiple unique barcodes. A fragment Jaccard index is calculated for all barcode pairs, which is then used to determine a threshold that, should two beads have an identical proportion of fragments above, dictates their merging.

### Inputs

The output of the Alignments Tool is used as the input to Bead Filtration. If a different alignment method is used, a directory containing a position-sorted, indexed .bam file with bead barcodes annotated as XB:Z can be substituted.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-filter-beads \
  -i /data/alignments/ \
  -o /data/bead_filtration/ \
  -r hg19-mm10
```

### Outputs

A number of outputs are generated during bead filtration. Two directories containing the results of bead filtration and Jaccard index calculation are produced. A detailed discussion of these files is beyond the scope of this tutorial. Their main use is in the following step, where the input .bam file is processed with the above-knee bead barcodes and Jaccard index calculated in this step.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── alignment_qc
├── alignments
└── bead_filtration
│   ├── bamMetrics.csv
│   ├── jaccard
│   │   ├── jaccard_kneeCurve.png
│   │   └── threshold.csv
│   └── whitelist
│       ├── aboveKneeBarcodes.csv
│       ├── alignments.possorted.tagged.barcodequants.csv
│       ├── alignments.possorted.tagged.filtered.barcodequants.csv
│       ├── bap_bead_whitelist.csv
│       ├── bead_kneeCall.rds
│       ├── bead_kneeCurve.png
│       └── thresholds.csv
├── debarcoded_reads/
├── fastqc_results
└── trimmed_reads/
```

## Bead Deconvolution

(bioraddbg/atac-seq-deconvolute)

### Summary

This Tool uses `BAP` (https://github.com/caleblareau/bap) to perform the deconvolution of bead multiplets. At a high level, this Tool takes the bead barcodes that were deemed to contain DNA fragments from cells and merges bead barcodes that "see" the same cell to a droplet barcode.

### Inputs

The output of Bead Filtration, above, is used as the input to this Tool. It is strongly recommended that the the aforementioned Tool be used to generate the appropriate input to this container.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-deconvolute \
  -i /data/alignments/ \
  -f /data/bead_filtration/ \
  -r hg19-mm10 \
  -o /data/deconvoluted_data/
```

### Outputs

A number of outputs are generated in Bead Deconvolution. The main output is a .bam file with cell barcodes annotated with the DB tag and alignments deduplicated at the cell level. A number of quality control statistics are generated, which are summarized in the report generated at the end of this workflow.

**Note:** `alignments.possorted.tagged.jaccardOverlapKnee.pdf` is a blank file that can be ignored.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── alignment_qc
├── alignments
├── bead_filtration
├── debarcoded_reads/
├── deconvoluted_data
│    ├── final
│    │    ├── alignments.possorted.tagged.NCsumstats.tsv
│    │    ├── alignments.possorted.tagged.QCstats.csv
│    │    ├── alignments.possorted.tagged.bap.bam
│    │    ├── alignments.possorted.tagged.bap.bam.bai
│    │    ├── alignments.possorted.tagged.barcodeTranslate.tsv
│    │    ├── alignments.possorted.tagged.barcodequants.csv
│    │    └── alignments.possorted.tagged.implicatedBarcodes.csv.gz
│    ├── knee
│    │    ├── alignments.possorted.tagged.bapParams.csv
│    │    └── alignments.possorted.tagged.jaccardOverlapKnee.pdf
│    └── logs
├── fastqc_results
└── trimmed_reads/
```

## Cell Filtration

(bioraddbg/atac-seq-cell-filter)

### Summary

This Tool performs a “knee call” on the cell level to produce a final cell count.

### Inputs

The output of Bead Deconvolution is used as input to this Tool.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-cell-filter \
  -i /data/deconvoluted_data/ \
  -o /data/cells_filtered/ \
  -r hg19-mm10
```

### Outputs

Numerous files are output by the bead filtration container. A description of each file is beyond the scope of this tutorial. Key files are:

1. alignments.possorted.tagged.final.bam: Contains **above knee** cells annotated with the RG:Z tag, making this .bam file ready for input to `chromVAR` (https://www.nature.com/articles/nmeth.4401).
2. crosstalk.csv, crosstalk.*.png: Crosstalk metrics and plots if the experiment is mixed species.
3. kneecurve.png: Diagnostic plot illustrating the droplet knee call. If multiple thresholds were found in knee calling, these values are stored in crosstalk.csv.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── alignment_qc
├── alignments
├── bead_filtration
├── debarcoded_reads/
├── deconvoluted_data
├── fastqc_results
├── cells_filtered
│    ├── aboveKneeBarcodes.csv
│    ├── alignments.possorted.tagged.barcodesPerDrop.csv
│    ├── alignments.possorted.tagged.barcodesPerDrop.png
│    ├── alignments.possorted.tagged.dropBarcodeUniqueReadCounts.csv
│    ├── alignments.possorted.tagged.final.bam
│    ├── alignments.possorted.tagged.final.bam.bai
│    ├── alignments.possorted.tagged.mergedDroplets.csv
│    ├── belowKneeBarcodes.csv
│    ├── crosstalk.csv
│    ├── crosstalk.noLogScale.png
│    ├── crosstalk.withLogScale.png
│    ├── density.csv
│    ├── fragsByBeadsPerDrop.csv
│    ├── fragsByBeadsPerDropBin.png
│    ├── fragsByBeadsPerDropDist.png
│    ├── hg19_reads.csv
│    ├── kneeCurve.png
│    ├── local_maxs.csv
│    ├── local_mins.csv
│    ├── log_counts.tsv
│    ├── mm10_reads.csv
│    └── thresholds.csv
└── trimmed_reads/
```

## Peak Calling

(bioraddbg/atac-seq-call-peaks)

### Summary

This Tool performs peak calling using `MACS2` (https://github.com/taoliu/MACS). The primary input is the deconvoluted .bam file, which is treated as a bulk ATAC-Seq experiment for the purpose of peak calling.

### Inputs

The intended input to this Tool is the output of the deconvolution Tool.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-macs2 \
  -i /data/deconvoluted_data/ \
  -r hg19-mm10 \
  -o /data/peaks
```

### Outputs

This Tool outputs peak calls from MACS2 and postprocessed, nonoverlapping, fixed-width peaks of 250 base pairs around the peak summits.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── alignment_qc
├── alignments
├── bead_filtration
├── debarcoded_reads/
├── deconvoluted_data
├── fastqc_results
├── cells_filtered
├── peaks
│    ├── alignments.possorted.tagged.fixedwidthpeaks.bed
│    ├── alignments.possorted.tagged_peaks.narrowPeak
│    ├── alignments.possorted.tagged_peaks.xls
│    └── alignments.possorted.tagged_summits.bed
└── trimmed_reads/
```

## ATAC-Seq QC

(bioraddbg/atac-seq-qc)

### Summary

This Tool performs QC on the deconvoluted alignment data. This Tool ignores single cell information and computes common ATAC-Seq metrics from the deconvoluted alignments.

### Inputs

The intended inputs to this Tool are the deconvoluted alignment directory and peak calls.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-qc \
  -r hg19-mm10 \
  -d /data/deconvoluted_data/ \
  -p /data/peaks \
  -o /data/atac_qc
```

### Outputs

This Tool generates a FRIP score, insert size metrics histogram, and a TSS score with plots.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── alignment_qc
├── alignments
├── atac_qc
│    ├── frip.txt
│    ├── insert_size_histogram.pdf
│    ├── insert_size_metrics.txt
│    ├── tss.tsv
│    ├── tss_large.png
│    └── tss_small.png
├── bead_filtration
├── debarcoded_reads/
├── deconvoluted_data
├── fastqc_results
├── cells_filtered
├── peaks
└── trimmed_reads/
```

## Count Matrix

(bioraddbg/atac-seq-chromvar)

### Summary

The final processing step is the computation of a cells-by-peaks count matrix. This Tool uses chromVAR to generate a cells-by-peaks count matrix, where cells are denoted by the RG:Z tag on the final .bam file produced by the Cell Filtration Tool.

### Inputs

The inputs to this Tool are the directory containing the final .bam file and the peak calls.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-chromvar \
  -d /data/cells_filtered \
  -p /data/peaks \
  -o /data/count_matrix \
  -r hg19-mm10
```

### Outputs

The output of this Tool is a count matrix as a .csv file along with a count plot. Each column in the count matrix is an above-knee droplet barcode and each row is a peak in `alignments.possorted.tagged.fixedwithpeaks.bed` in the peaks directory. This count matrix is an appropriate input to chromVAR for tertiary analysis.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── alignment_qc
├── alignments
├── atac_qc
├── bead_filtration
├── count_matrix
│    ├── barcodes_out.csv
│    ├── count_matrix.csv
│    ├── count_matrix.mtx
│    ├── count_plot.png
│    ├── functionalPeaks.bed
│    └── peak_names_out.csv
├── debarcoded_reads/
├── deconvoluted_data
├── fastqc_results
├── cells_filtered
├── peaks
└── trimmed_reads/
```

## Report

(bioraddbg/atac-seq-report)

### Summary

This Tool aggregates the outputs of all other Tools in the Toolkit and generates a report summarizing the secondary analysis.

### Inputs

The input to this Tool is the root of the analysis directory.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-report \
  -i /data/ \
  -o /data/report
```

### Outputs

The main outputs of this Tool are `atacReport.html` and `atacReport.pdf`, which are aggregate reports of the outputs specifically generated by the Tools in this ATAC-Seq Analysis Toolkit. If users substitute their own software in place of those in the Toolkit, compatibility with `atac-seq-report` is not guaranteed. Therefore, we use `MultiQC` (https://multiqc.info) to generate a more general report alongside our own reporting scripts.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── alignment_qc
├── alignments
├── atac_qc
├── bead_filtration
├── count_matrix
├── debarcoded_reads/
├── deconvoluted_data
├── fastqc_results
├── cells_filtered
├── peaks
├── report
│    ├── atacReport.html
│    ├── multiqc_data
│    │    ├── multiqc.log
│    │    ├── multiqc_data.json
│    │    ├── multiqc_general_stats.txt
│    │    ├── multiqc_picard_AlignmentSummaryMetrics.txt
```

```
|    |    ├── multiqc_picard_insertSize.txt
|    |    └── multiqc_sources.txt
|    ├── multiqc_report.html
|    ├── pdfReport.pdf
|    └── qcMetricsReport.csv
└── trimmed_reads/
```

## Read Trimming

(bioraddbg/atac-seq-trim-reads)

### Summary

This Docker container runs one of two read trimming tools: `trimmomatic` (http://www.usadellab.org/cms/?page=trimmomatic) or `trimgalore/cutadapt` (https://www.bioinformatics.babraham.ac.uk/projects/trim_galore/) to trim FASTQ files of low-quality bases and adapter sequences. This step is not necessary to conduct a secondary analysis. It is intended to help provide the best quality data into alignment should the FASTQ quality control report indicate poor-quality reads.

For the sake of completeness, we will run the tool here.

### Inputs

The sample directory containing debarcoded FASTQ files as R1.fastq.gz and R2.fastq.gz.

### Execution

```
docker run --rm -v ~/tutorial/:/data/ \
  bioraddbg/atac-seq-trim-reads \
  -i /data/debarcoded_reads \
  -o /data/trimmed_reads
```

### Output

An HTML report and .zip file containing the QC data are generated for each FASTQ file. If performed, the output of read trimming should be used as the input to `atac-seq-bwa` for alignment.

```
~/tutorial/
├── N714-Exp68-sample11_S1_L001_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L001_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L002_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L003_R2_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R1_001.downSampled.fastq.gz
├── N714-Exp68-sample11_S1_L004_R2_001.downSampled.fastq.gz
├── debarcoded_reads/
├── fastqc_results
└── trimmed_reads/
    ├── R1.fastq.gz
    ├── R1_unpaired.fastq.gz
    ├── R2.fastq.gz
    ├── R2_unpaired.fastq.gz
    ├── trim_summary.txt
    └── trimlog.log
```

# Appendix

## Understanding Outputs

A number of files are generated throughout the workflow. The most relevant for downstream use are generated by `bioraddbg/atac-seq-cell-filter` and `bioraddbg/atac-seq-chromvar`.

`bioraddbg/atac-seq-cell-filter` produces `alignments.possorted.tagged.final.bam`, which is the fully processed alignment map with reads deduplicated for each cell. Reads from above-threshold cell barcodes are tagged as RG:Z.

`bioraddbg/atac-seq-chromvar` produces the reads-in-peaks count matrices that are consumed by downstream tertiary analysis Tools. These matrices are produced in both sparse and dense formats. The dense formatted matrices are in .csv format and named `count_matrix.csv`. Dense formatted matrices are typically quite large and contain many zero values for each cell barcode. The file size of these dense matrices makes downstream work challenging, thus we also produce sparse formatted matrices in the MatrixMarket format. This sparse matrix is written to `count_matrix.mtx`. The row and column names of this sparse matrix are written out to `peak_names_out.csv` and `barcodes_out.csv`, respectively.

## Generating a Genome Index for Use with the Toolkit

### Introduction

This section describes the process of creating a reference genome for use with BWA in the Bio-Rad SureCell ATAC-Seq Analysis Toolkit. The Toolkit supports hg19, mm10, and hg19-mm10. We will demonstrate the generation of a genome index for hg19 using the Alignment Tool.

### Prerequisites

A UNIX-like environment with Docker, bedtools, awk, and wget installed.

### Steps

1. Create a directory on your local file system to store the genome reference.

```
mkdir -p ~/genomes/hg19/fasta
mkdir -p ~/genomes/hg19/TSS
mkdir -p ~/genomes/hg19/blacklist
```

2. Download the FASTA files for the genome from UCSC.

```
cd ~/genomes/hg19/fasta
wget ftp://hgdownload.cse.ucsc.edu/goldenPath/hg19/chromosomes/*
```

3. Concatenate only the major chromosomal contigs, then clean up files; genome must be uncompressed FASTA format.

```
cat $(ls *.fa.gz | grep -v _) > hg19.fa.gz
rm $(ls | grep -v hg19.fa.gz)
gunzip hg19.fa.gz
```

4. Download blacklist and TSS annotation.

```
cd ~/genomes/hg19/TSS
wget http://hgdownload.cse.ucsc.edu/goldenpath/mm10/database/refGene.txt.gz
gunzip refGene.txt.gz
mv refGene.txt hg19.refgene.txt
```

```
awk '{if($4=="-"){v=$6}else{v=$5} print $3,v,v,$4}' OFS="\t" hg19.refgene.txt | awk
'length($1) < 10 {print $0}' | awk '$1 != "hg19_chrY" {print $0}' | sortBed -i
stdin | uniq > hg19.refGene.TSS.bed
wget http://hgdownload.cse.ucsc.edu/goldenpath/hg19/database/chromInfo.txt.gz
gunzip chromInfo.txt.gz
bedtools slop -i hg19.refGene.TSS.bed -g chromInfo.txt -b 2000 > hg19.refGene.
tss.2k.bed
cd ~/genomes/hg19/blacklist
wget https://raw.githubusercontent.com/buenrostrolab/proatac/v0.2.0/proatac/anno/
blacklist/hg19.full.blacklist.bed
```

5. Launch the Alignment Tool interactively with Docker, mounting the `~/genomes/` directory as a volume.

```
docker run --rm -it -v ~/genomes/hg19/:/genome/ --entrypoint "/bin/bash"
bioraddbg/atac-seq-bwa
```

6. Build the genome using only the major chromosomal contigs.

```
mkdir -p /genome/bwa
bwa-0.7.17/bwa index -p /genome/bwa/hg19 /genome/fasta/hg19.fa
```

The BWA index will be written to your `~/genome/bwa` directory as: `hg19.amb, hg19.ann, hg19.bwt, hg19.pac, hg19.sa`.

Press `ctrl + d` to exit the Alignment Tool.

You can now point the Alignment Tool genome toward the local path `~/genomes/hg19/bwa` for hg19.

### Automating the Toolkit

Below, we present a bash script that will run the above steps sequentially without the need for user intervention after the execution of each Tool.

It requires three inputs:

1. The path to the sample directory containing FASTQ files.
2. The reference genome (hg19, mm10, or hg19-mm10).
3. A path to the reference genome directory, such as the one presented above in the Alignment section.

Using this script `runPipeline.sh`, we can execute the entire pipeline sequentially for one sample.

```
./runPipeline.sh -i ~/tutorial/ -r hg19-mm10 -g ~/genomes/hg19-mm10/
```

```
#!/bin/bash

set -e

#BEGIN ARGUMENT PARSING
function usage
{
    echo "  ";
    echo "usage: runPipeline.sh -i sample_dir -g genome_name -d genome_dir";
    echo "  ";
    echo "  -i | --sample_directory     : Sample directory containing fastq
files for one sample index.";
    echo "  -g | --genome_name          : One of hg19, mm10, hg19-mm10";
    echo "  -d | --genome_dir           : Path to root of genome directory";
    echo "  -h | --help             : This message";
    echo "  ";
}
function parse_args
{
    # positional args
    args=()
    # named args
    while [ "$1" != "" ]; do
        case "$1" in
            -i | --sample_directory )        sample_dir="$2";    shift;;
            -g | --genome_name )        genome_name="$2";     shift;;
            -d | --genome_directory )        genome_dir="$2";    shift;;
            -h | --help )                   usage;
exit;; # quit and show usage
            * )                             args+=("$1")
# if no match, add it to the positional args
        esac
    shift # move to next kv pair
    done
    # restore positional args
    set -- "${args[@]}"
    #Input data directory selected?
    if [[ -z "$sample_dir" ]]; then
        echo "No sample directory input."
        usage
        exit 1;
    fi
    #Input data directory exists?
    if [[ ! -d "$sample_dir" ]]; then
        echo "Sample directory does not exist -- provide the path to a
directory containing fastq files for one sample index."
        usage
        exit 1;
    fi
    #validate required args
    if [[ -z "$genome_dir" ]]; then
        echo "No genome directory input."
        usage
        exit 1;
    fi
    #Valid genome bwa?
    if [[ ! -d "$genome_dir/bwa/" ]]; then
        echo "Improper genome directory format. Need /genome_dir/bwa/"
        usage
        exit 1;
    fi
    #Valid genome fasta?
    if [[ ! -d "$genome_dir/fasta/" ]]; then
        echo "Improper genome directory format. Need /genome_dir/fasta/"
        usage
        exit 1;
    fi
    #Valid reference genome?
    case "$genome_name" in
        hg19|mm10|"$genome_name")
            echo Reference genome: "$genome_name" ;;
        *)
            echo "Invalid reference genome name. Use one of hg19, mm10,
"$genome_name"."
            usage
            exit 1 ;;
    esac
}
function run
{
    parse_args "$@"
    echo "Sample Directory: $sample_dir"
    echo "Genome Directory: $genome_dir"
}
run "$@";
if [[ -d "$sample_dir/fastqc_results" ]]; then
    echo "Found fastqc results, skipping FASTQC."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-fastqc -i /data/
-o /data/fastqc_results
fi
if [[ -d "$sample_dir/debarcoded_reads" ]]; then
    echo "Found debarcoded reads, skipping debarcoding."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-debarcode-dbg -i
/data/ -o /data/debarcoded_reads
fi
if [[ -d "$sample_dir/alignments" ]]; then
    echo "Found alignments, skipping alignment."
else
    docker run --rm -v "$sample_dir":/data/ -v "$genome_dir"/bwa/:/genome/
bioraddbg/atac-seq-bwa -i /data/debarcoded_reads -o /data/alignments -r /genome
fi
if [[ -d "$sample_dir/alignment_qc" ]]; then
    echo "Found alignment qc, skipping alignment qc."
else
    docker run --rm -v "$sample_dir":/data/ -v "$genome_dir"/fasta/:/genome/
bioraddbg/atac-seq-alignment-qc -i /data/alignments/ -r /genome/"$genome_name".
fa -o /data/alignment_qc
```

```
fi
if [[ -d "$sample_dir/bead_filtration" ]]; then
    echo "Found bead filtration, skipping bead filtration."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-filter-beads -i /
data/alignments/ -o /data/bead_filtration/ -r "$genome_name"
fi
if [[ -d "$sample_dir/deconvoluted_data" ]]; then
    echo "Found deconvolution, skipping deconvolution."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-deconvolute
-i /data/alignments/ -f /data/bead_filtration/ -r "$genome_name" -o /data/
deconvoluted_data/
fi
if [[ -d "$sample_dir/cells_filtered" ]]; then
    echo "Found cell filtration, skipping cell filtration."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-cell-filter -i /
data/deconvoluted_data/ -o /data/cells_filtered -r "$genome_name"
fi
if [[ -d "$sample_dir/peaks" ]]; then
    echo "Found peaks, skipping MACS2."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-macs2 -i /data/
deconvoluted_data/ -o /data/peaks -r "$genome_name"
fi
if [[ -d "$sample_dir/atac_qc" ]]; then
    echo "Found bulk QC, skipping bulk QC."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-qc -r "$genome_
name" -d /data/deconvoluted_data/ -p /data/peaks/ -o /data/atac_qc
fi
if [[ -d "$sample_dir/count_matrix" ]]; then
    echo "Found count matrix, skipping count matrix."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-chromvar -d /
data/cells_filtered -p /data/peaks -o /data/count_matrix -r "$genome_name"
fi
if [[ -d "$sample_dir/report" ]]; then
    echo "Found report, skipping report."
else
    docker run --rm -v "$sample_dir":/data/ bioraddbg/atac-seq-report -i /data/
-o /data/report
fi
```

# Bio-Rad ATAC-Seq Analysis Toolkit Version Change Log

| Document Version | Toolkit Version | Date | Description |
|---|---|---|---|
| Bulletin_7191_ Ver B | v1.0.1 | January 2020 | Fixes a crash for some users during count matrix generation |
| Bulletin_7191_ Ver A | v1.0.0 | April 2019 | Original release of the toollkit |

Visit bio-rad.com/scATACSeqKit for more information.

BIO-RAD

***Bio-Rad***
***Laboratories, Inc.***

*Life Science*
*Group*

***Website*** *bio-rad.com* ***USA*** *1 800 424 6723* ***Australia*** *61 2 9914 2800* ***Austria*** *00 800 00 24 67 23* ***Belgium*** *00 800 00 24 67 23*
***Brazil*** *4003 0399* ***Canada*** *1 905 364 3435* ***China*** *86 21 6169 8500* ***Czech Republic*** *00 800 00 24 67 23* ***Denmark*** *00 800 00 24 67 23*
***Finland*** *00 800 00 24 67 23* ***France*** *00 800 00 24 67 23* ***Germany*** *00 800 00 24 67 23* ***Hong Kong*** *852 2789 3300*
***Hungary*** *00 800 00 24 67 23* ***India*** *91 124 4029300* ***Israel*** *0 3 9636050* ***Italy*** *00 800 00 24 67 23* ***Japan*** *81 3 6361 7000*
***Korea*** *82 2 3473 4460* ***Luxembourg*** *00 800 00 24 67 23* ***Mexico*** *52 555 488 7670* ***The Netherlands*** *00 800 00 24 67 23*
***New Zealand*** *64 9 415 2280* ***Norway*** *00 800 00 24 67 23* ***Poland*** *00 800 00 24 67 23* ***Portugal*** *00 800 00 24 67 23*
***Russian Federation*** *00 800 00 24 67 23* ***Singapore*** *65 6415 3188* ***South Africa*** *00 800 00 24 67 23* ***Spain*** *00 800 00 24 67 23*
***Sweden*** *00 800 00 24 67 23* ***Switzerland*** *00 800 00 24 67 23* ***Taiwan*** *886 2 2578 7189* ***Thailand*** *66 2 651 8311*
***United Arab Emirates*** *36 1 459 6150* ***United Kingdom*** *00 800 00 24 67 23*

*Bulletin 7191 Ver B US/EG* *20-0465 0720 Sig 0220*