EI Single Cell Course: Cell Ranger Tutorial

Generating FASTQs with cellranger mkfastq

Table of Contents

- Overview
- Example Workflows
- Arguments and Options
- Example Data
- Running mkfastq with a simple CSV samplesheet
- Running mkfastq with an Illumina Experiment Manager sample sheet
- Checking FASTQ output
- Troubleshooting

Overview

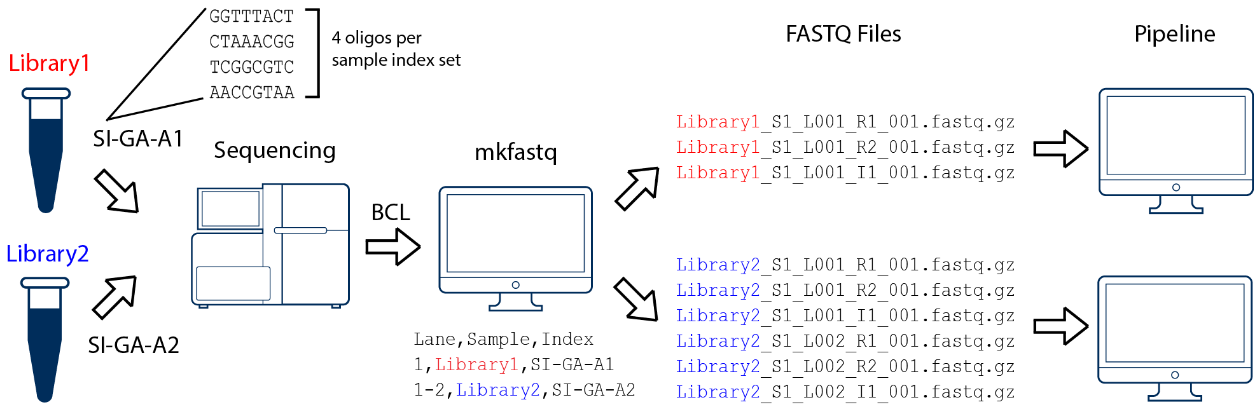
The cellranger workflow starts by demultiplexing the Illumina sequencer's base call files (BCLs) for each flowcell directory into FASTQ files. 10x has developed cellranger mkfastq, a pipeline that wraps Illumina's `bcl2fastq` and provides a number of convenient features in addition to the features of `bcl2fastq`:

- Translates 10x sample index names into the corresponding oligonucleotides in the sample index.
- Supports a simplified CSV samplesheet format to handle 10x use cases.
- Generates sequencing and 10x-specific quality control metrics, including barcode quality, accuracy, and diversity.
- Supports most `bcl2fastq` arguments, such as `-use-bases-mask`.

cellranger mkfastq supports single-indexed and dual-indexed flowcells. It will select the appropriate mode depending on the sample indexes used, and enable index-hopping filtering automatically for dual-indexed flowcells. For example, with the Dual Index Kit TT Set A, well A1 can be specified in the samplesheet as SI-TT-A1, and cellranger mkfastq will recognize the i7 and i5 indices as `GTAACATGCG` and `AGTGTTACCT`, respectively. Similarly for Single Index Kit T Set A, well A1 can be specified in the samplesheet as SI-GA-A1, and cellranger mkfastq will recognize the four i7 indexes GGTTTACT, CTAAACGG, TCGGCGTC, and AACCGTAA and merge the resulting FASTQ files.

Example Workflows

In this example, we have two 10x libraries (each processed through a separate Chromium chip channel) that are multiplexed on a single flowcell. Note that after running cellranger mkfastq, we run a separate instance of the pipeline on each library:

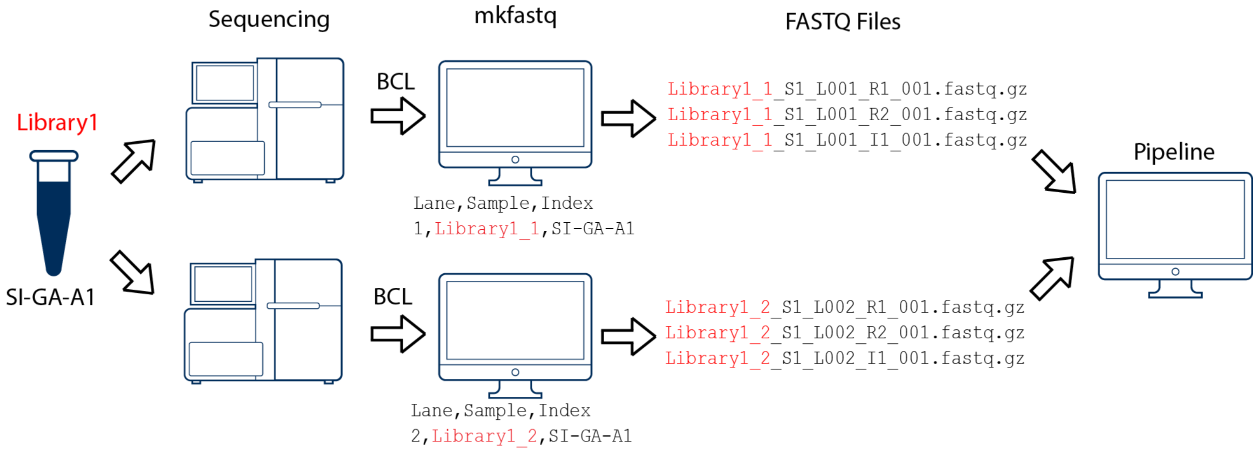


```
Lane,Sample,Index
1,Library1,SI-GA-A1
1-2,Library2,SI-GA-A2
```

```
Library1_S1_L001_R1_001.fastq.gz
Library1_S1_L001_R2_001.fastq.gz
Library1_S1_L001_I1_001.fastq.gz
```

```
Library2_S1_L001_R1_001.fastq.gz
Library2_S1_L001_R2_001.fastq.gz
Library2_S1_L001_I1_001.fastq.gz
Library2_S1_L002_R1_001.fastq.gz
Library2_S1_L002_R2_001.fastq.gz
Library2_S1_L002_I1_001.fastq.gz
```

In this example, we have one 10x library sequenced on two flowcells. Note that after running cellranger mkfastq, we run a single instance of the pipeline on all the FASTQ files generated:


Sequencing
mkfastq
FASTQ Files
Library1
SI-GA-A1
BCL
BCL
Library1_1_S1_L001_R1_001.fastq.gz
Library1_1_S1_L001_R2_001.fastq.gz
Library1_1_S1_L001_I1_001.fastq.gz
Lane,Sample,Index
1,Library1_1,SI-GA-A1
Pipeline
Library1_2_S1_L002_R1_001.fastq.gz
Library1_2_S1_L002_R2_001.fastq.gz
Library1_2_S1_L002_I1_001.fastq.gz
Lane,Sample,Index
2,Library1_2,SI-GA-A1

Arguments and Options

`cellranger mkfastq` accepts additional options beyond those shown in the table below because it is a wrapper around `bcl2fastq`. Consult the User Guide for Illumina's `bcl2fastq` for more information.

Aa Parameter	Function
--run	*(Required)* The path of Illumina BCL run folder.
--id	*(Optional; defaults to the name of the flowcell referred to by `--run`)* Name of the folder created by mkfastq.
--samplesheet	*(Optional)* Path to an Illumina Experiment Manager-compatible sample sheet which contains 10x sample index names (e.g., SI-GA-A1 or SI-TT-A12) in the sample index column. All other information, such as sample names and lanes, should be in the sample sheet.
--sample-sheet	*(Optional)* Equivalent to --samplesheet above.
--csv	*(Optional)* Path to a simple CSV with lane, sample, and index columns, which describe the way to demultiplex the flowcell. The index column should contain a 10x sample dual-index name (e.g., SI-TT-A12). This is an alternative to the Illumina IEM sample sheet, and will be ignored if `--samplesheet` is specified.
--simple-csv	*(Optional)* Equivalent to --csv above.

Aa Parameter	Function
--filter-dual-index	*(Optional)* Only demultiplex samples identified by i7/i5 dual-indices (e.g., SI-TT-A6), ignoring single-index samples. Single-index samples will not be demultiplexed. Also notice that cellranger will run single-index data, but it is not supported.
--filter-single-index	*(Optional)*Only demultiplex samples identified by an i7-only sample index, ignoring dual-indexed samples. Dual-indexed samples will not be demultiplexed.
--lanes	*(bcl2fastq option)* Comma-delimited series of lanes to demultiplex (e.g. 1,3). Use this if you have a sample sheet for an entire flowcell but only want to generate a few lanes for further 10x analysis.
--use-bases-mask	*(bcl2fastq option)* Same meaning as for `bcl2fastq`. Use to clip extra bases off a read if you ran extra cycles for QC.
--delete-undetermined	*(bcl2fastq option)* Delete the `Undetermined` FASTQs generated by `bcl2fastq`. Useful if you are demultiplexing a small number of samples from a large flowcell.
--output-dir	*(bcl2fastq option)* Generate FASTQ output in a path of your own choosing, instead of `flowcell_id/outs/fastq_path`.
--project	*(bcl2fastq option)* Custom project name, to override the samplesheet or to use in conjunction with the `--csv` argument.
--jobmode	*(Martian option)* Job manager to use. Valid options: `local` (default), `sge`, `lsf`, or a .template file.
--localcores	*(Martian option)* Set max cores the pipeline may request at one time. Only applies when `--jobmode=local`.
--localmem	*(Martian option)* Set max GB the pipeline may request at one time. Only applies when `--jobmode=local`.

Example Data

cellranger mkfastq recognizes two file formats for describing samples: a simple, three-column CSV format, and the Illumina Experiment Manager (IEM) sample sheet format used by `bcl2fastq`. There is an example below for running mkfastq with each format.

To follow along, do the following:

1. Download the tiny-bcl tar file.
2. Untar the tiny-bcl tar file in a convenient location. This will create a new `tiny-bcl` subdirectory.

3. Download the simple CSV layout file: cellranger-tiny-bcl-simple-1.2.0.csv.
4. Download the Illumina Experiment Manager sample sheet: cellranger-tiny-bcl-samplesheet-1.2.0.csv.

Running mkfastq with a Simple CSV Samplesheet

A simple csv samplesheet is recommended for most sequencing experiments. The simple csv format has only three columns (Lane, Sample, Index), and is thus less prone to formatting errors. You can see an example of this in `cellranger-tiny-bcl-simple-1.2.0.csv`:

```
Lane,Sample,Index
1,test_sample,SI-TT-D9
```

Here are the options for each column:

Aa Lane	☰ Which lane(s) of the flowcell to process. Can be either a single lane, a range (e.g., 2-4) or '*' for all lanes in the flowcell.
Sample	The name of the sample. This name is the prefix to all the generated FASTQs, and corresponds to the `--sample` argument in all downstream 10x pipelines.Sample names must conform to the Illumina `bcl2fastq` naming requirements. Only letters, numbers, underscores and hyphens area allowed; no other symbols, including dots (".") are allowed.
Index	The 10x sample index that was used in library construction, e.g., SI-TT-D9 or SI-GA-A1

To run `mkfastq` with a simple layout CSV, use the `--csv` argument. Here's how to run `mkfastq` on the `tiny-bcl` sequencing run with the simple layout:

```
$ cellranger mkfastq --id=tiny-bcl \
                     --run=/path/to/tiny_bcl \
                     --csv=cellranger-tiny-bcl-simple-1.2.0.csv

cellranger mkfastq
Copyright (c) 2019 10x Genomics, Inc.  All rights reserved.
-------------------------------------------------------------------------------
Martian Runtime - 6.1.2-v4.0.6
Running preflight checks (please wait)...
2019-11-14 16:33:54 [runtime] (ready)           ID.tiny-bcl.MAKE_FASTQS_CS.MAKE_FASTQS.PRE
PARE_SAMPLESHEET
```

```
2019-11-14 16:33:57 [runtime] (split_complete)  ID.tiny-bcl.MAKE_FASTQS_CS.MAKE_FASTQS.PRE
PARE_SAMPLESHEET
2019-11-14 16:33:57 [runtime] (run:local)       ID.tiny-bcl.MAKE_FASTQS_CS.MAKE_FASTQS.PRE
PARE_SAMPLESHEET.fork0.chnk0.main
2019-11-14 16:34:00 [runtime] (chunks_complete) ID.tiny-bcl.MAKE_FASTQS_CS.MAKE_FASTQS.PRE
PARE_SAMPLESHEET
...
```

Running mkfastq with an Illumina Experiment Manager Sample Sheet

The cellranger mkfastq pipeline can also be run with a samplesheet in the Illumina Experiment Manager (IEM) format. If you didn't sequence with sample indices, you'll need to use this format. Briefly look at `cellranger-tiny-bcl-samplesheet-1.2.0.csv` before running the pipeline. You will see a number of fields specific to running on Illumina platforms, and then a [Data] section. That section is where you put your sample, lane and index information. Here's an dual-indexing example:

```
[Data]
Lane,Sample_ID,Sample_Name,Sample_Plate,Sample_Well,I7_Index_ID,index,I5_Index_ID,index2,S
ample_Project,Description
1,s1,test_sample,,,SI-TT-D9,SI-TT-D9,SI-TT-D9,SI-TT-D9,p1,
```

Here, SI-TT-D9 refers to a 10x dual-index sample index.

In this example, only reads from lane 1 will be used. To demultiplex the given sample index across all lanes, omit the lanes column entirely.

Here's a single-indexing example:

```
[Data]
Lane,Sample_ID,index,Sample_Project
1,Sample1,SI-GA-A3,tiny-bcl
```

Here, SI-GA-A3 refers to a 10x single-index sample index, a set of four oligo sequences. cellranger mkfastq also supports listing oligo sequences explicitly.

Sample names must conform to the Illumina `bcl2fastq` naming requirements. Specifcally only letters, numbers, underscores and hyphens area allowed. No other symbols, including dots (.) are allowed.

Also note that while an authentic IEM sample sheet will contain other sections above the [Data] section, these are optional for demultiplexing. For demultiplexing an existing run with cellranger mkfastq, only the [Data] section is required.

Next, run the cellranger mkfastq pipeline, using the --samplesheet argument:

```
$ cellranger mkfastq --id=tiny-bcl \
                     --run=/path/to/tiny_bcl \
                     --samplesheet=cellranger-tiny-bcl-samplesheet-1.2.0.csv

cellranger mkfastq
Copyright (c) 2019 10x Genomics, Inc.  All rights reserved.
-------------------------------------------------------------------------------
Martian Runtime - 6.1.2-v4.0.6
Running preflight checks (please wait)...
2019-11-14 16:35:49 [runtime] (ready)           ID.tiny-bcl.MAKE_FASTQS_CS.MAKE_FASTQS.PRE
PARE_SAMPLESHEET
2019-11-14 16:35:52 [runtime] (split_complete)  ID.tiny-bcl.MAKE_FASTQS_CS.MAKE_FASTQS.PRE
PARE_SAMPLESHEET
2019-11-14 16:35:52 [runtime] (run:local)       ID.tiny-bcl.MAKE_FASTQS_CS.MAKE_FASTQS.PRE
PARE_SAMPLESHEET.fork0.chnk0.main
2019-11-14 16:35:58 [runtime] (chunks_complete) ID.tiny-bcl.MAKE_FASTQS_CS.MAKE_FASTQS.PRE
PARE_SAMPLESHEET
...
```

If you encounter any preflight errors, refer to the Troubleshooting page.

Checking FASTQ Output

Once the cellranger mkfastq pipeline has successfully completed, the output can be found in a new folder named with the value you provided to cellranger mkfastq in the `--id` option (if not specified, defaults to the name of the flowcell):

```
$ ls -l
drwxr-xr-x 4 jdoe  jdoe     4096 Nov 14 12:05 tiny-bcl
```

The key output files can be found in `outs/fastq_path`, and are organized in the same manner as a conventional `bcl2fastq` run:

```
$ ls -l tiny-bcl/outs/fastq_path/
drwxr-xr-x 3 jdoe jdoe         3 Nov  14 12:26 Reports
drwxr-xr-x 2 jdoe jdoe         8 Nov  14 12:26 Stats
drwxr-xr-x 3 jdoe jdoe         3 Nov  14 12:26 tiny-bcl
```

```
-rw-r--r-- 1 jdoe jdoe  20615106 Nov  14 12:26 Undetermined_S0_L001_I1_001.fastq.gz
-rw-r--r-- 1 jdoe jdoe  20615106 Nov  14 12:26 Undetermined_S0_L001_I2_001.fastq.gz
-rw-r--r-- 1 jdoe jdoe  51499694 Nov  14 12:26 Undetermined_S0_L001_R1_001.fastq.gz
-rw-r--r-- 1 jdoe jdoe 152692701 Nov  14 12:26 Undetermined_S0_L001_R2_001.fastq.gz

$ tree tiny-bcl/outs/fastq_path/tiny_bcl/
tiny-bcl/outs/fastq_path/tiny_bcl/
  Sample1
    Sample1_S1_L001_I1_001.fastq.gz
    Sample1_S1_L001_I2_001.fastq.gz
    Sample1_S1_L001_R1_001.fastq.gz
    Sample1_S1_L001_R2_001.fastq.gz
```

This example was produced with a sample sheet that included "tiny-bcl" as the Sample_Project, so the directory containing the sample folders is named tiny-bcl. If a Sample_Project wasn't specified, or if a simple layout CSV file was used (which does not have a Sample_Project column), the directory containing the sample folders would be named according to the flow cell ID instead.

If you want to remove the `Undetermined` FASTQs from the output to save space, you can run `mkfastq` with the `--delete-undetermined` flag. To see all cellranger mkfastq options, run cellranger mkfastq --help.

Troubleshooting

If you encounter a crash while running cellranger mkfastq, upload the tarball (with the extension .mri.tgz) in your output directory:

cellranger upload youremail@institution.edu jobid.mri.tgz

where jobid is what you input into the --id option of mkfastq (if not specified, defaults to the ID of the flowcell). This tarball contains numerous diagnostic logs that we can use for debugging.

You will receive an automated email from 10x Genomics. If not, email support@10xgenomics.com. For the fastest service, respond with the following:

- The exact command line you used.
- The sample sheet that you used.
- The RunInfo.xml and runParameters.xml files from your BCL directory.
- The kind of libraries you are demultiplexing (including chemistry).

Single-Library Analysis with cellranger count

Cell Ranger's pipelines analyze sequencing data produced from Chromium Single Cell Gene Expression. It also processes data generated by using Feature Barcode technology and/or Single Cell Targeted Gene Expression. The analysis involves the following steps:

1. Run `cellranger mkfastq` on the Illumina BCL output folder to generate FASTQ files.
2. Run cellranger count on each GEM well that was demultiplexed by cellranger mkfastq. For Targeted Gene Expression libraries, see Targeted Gene Expression Analysis for instructions on how to provide the target gene panel information. If you created a Feature Barcode library alongside the Gene Expression library, you will pass them both to cellranger count at this point. See Feature Barcode Analysis for details.
3. Optionally, run `cellranger aggr` to aggregate multiple GEM wells from a single experiment that were analyzed by cellranger count.
4. Optionally run `cellranger reanalyze` to re-run the secondary analysis on a library or aggregated set of libraries (i.e., PCA, t-SNE, and clustering) and be able to fine-tune parameters.

For the following example, assume that the Illumina BCL output is in a folder named `/sequencing/140101_D00123_0111_AHAWT7ADXX`.

Run cellranger mkfastq

First, follow the instructions on running cellranger mkfastq to generate FASTQ files. For example, if the flowcell serial number was `HAWT7ADXX`, then cellranger mkfastq will output FASTQ files in `HAWT7ADXX/outs/fastq_path`.

Run cellranger count

To generate single cell feature counts for a single library, run cellranger count with the following arguments. For a complete listing of the arguments accepted, see the Command Line Argument Reference below, or run cellranger count --help.

```
--expect-cells
```

```
--force-cells
```

Specifying Input FASTQ Files for 10x Pipelines

After determining these input arguments, run cellranger:

```
$ cd /home/jdoe/runs
$ cellranger count --id=sample345 \
                   --transcriptome=/opt/refdata-gex-GRCh38-2020-A \
                   --fastqs=/home/jdoe/runs/HAWT7ADXX/outs/fastq_path \
                   --sample=mysample \
                   --expect-cells=1000 \
                   --localcores=8 \
                   --localmem=64
```

Following a series of checks to validate input arguments, cellranger count pipeline stages will begin to run:

```
Martian Runtime - v4.0.6

Running preflight checks (please wait)...
Checking sample info...
Checking FASTQ folder...
Checking reference...
Checking optional arguments...
...
```

By default, cellranger will use all of the cores available on your system to execute pipeline stages. You can specify a different number of cores to use with the `--localcores` option; for example, `--localcores=16` will limit cellranger to using up to sixteen cores at once. Similarly, `--localmem` will restrict the amount of memory (in GB) used by cellranger.

The pipeline will create a new folder named with the sample ID you specified (e.g. `/home/jdoe/runs/sample345`) for its output. If this folder already exists, cellranger will assume it is an existing pipestance and attempt to resume running it.

Output Files

A successful cellranger count run should conclude with a message similar to this:

```
Outputs:
- Run summary HTML:                            /opt/sample345/outs/web_summary.html
- Run summary CSV:                             /opt/sample345/outs/metrics_summary.csv
- BAM:                                         /opt/sample345/outs/possorted_genome_bam.bam
- BAM index:                                   /opt/sample345/outs/possorted_genome_bam.bam.b
ai
- Filtered feature-barcode matrices MEX:       /opt/sample345/outs/filtered_feature_bc_matrix
- Filtered feature-barcode matrices HDF5:      /opt/sample345/outs/filtered_feature_bc_matri
x.h5
- Unfiltered feature-barcode matrices MEX:     /opt/sample345/outs/raw_feature_bc_matrix
- Unfiltered feature-barcode matrices HDF5:    /opt/sample345/outs/raw_feature_bc_matrix.h5
- Secondary analysis output CSV:               /opt/sample345/outs/analysis
- Per-molecule read information:               /opt/sample345/outs/molecule_info.h5
- CRISPR-specific analysis:                    null
- Loupe Browser file:                          /opt/sample345/outs/cloupe.cloupe
- Feature Reference:                           null
- Target Panel File:                           null
Waiting 6 seconds for UI to do final refresh.
Pipestance completed successfully!yyyy-mm-dd hh:mm:ss Shutting down.
Saving pipestance info to "tiny/tiny.mri.tgz"
```

The output of the pipeline will be contained in a folder named with the sample ID you specified (e.g. `sample345`). The subfolder named `outs` will contain the main pipeline output files:

File Name	Description
web_summary.html	Run summary metrics and charts in HTML format
metrics_summary.csv	Run summary metrics in CSV format
possorted_genome_bam.bam	BAM file containing both unaligned reads and reads aligned to the genome and transcriptome annotated with barcode information
possorted_genome_bam.bam.bai	Index for `possorted_genome_bam.bam`

File Name	Description
filtered_feature_bc_matrix	Filtered feature-barcode matrices containing only cellular barcodes in MEX format. (In Targeted Gene Expression samples, the non-targeted genes are not present.)
filtered_feature_bc_matrix_h5.h5	Filtered feature-barcode matrices containing only cellular barcodes in HDF5 format. (In Targeted Gene Expression samples, the non-targeted genes are not present.)
raw_feature_bc_matrices	Unfiltered feature-barcode matrices containing all barcodes in MEX format
raw_feature_bc_matrix_h5.h5	Unfiltered feature-barcode matrices containing all barcodes in HDF5 format
analysis	Secondary analysis data including dimensionality reduction, cell clustering, and differential expression
molecule_info.h5	Molecule-level information used by cellranger aggr to aggregate samples into larger datasets
cloupe.cloupe	Loupe Browser visualization and analysis file
feature_reference.csv	(Feature Barcode only) Feature Reference CSV file
target_panel.csv	(Targeted GEX only) Targed panel CSV file

Once cellranger count has successfully completed, you can browse the resulting summary HTML file in any supported web browser, open the .cloupe file in Loupe Browser, or refer to the Understanding Output section to explore the data by hand.

Command-Line Argument Reference

Argument	Description
--id	A unique run ID string: e.g. `sample345`

Argument	Description
--fastqs	Either:Path of the fastq_path folder generated by cellranger mkfastqe.g. `/home/jdoe/runs/HAWT7ADXX/outs/fastq_path`. This contains a directory hierarchy that cellranger count will automatically traverse.- OR -Any folder containing fastq files, for example if the fastq files were generated by a service provider and delivered outside the context of the mkfastq output directory structure.Can take multiple comma-separated paths, which is helpful if the same library was sequenced on multiple flowcells.Doing this will treat all reads from the library, across flowcells, as one sample.If you have multiple libraries for the sample, you will need to run cellranger count on them individually, and then combine them with cellranger aggr.This argument cannot be used when performing Feature Barcode analysis; use `--libraries` instead.
--libraries	Path to a `libraries.csv` file declaring FASTQ paths and library types of input libraries. Required for gene expression + feature barcode analysis. See Feature Barcode Analysis for details. When using this argument, `--fastqs` and `--sample` must not be passed.This argument should not be used when performing gene expression-only analysis; use `--fastqs` instead.
--sample	Sample name as specified in the sample sheet supplied to cellranger mkfastq.Can take multiple comma-separated values, which is helpful if the same library was sequenced on multiple flowcells and the sample name used (and therefore fastq file prefix) is not identical between them.Doing this will treat all reads from the library, across flowcells, as one sample.If you have multiple libraries for the sample, you will need to run cellranger count on them individually, and then combine them with cellranger aggr.Allowable characters in sample names are letters, numbers, hyphens, and underscores.
--transcriptome	Path to the Cell Ranger compatible transcriptome reference e.g. • For a human-only sample, use `/opt/refdata-gex-GRCh38-2020-A` • For a human and mouse mixture sample, use `/opt/refdata-gex-GRCh38-and-mm10-2020-A`
--feature-ref	Path to a Feature Reference CSV file declaring the Feature Barcode reagents in use in the experiment. Required for Feature Barcode analysis. See Feature Barcode Reference for details on how to construct the feature reference.
--target-panel	Path to a Target Panel CSV file declaring the target panel used, if any. Required for Targeted Gene Expression analysis. See Targeted Gene Expression Analysis for details
--no-target-umi-filter	(optional) Add this flag to disable targeted UMI filtering. See Targeted Algorithms for details.
--expect-cells	(optional, recommended) Expected number of recovered cells. Default: 3,000 cells.

Argument	Description
--force-cells	(optional) Force pipeline to use this number of cells, bypassing the cell detection algorithm. Use this if the number of cells estimated by Cell Ranger is not consistent with the barcode rank plot.
--include-introns	(optional) Add this flag to count reads mapping to intronic regions. This may improve sensitivity for samples with a significant amount of pre-mRNA molecules, such as nuclei. This flag should be used instead of the deprecated pre-mRNA reference.
--nosecondary	(optional) Add this flag to skip secondary analysis of the feature-barcode matrix (dimensionality reduction, clustering and visualization). Set this if you plan to use cellranger reanalyze or your own custom analysis.
--no-bam	(optional). Do not generate a bam file. Default: false.
--no-libraries	Proceed with processing using a --feature-ref but no feature-barcode data specified with the --libraries flag.
--chemistry	(optional) Assay configuration. NOTE: by default the assay configuration is detected automatically, which is the recommended mode. You should only specify chemistry if there is an error in automatic detection. Select one of: • `auto` for auto-detection (default), • `threeprime` for Single Cell 3′, • `fiveprime` for Single Cell 5′, • `SC3Pv2` for Single Cell 3′ v2, • `SC3Pv3` for Single Cell 3′ v3, • `SC3Pv3LT` for Single Cell 3′ v3 LT, • `SC3Pv3HT` for Single Cell 3′ v3 HT, • `SC5P-PE` for Single Cell 5′ paired-end (both R1 and R2 are used for alignment), • `SC5P-R2` for Single Cell 5′ R2-only (where only R2 is used for alignment). • `SC3Pv1` for Single Cell 3′ v1. NOTE: this mode cannot be auto-detected. It must be set explicitly with this option.
--r1-length	(optional) Hard-trim the input R1 sequence to this length. Note that the length includes the Barcode and UMI sequences so do not set this below 26 for Single Cell 3′ v2 or Single Cell 5′. This and `--r2-length` are useful for determining the optimal read length for sequencing.
--r2-length	(optional) Hard-trim the input R2 sequence to this length.
--lanes	(optional) Lanes associated with this sample
--localcores	Restricts cellranger to use specified number of cores to execute pipeline stages. By default, cellranger will use all of the cores available on your system.
--localmem	Restricts cellranger to use specified amount of memory (in GB) to execute pipeline stages. By default, cellranger will use 90% of the memory available on your system.