Takara Bio USA

# Trekker® Primary Analysis Pipeline User Manual

v1.4.11
(071426)

**Takara Bio USA, Inc.**
2560 Orchard Parkway, San Jose, CA 95131, USA
U.S. Technical Support: technical_support@takarabio.com

| United States/Canada | Asia Pacific | Europe | Japan |
|---|---|---|---|
| 800.662.2566 | +1.650.919.7300 | +33.(0)1.3904.6880 | +81.(0)77.565.6999 |

# Table of Contents

## Table of Figures



## Table of Tables

**IMPORTANT:**
If you are processing Trekker FX data, it is important that you begin with "Trekker FX Primary Pipeline Analysis Quick Start Guide" to learn about the *four* key analysis steps and refer to this extended user guide for detailed instructions for each step.

Failure to follow this sequence may result in suboptimal spatial results or pipeline failure.

# I. Introduction

The Trekker® Primary Analysis Pipeline (Figure 1) processes sequenced data from the Trekker Single-Cell Spatial Mapping Kits.

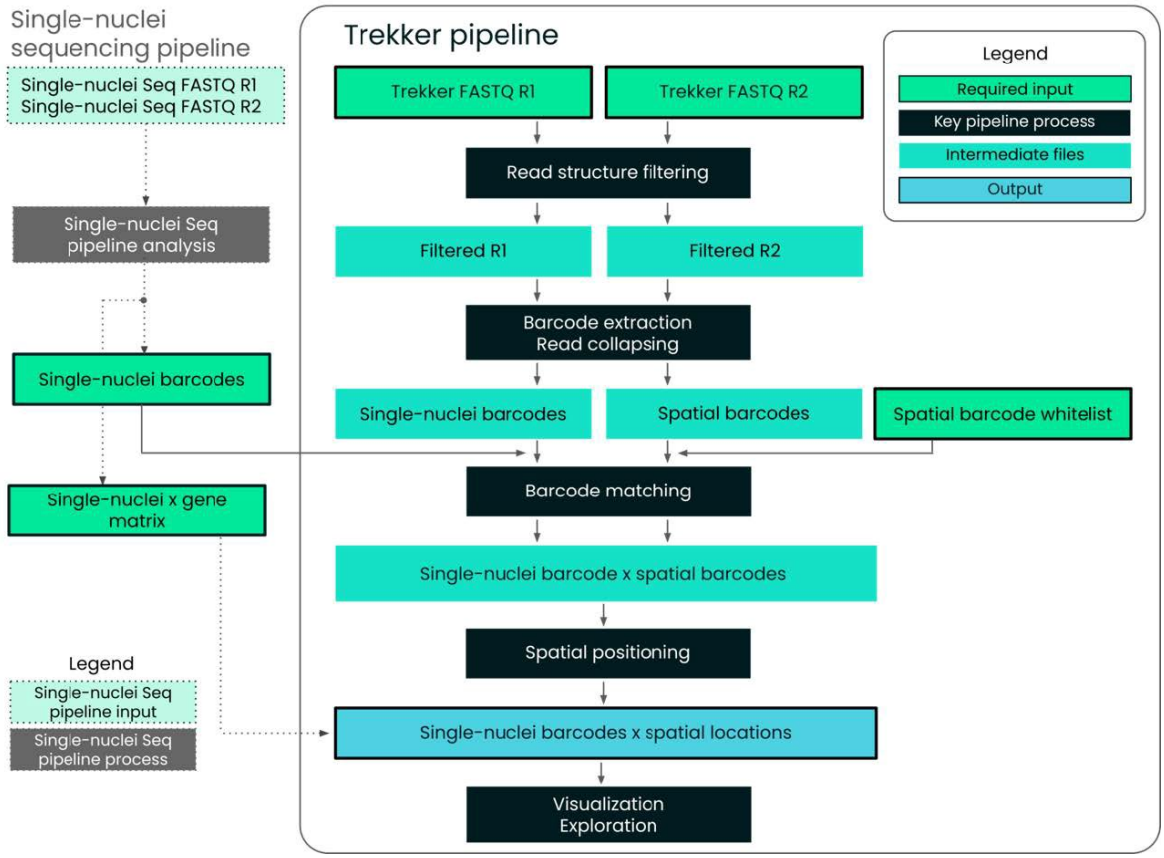


**Figure 1. Workflow of the Trekker Primary Analysis Pipeline (Trekker pipeline).**

The Trekker pipeline requires the following inputs:

- FASTQ pairs from the Trekker spatial barcode library
- Selected outputs from the bioinformatics pipeline of the single-nuclei sequencing library
- The bead barcode file for the Trekker tile used

- A sample sheet associating sample metadata with aforementioned inputs

The Trekker pipeline extracts single-nuclei barcodes and spatial barcodes from Trekker FASTQ pairs. It also retrieves UMIs (unique molecular identifiers) linked to each spatial barcode (Figure 2). After barcode matching, spatial positioning is done using the spatial barcodes associated with each single nucleus, resulting in a spatial location assigned to each single nucleus. The pipeline additionally conducts quality control to help determine the next steps.

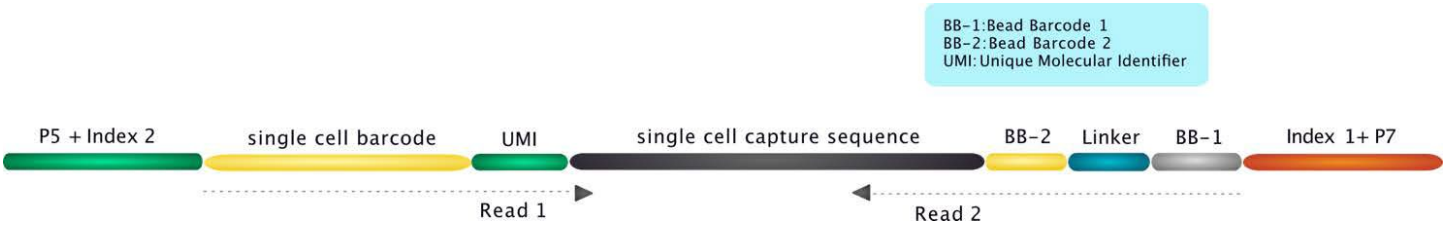


**Figure 2. Example read structure of the Trekker spatial barcode library FASTQ pairs.**

## A. Tools and Steps in the Pipeline

The Trekker pipeline's backbone is written in Bash. Its steps are written in Python and R, and its dependencies can be executed through Singularity (now Apptainer), Docker, and Conda.

The Trekker pipeline consists of the following major steps. It utilizes both publicly available and custom-written bioinformatic software.

- Read filtering and barcode matching: custom Python scripts
- Spatial positioning: custom R scripts and DBSCAN (https://www.kdnuggets.com/2020/04/dbscan-clustering-algorithm-machine-learning.html)
- Analysis and visualization: custom R scripts and Seurat (https://satijalab.org/seurat/)

## B. Ways to Run the Pipeline

The Trekker pipeline can be run in the following ways:

- Via installation on local workstations or clusters (Section II, "Trekker Pipeline Local Installation")
- Via cloud analysis

This document covers running the pipeline via local installation.

## C. Technical Support

If you have any questions related to the Trekker or Seeker kits, protocols, or data interpretation downstream of bioinformatics workflows, please email technical_support@takarabio.com.

# II. Trekker Pipeline Local Installation

## A. System Requirements

- Linux platforms with ≥256 GB RAM and 12 cores.

**NOTE:** The pipeline does NOT support Mac or Windows machines.

## B. Download trekker-v1.4.11

**NOTE:** For best results, use Adobe Acrobat Reader to view this user manual. If you are copying and pasting these commands, ensure that the commands include the correct punctuation (e.g., hyphens) before execution.

Use the following command to download trekker-v1.4.11.

```
wget https://takarabio.com/resourcedocument/x357434 -O - | tar -xzf -
```

Use the following command to navigate into the script folder.

```
cd trekker-*
```

## C. Set Up Output Path

Modify the bolded value shown below in `nuclei_locater_toplevel.sh` (in the `trekker-v1.4.11` script folder) to match the path on your platform.

```
#============== Define Top-Level Directory Path For Output ================

OUT_DIR=/home/trekker_out/
```

**NOTES:** See below for further details on the values you are modifying.

– `OUT_DIR`: Absolute path to a folder where the Trekker pipeline output should be directed to. (This is the top-level output folder. The pipeline will create sample-specific folders inside.)

## D. Set Up Dependencies

The pipeline can be executed through Docker, Singularity, or Conda.

### 1. Docker

1. Install Docker if it is not already installed on the server.
   Visit the Docker page for installation instructions.
2. Pull the Trekker Docker image, using the command below.

```
docker pull tbusaspatial/trekker:v1.4.11
```

### 2. Singularity

Install Singularity if not already installed.

Visit the Singularity page for installation instructions.

**NOTE:** You DO NOT need to download the Singularity image `trekker-v1.4.11.sif`. It is in folder `trekker-v1.4.11`.

### 3. Conda

**NOTE:** Steps 1 & 2 below are only required if you are using Conda to execute dependencies.

1. Create and activate a trekker conda environment using the `environment.yml` (in the `trekker-v1.4.11` script folder) we created for the pipeline.

```
cd trekker-*
conda env create -f environment.yml
conda activate trekker
```

**NOTE:** For faster execution, use `mamba env create -f environment.yml`.

2. Modify the bolded values in the two lines shown below in `nuclei_locater_conda.sh` (in the `trekker-v1.4.11` script folder) to match the paths on your platform.

```
#==================== Define Conda Paths ======================
PROFILE_CONDA_PATH=/home/tools/miniconda3/envs/trekker/
source /home/tools/miniconda3/etc/profile.d/conda.sh
```

**NOTES:** See below for further details on the values you are modifying.

- **`/home/tools/miniconda3/envs/trekker/`**: Replace this value with the absolute path to the trekker conda environment you created in Step l. To find the path, type:

  ```
  conda info --envs
  ```

  Copy and paste the path displayed next to trekker, as bolded below.

  ```
  trekker                  /home/tools/miniconda3/envs/trekker
  ```

- **`/home/tools/miniconda3/`**: Replace this value with the path to your miniconda installation. To find the path, type:

  ```
  conda info | grep -i 'base environment'
  ```

## E. Check Input Format

**IMPORTANT:**

If you are processing Trekker FX data, it is important to prepare both GEX outputs and Trekker FASTQ pairs in "partitioned" format. To learn how, start with the Trekker FX Primary Pipeline Analysis Quick Start Guide (Steps 1 and 2) and refer to this extended user manual for detailed instructions.

Failure to follow this sequence may result in pipeline failure or suboptimal spatial results.

### 1. Check if your Trekker input requires preprocessing

**IMPORTANT:**

- If your data comes from the following single-cell platforms, an additional preprocessing step is required to ensure your Trekker inputs are compatible with the Trekker pipeline.
  - 10x Chromium, GEM-X Flex (i.e. Trekker FX)
  - Parse, Evercode WT v3

  Use the links in Table 1 to access preprocessing instructions for your single-cell platform.

  Once preprocessing is complete, verify your *converted* Trekker input format, following the guidelines in Step 2.
- For data from all other single-cell platforms, no preprocessing is needed. Check your Trekker input format, following the guidelines in Step 2.

Table 1. Trekker input preprocessing requirements and instructions.

| Single-cell assay platform | Is preprocessing required? | Preprocessing instructions |
|---|---|---|
| **10x, GEM-X Flex (i.e. Trekker FX)** | Yes | Appendix A, Section A, "Preprocessing: Trekker FX" |
| **10x, Next GEM 3' v3.1** | No | – |
| **10x, GEM-X 3' v4** | No | – |
| **10x, GEM-X 5'** | No | – |
| **10x, Next GEM Multiome ATAC + Gene Expression** | No | – |
| **All BD Rhapsody platforms** | No | – |
| **Illumina, Single Cell 3′ RNA Prep (If processed by DRAGEN)** | No | – |
| **Illumina, Single Cell 3′ RNA Prep (if processed by PIPSeeker)** | No | – |
| **Parse, Evercode WT v3** | Yes | Appendix A, Section B, "Preprocessing: Parse Evercode" |

## 2. Check if your Trekker input formats conform to the following requirements

### a) *FASTQ Pairs from the Trekker Spatial Barcode Library*

**IMPORTANT:**

- If your tissue section was processed on a single Trekker tile but split into multiple single-nuclei reactions, i.e. processed in multiple:
  - 10x Chromium wells
  - BD Rhapsody lanes
  - Illumina PIPseq reactions
  - Parse sub-libraries (See Appendix A Section C "Preprocessing: Parse Evercode" for details)

  These reactions will have different sample indices during sequencing. **DO NOT** combine FASTQ files from these reactions. Process Trekker FASTQ pairs for each reaction and combine their pipeline outputs using the merging script from Appendix B, "Trekker Output Merger".
- The scenario above is different from sequencing a reaction in multiple sequencing lanes, for which you must concatenate FASTQ files from multiple lanes into a single Read 1 and a single Read 2.

- For each Trekker reaction, prepare *a single* Read 1 and *a single* Read 2 FASTQ in .gz format.

**NOTE:** Refer to Appendix C, Section A, "How to Concatenate Multiple FASTQ Files" for instructions on concatenating multiple FASTQ files.

- The FASTQ pairs must be in the compressed .gz format.

**NOTE:** Refer to Appendix C, Section B, "How to Compress a FASTQ File" for instructions on compressing a FASTQ file into .gz format.

- All reads in Trekker Read 1 FASTQ should have the same length and have the minimal length required by the single-nuclei bioinformatics pipeline.
- All reads in Trekker Read 2 FASTQ should have the same length and be at least 32 bp.
- (Trekker FX and Parse only) Make sure your FASTQ pairs have gone through preprocessing. Follow Table 2 to locate the preprocessed FASTQs to be fed into the Trekker pipeline.

**Table 2. Which Trekker FASTQ file to use for the Trekker pipeline, for single-cell platforms requiring preprocessing.**

| Single-cell assay platform | Trekker Read 1 FASTQ | Trekker Read 2 FASTQ |
|---|---|---|
| **10x, GEM-X Flex (i.e., Trekker FX)** | In the folder `TrekkerFX_FLEX_DEMUX_FASTQS` | |
| **Parse, Evercode WT v3** | In the folder defined by `--opath` in the preprocessing step | |

### b) *Selected Outputs from the Bioinformatics Pipeline of the Single-Nuclei Sequencing Library*

For each snRNA-seq reaction, generate the following three output files using the associated single-cell bioinformatics pipeline (and preprocessing script, in the case of Illumina Single Cell 3′ RNA Prep processed by PIPSeeker). Place them in a single folder.

– For Trekker FX data, the three files have the following names, and must be in "partitioned" format:

`barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`

**IMPORTANT:** To learn how to prepare these files in partitioned format, refer to Trekker FX Primary Pipeline Analysis Quick Start Guide (Step 1) and see in this manual Appendix C, Section C (How to Run Cell Ranger in Partitioned Mode).

– For all standard 10x, BD Rhapsody, and Illumina (processed by PIPSeeker) data, the three files have the following names:

`barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`

– For Illumina data processed by DRAGEN, the three files have the following names:

`${Sample}.scRNA.filtered.barcodes.tsv.gz`

`${Sample}.scRNA.filtered.features.tsv.gz`

`${Sample}.scRNA.filtered.matrix.mtx.gz`

– For Parse Evercode WT v3, the three files have the following names:

`cell_metadata.csv.gz, all_genes.csv.gz, count_matrix.mtx.gz`

**IMPORTANT:**

- Most single-cell pipelines generate these three files in unfiltered and "quality-filtered" versions. Use the "quality-filtered" version where high-quality nuclei were retained.
- Use Table 3 (next page) to locate these files, and check if they have the expected format.
- See example input to check if your files have the proper format.

**Table 3. Typical location and format for required single-cell pipeline outputs.**

| Single-cell assay platform | Which folder contains the three required files? | Typical format in `barcodes.tsv.gz` |
|---|---|---|
| All 10x Chromium platforms (except for GEM-X Flex) | `filtered_feature_bc_matrix` | `AAACCCAAGCAAATGT-1`<br>`AAACCCAAGGTAAGAG-1`<br>`AAACCCAAGTTTGTCG-1` |
| 10x, GEM-X Flex | `outs/per_sample_outs/${partition_id}/count/sample_filtered_feature_bc_matrix`<br>**NOTE:** See Appendix C, Section D "How to Locate Partitioned Cell Ranger Outputs for Trekker Pipeline?" for details. | `AAACAAGCACATAAACAGTAGGCT-1`<br>`AAACAAGCACCATCCAAGTAGGCT-1`<br>`AAACAAGCACTGACAAAGTAGGCT-1` |
| All BD Rhapsody platforms | `${Sample}_RSEC_MolsPerCell_MEX` | `10632`<br>`14108`<br>`14878` |
| Illumina, Single Cell 3' RNA Prep (Processed by DRAGEN) | `${Sample}` | `AAACAAACAAACACCTATGGAAGGATGA`<br>`AAACAAACAAACACTTAACGTGCACCAG`<br>`AAACAAACAAACCGGAAACCACTAATTG` |
| Illumina, Single Cell 3' RNA Prep (Processed by PIPSeeker) | `filtered_matrix/sensitivity_${n}, n=[1,5]` | `AAAAAAAACCATGAAT`<br>`AAAAAAAACTGACGAC`<br>`AAAAAAAAGGTCCGAT` |
| Parse, Evercode WT v3 | `DGE_filtered`<br>**NOTE:** DO NOT use the files from `output_combined` and `all-samples`, as this reduces positioning rate. See instructions in Appendix A, Section B.8, "Pairing Demuxed Trekker FASTQs" for which version to use. | See `cell_metadata.csv.gz` in Section II.J.1, "Download Example Inputs" (TrekkerQ_P) |

## F. Input

**IMPORTANT:**
If you are processing Trekker FX data, it is important to prepare both GEX outputs and Trekker FASTQ pairs in "partitioned" format. To learn how, start with the "Trekker FX Primary Pipeline Analysis Quick Start Guide" and refer to this extended user manual for detailed instructions.

Failure to follow this sequence may result in pipeline failure or suboptimal spatial results.

Once GEX output and Trekker FASTQ pairs are properly partitioned, please return to this section to prepare the remaining inputs.

**IMPORTANT:**

- If you are processing data from BD Rhapsody Single-Cell TCR/BCR Next + mRNA Whole Transcriptome Analysis (WTA) Kit (TrekkerU_RVDJ), skip this entire "Input" section, and prepare your input following the instructions in Appendix D, Section A, "Input Preparation: BD Rhapsody Single-Cell TCR/BCR Next".
- If you are processing data from BD Rhapsody Single-Cell ATAC-Seq + mRNA WTA Kit (TrekkerU_RATAC), skip this entire "Input" section, and prepare your input following the instructions in Appendix D, Section B, "Input Preparation: BD Rhapsody Single-Cell ATAC-Seq".
- For data from all other single-cell platforms, prepare your input following the instructions below.

Prepare the following inputs before triggering the pipeline.

**NOTE:** Before processing your own data, we recommend using example inputs to test if the pipeline is properly set up and if your files have the proper format.

### 1. Bead Barcode File

`${Tile_ID}_BeadBarcodes.txt`

Download the bead barcode file (e.g., `U0001_001_BeadBarcodes.txt`) using our Tile bead barcode file retrieval tool.

**NOTE:** Don't forget to unzip the downloaded file before use.

### 2. Trekker Paired FASTQ Files

**IMPORTANT:**
If you are processing Trekker FX data, it is important to split your Trekker FASTQ pair from a single 10x well into "partitions". Each partition is associated with a multiplexing barcode (e.g. AB001).

To learn how, read "Trekker FX Primary Pipeline Analysis Quick Start Guide.pdf" (Step 2) and see in this manual Appendix A, Section A (Preprocessing: Trekker FX) for detailed instruction.

Failure to partition will lead to sub-optimal spatial results.

Example file names:

```
MouseBrain_Trekker_R1.fastq.gz
MouseBrain_Trekker_R2.fastq.gz
```

**NOTE:** Don't forget to check the format conformity of these files (Section II.E, "Check Input Format"). Improper format will lead to pipeline failures.

### 3. Selected Outputs from the Bioinformatics Pipeline of the Single-Nuclei Sequencing Library

**IMPORTANT:**
If you are processing Trekker FX data, it is important to run Cell Ranger in "partitioned" mode and split your single-nuclei pipeline output into partitions. Each partition is associated with a multiplexing barcode (e.g. BC001).

To learn how, read "Trekker FX Primary Pipeline Analysis Quick Start Guide.pdf" (Step 1) and see in this manual Appendix C Section C (How to Run Cell Ranger in Partitioned Mode) for detailed instructions.

Failure to partition will lead to pipeline failure.

- For Trekker FX data, the three files have the following names, and must be in "partitioned" format. See message above for how to prepare them.
  `barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`

- For all standard 10x, BD Rhapsody, and Illumina (processed by PIPSeeker) data, the three files have the following names:
  `barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`
- For Illumina data processed by DRAGEN, the three files have the following names:
  `${Sample}.scRNA.filtered.barcodes.tsv.gz`
  `${Sample}.scRNA.filtered.features.tsv.gz`
  `${Sample}.scRNA.filtered.matrix.mtx.gz`
- For Parse Evercode WT v3, the three files have the following names:
  `cell_metadata.csv.gz, all_genes.csv.gz, count_matrix.mtx.gz`

**NOTE:** Don't forget to check the format conformity of these files (Section II.E, "Check Input Format"). Improper format will lead to pipeline failures.

### 4. Sample Sheet

`samplesheet.csv`

Download this example `samplesheet.csv` to use as a starting point to enter information on your sample to be analyzed.

**NOTES:**

- Prepare a separate `samplesheet.csv` for each reaction (i.e., a pair of Trekker FASTQs). The pipeline does NOT support multiple rows (reactions) in a sample sheet.
  - For Trekker FX data, prepare one sample sheet per partition.
- DO NOT swap the column orders in the sample sheet. Re-ordered columns will lead to pipeline failure.
- The box below (next page) shows an example of what the sample sheet file contents might look like. See Table 4 for a description of each column type.

```
sample,sc_sample,experiment_date,barcode_file,fastq_1,fastq_2,sc_outdir,sc_platform,prof
ile,subsample,cores

TrekkerU_C_MouseBrain,TrekkerU_C_MouseBrain_scRNAseq,20240916,/path/to/LTTag0053_003_Bea
dBarcodes.txt,/path/to/TrekkerU_C_Mouse_brain_R1_001.fastq.gz,/path/to/TrekkerU_C_Mouse_
brain_R2_001.fastq.gz,/path/to/TrekkerU_C_MouseBrain_scRNAseqOut/filtered_feature_bc_mat
rix/,TrekkerU_C,singularity,no,8
```

**Table 4. `samplesheet.csv` column names and descriptions.** Values are required for all columns.

| Column name | Description |
|---|---|
| sample | Prefix for naming the Trekker pipeline output. Avoid using '.' or space. |
| sc_sample | Prefix for book-keeping single-nuclei pipeline analysis output. (For book-keeping purposes only. Use a desired string with no space or special characters.) |
| experiment_date | Date of analysis in format YYYYMMDD. (This will be used as part of the output folder name.) |

| | |
|---|---|
| barcode_file | Absolute path to the Trekker bead barcode file. |
| fastq_1 | Absolute path to the Trekker FASTQ Read 1.<br>• For Trekker FX data, do not forget to partition Read 1 as described in Appendix A, Section A (Preprocessing: Trekker FX). |
| fastq_2 | Absolute path to the Trekker FASTQ Read 2.<br>• For Trekker FX data, do not forget to partition Read 2 as described in Appendix A, Section A (Preprocessing: Trekker FX). |
| sc_outdir | Absolute path to the folder containing the following three required single-nuclei pipeline analysis output.<br>• For Trekker FX data, the three files have the following names, and must be in "partitioned" format:<br>`barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`<br>**IMPORTANT:** To learn how to prepare these files in partitioned format, refer to "***Trekker FX Primary Pipeline Analysis Quick Start Guide.pdf***" (Step 1) and see in this manual Appendix C, Section C (How to Run Cell Ranger in Partitioned Mode). See also Appendix C, Section D (How to Locate Partitioned Cell Ranger Outputs for Trekker Pipeline?).<br>• For all standard 10x, BD Rhapsody, and Illumina (processed by PIPSeeker) data, the three files have the following names:<br>`barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`<br>• For Illumina data processed by DRAGEN, the three files have the following names:<br>`${Sample}.scRNA.filtered.barcodes.tsv.gz`<br>`${Sample}.scRNA.filtered.features.tsv.gz`<br>`${Sample}.scRNA.filtered.matrix.mtx.gz`<br>• For Parse Evercode WT v3, the three files have the following names:<br>`cell_metadata.csv.gz, all_genes.csv.gz, count_matrix.mtx.gz`<br>**IMPORTANT:** DO NOT use the files from `output_combined` and `all-samples`, as this reduces positioning rate. See instructions in Appendix A, Section C.8, "Pairing Demuxed Trekker FASTQs" for which version to use. |
| sc_platform | Select a value from the 2nd column of Table 5 (below) (e.g., `TrekkerU_C`), based on the single-cell assay platform you are using from the 1st column.<br>**IMPORTANT:** Make sure to choose the correct value. An incorrect value will lead to pipeline failure. |
| profile | How to execute dependencies. Choose among Docker, Singularity, or Conda based on what you set up in Section II.D, "Set Up Dependencies". Do not use quotes; case does not matter. |
| subsample | If subsampling should be performed for spatial positioning parameterization. Use 'no' unless you have >5 x $10^6$ nuclei to be positioned. Do not use quotes; case does not matter. |

| | |
|---|---|
| cores | Number of cores to use for the spatial positioning step. We recommend 8, which is sufficient for most cases. |

**Table 5. "sc_platform" values based on the single-cell assay platform being used.**

| Single-cell assay platform | "sc_platform" value |
|---|---|
| 10x Chromium, GEM-X Flex | `TrekkerFX_FLEX` |
| 10x Chromium, Next GEM 3′ v3.1, polyT capture | `TrekkerU_C` |
| 10x Chromium, GEM-X 3′ v4, polyT capture | `TrekkerU_CX` |
| BD Rhapsody, WTA, polyT capture | `TrekkerU_R` |
| 10x Chromium, Next GEM Multiome ATAC + Gene Expression | `TrekkerU_M` |
| 10x Chromium, 5’ | `Trekker5C_CX` |
| Illumina, Single Cell 3′ RNA Prep (Processed by DRAGEN) | `TrekkerU_IL` |
| Illumina, Single Cell 3′ RNA Prep (Processed by PIPSeeker) | `TrekkerU_PIP` |
| Parse, Evercode WT v3 | `TrekkerQ_P` |

## G. Trigger the Pipeline

Follow the example below to trigger the pipeline.

**NOTE:** Before processing your own data, we recommend using our example inputs to test if the pipeline is properly set up.

**IMPORTANT:**
If you are analyzing Trekker FX data, ensure that you have reviewed the “Trekker FX Primary Pipeline Analysis Quick Start Guide” and have correctly completed Steps **1** and **2** described there.

As described in the Quick Start Guide, GEX output and Trekker FASTQ pairs from a single 10x well must be split into “*partitions*”. Each partition is associated with a multiplexing barcode (e.g., BC001 for GEX and AB001 for Trekker FASTQs).

Each partition should then be processed separately using the command below.

Failure to partition the data will result in pipeline failure or suboptimal spatial results.

```
$ cd trekker-v1.4.11
bash nuclei_locater_toplevel.sh /path/to/samplesheet.csv
```

## H. Monitor the Run

The STDOUT (standard output) of a successful pipeline run will look similar to the text box below (next page):

```
Checking samplesheet...
'/home/TrekkerU_C_ExampleInput_MouseBrain/samplesheet.csv' exists.

Checking Trekker FASTQ R1...
'/home/TrekkerU_C_ExampleInput_MouseBrain/TrekkerU_C_Mouse_brain_R1_001.fastq.gz'
exists.

Checking Trekker FASTQ R2...
'/home/TrekkerU_C_ExampleInput_MouseBrain/TrekkerU_C_Mouse_brain_R2_001.fastq.gz'
exists.

Checking Trekker tile spatial barcode whitelist...
'/home/TrekkerU_C_ExampleInput_MouseBrain/LTTag0053_003_BeadBarcodes.txt' exists.

Checking nuclei subsampling option...
Nuclei subsampling option is set to: no

Checking the number of cores requested...
Number of cores requested is set to: 8

Checking snRNAseq outputs…
snRNAseq outputs located.

Starting Trekker Analysis (TrekkerU_C) using singularity

Start fastq_parser
fastq_parser Done

Start splitspatialbarcodes
splitspatialbarcodes Done

Start bead_matching
bead_matching Done

Start spatial
spatial Done

Start analysis
analysis Done

Start mismatch_analysis
mismatch_analysis Done

Start genmetrics
genmetrics Done

Start genreport
genreport Done
```

A log folder containing stepwise logs can be found in the folder where you stored your samplesheet.csv. Use the command below to access detailed logs.

```
$ cd /path/to/where-samplesheet.csv-is-stored
cd log/${sample}
ls
```

## I. Output

Trekker outputs are in

```
${OUT_DIR}/${experiment_date}_{sample_ID}/.
```

The folder `output` contains key output files for downstream analysis, as shown below. A detailed description of each key output file can be found in Section III, "Interpreting Trekker Pipeline Output". The other folders contain intermediate files and run summaries.

```
output
├── ${Sample_ID}_ConfPositioned_seurat_spatial.rds
├── ${Sample_ID}_ConfPositioned_anndata_matched.h5ad
├── ${Sample_ID}_MoleculesPer_ConfPositionedNuclei.mtx
├── ${Sample_ID}_barcodes_ConfPositionedNuclei.tsv
├── ${Sample_ID}_genes_ConfPositionedNuclei.tsv
├── ${Sample_ID}_Location_ConfPositionedNuclei.csv
├── ${Sample_ID}_Trekker_Report.html
├── ${Sample_ID}_summary_metrics.csv
├── ${Sample_ID}_variable_features_clusters.csv
└── ${Sample_ID}_variable_features_spatial_moransi.txt
├── coords_${Sample_ID}.txt
├── intermediates
└── plots
```

## J. Testing and Example Datasets

To test pipeline configuration, use our example inputs to trigger the pipeline and compare the results to our example output.

**NOTE:** For best results, use Adobe Acrobat Reader to view this user manual. If you are copying and pasting these commands, ensure that the commands include the correct punctuation (e.g., hyphens) before execution.

### 1. Download Example Inputs

Single-nuclei assay platform: TrekkerU_C

```
wget https://www.takarabio.com/resourcedocument/x353756 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_M

```
wget https://www.takarabio.com/resourcedocument/x353757 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_PIP

```
wget https://www.takarabio.com/resourcedocument/x357474 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_IL

```
wget https://www.takarabio.com/resourcedocument/x355191 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerQ_P

```
wget https://www.takarabio.com/resourcedocument/x353759 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_RATAC

```
wget https://www.takarabio.com/resourcedocument/x353760 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_RVDJ

```
wget https://www.takarabio.com/resourcedocument/x353761 -O - |
tar -xzf -
```

Single-nuclei assay platform: Trekker5C_CX

```
wget https://www.takarabio.com/resourcedocument/x355189 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerFX_FLEX

```
wget https://www.takarabio.com/resourcedocument/x355190 -O - |
tar -xzf -
```

## 2. Download Example Outputs

Single-nuclei assay platform: TrekkerU_C

```
wget https://www.takarabio.com/resourcedocument/x357439 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_M

```
wget https://www.takarabio.com/resourcedocument/x357440 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_PIP

```
wget https://www.takarabio.com/resourcedocument/x357441 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_IL

```
wget https://www.takarabio.com/resourcedocument/x357438 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerQ_P

```
wget https://www.takarabio.com/resourcedocument/x357442 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_RATAC

```
wget https://www.takarabio.com/resourcedocument/x357443 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerU_RVDJ

```
wget https://www.takarabio.com/resourcedocument/x357444 -O - |
tar -xzf -
```

Single-nuclei assay platform: Trekker5C_CX

```
wget https://www.takarabio.com/resourcedocument/x357445 -O - |
tar -xzf -
```

Single-nuclei assay platform: TrekkerFX_FLEX

```
wget https://www.takarabio.com/resourcedocument/x357437 -O - |
tar -xzf -
```

# III. Interpreting Trekker Pipeline Output

Trekker outputs can be found at `${OUT_DIR}/${experiment_date}_{sample_ID}/`.

In this section, you can find descriptions for:

- Key output files
- Quality metrics

Check out Section 0, "How Positioning Works" to understand how the Trekker pipeline positioned the nuclei bioinformatically.

## A. Definition of Key Output Files

Key output files are in folder `output`.

**Table 6. Files in folder `output`.**

| File | Definition |
|---|---|
| `${Sample_ID}_summary_metrics.csv` | Run quality metrics |
| `${Sample_ID}_Report.html` | Run quality report |
| `${Sample_ID}_Report_files` | Individual plots in ${Sample_ID}_ST_Report.html |
| `${Sample_ID}_ConfPositioned_seurat_spatial.rds` | SEURAT object with expression table, spatial coordinates, and clustering results for nuclei positioned with 1 spatial location |
| `${Sample_ID}_ConfPositioned_anndata_matched.h5ad` | H5AD object with expression table, spatial coordinates, and clustering results for nuclei positioned with 1 spatial location |
| `${Sample_ID}_MoleculesPerConfPositionedNuclei.mtx` | Expression table in sparse matrix format for nuclei positioned with 1 spatial location (column=cell barcodes, row=genes) |
| `${Sample_ID}_barcodesConfPositionedNuclei.tsv` | Cell barcodes for nuclei positioned with 1 spatial location |
| `${Sample_ID}_genesConfPositionedNuclei.tsv` | Genes for nuclei positioned with 1 spatial location |
| `${Sample_ID}_ConfPositionedNucleiLocation.csv` | Spatial coordinates for nuclei positioned with 1 spatial location |
| `${Sample_ID}_variable_features_clusters.txt` | Top cluster defining genes |
| `${Sample_ID}_variable_features_spatial_moransi.txt` | Top spatially variable genes |
| `coordinates_df_${Sample_ID}_ST.txt` | Spatial coordinates of all positioned nuclei regardless of positioning status |
| `plots` | A folder containing additional quality control plots |
| `intermediates` | A folder containing expression table, spatial coordinates for all nuclei (regardless of positioning status) and all positioned nuclei (regardless of positioning confidence) in Seurat and sparse matrix format. |

**Table 7. Files in folder `intermediates`.**

| File | Definition |
|---|---|
| `${Sample_ID}_Positioned_seurat_spatial.rds` | Seurat object with expression table, spatial coordinates, and clustering results for all positioned nuclei (including those with 2+ spatial locations) |
| `${Sample_ID}_Positioned_anndata_matched.h5ad` | H5AD object with expression table, spatial coordinates, and clustering results for all positioned nuclei (including those with 2+ spatial locations) |

| | |
|---|---|
| `${Sample_ID}_MoleculesPerPositionedNuclei.mtx` | Expression table in sparse matrix format for all positioned nuclei (including those with 2+ spatial locations, column=cell barcodes, row=genes) |
| `${Sample_ID}_barcodes_PositionedNuclei.tsv` | Cell barcodes for all positioned nuclei (including those with 2+ spatial locations) |
| `${Sample_ID}_genes_PositionedNuclei.tsv` | Genes for all positioned nuclei (including those with 2+ spatial locations) |
| `${Sample_ID}_PositionedNucleiLocation.csv` | Spatial coordinates for all positioned nuclei (including those with 2+ spatial locations) |
| `${Sample_ID}_seurat_spatial.rds` | Seurat object with expression table, spatial coordinates, and clustering results for all nuclei (including those without spatial locations) |
| `${Sample_ID}_anndata_matched.h5ad` | H5AD object with expression table, spatial coordinates, and clustering results for all nuclei (including those without spatial locations) |
| `${Sample_ID}_MoleculesPer_Nuclei.mtx` | Expression table in sparse matrix format for all nuclei (including those without spatial locations; column=cell barcodes, row=genes) |
| `${Sample_ID}_barcodes_Nuclei.tsv` | Cell barcodes for all nuclei (including those without spatial locations) |
| `${Sample_ID}_genes_Nuclei.tsv` | Genes for all nuclei (including those without spatial locations) |
| `${Sample_ID}_Location_Nuclei.csv` | Spatial coordinates for all nuclei (including those without spatial locations) |

## B. Definition and Interpretations of Quality Metrics

Below are definitions and interpretations for metrics in:

1. `${Sample_ID}_Trekker_Report.html`
2. `${Sample_ID}_summary_metrics.csv`

### 1. Metrics in `${Sample_ID}_Trekker_Report.html`

**Table 8. Metrics in `${Sample_ID}_Trekker_Report.html`.**

| Metric name | Definition | Interpretation and suggested actions |
|---|---|---|
| Min spatial barcodes used to locate a nucleus centroid | Minimum number of spatial barcodes used to assign 1 spatial location. This is the parameter minPts used for DBSCAN. For each sample, it is dynamically chosen within an expected range (10x Chromium 3' v3.1, 3' v4 [4,40]; 10x Chromium Multiome [4,26]; BD Rhapsody WTA, ATAC+WTA, VDJ+WTA [4,80]; Parse Evercode WT v3, Illumina 3' RNA Prep [4,60], 10x Chromium Flex [4,80]). The chosen number maximizes the metric "Pct nuclei confidently positioned". | – |
| Total nuclei from single-nuclei sequencing library | Total high-quality nuclei retained from single-nuclei pipeline analysis | If significantly lower or higher than expected, check if the threshold set by the single-nuclei analysis pipeline to retain high-quality nuclei is appropriate. Alternatively, check if nuclei counting was performed appropriately. If lower than expected, check if the assayed tissue had compromised quality. |
| Pct nuclei in Trekker library | % nuclei from single-nuclei pipeline analysis found in the Trekker library | If <95%, check your samplesheet.csv to see if a wrong single-nuclei assay platform was selected for the Trekker pipeline analysis, and if your Trekker FASTQ inputs and single-nuclei output are incorrectly paired and are not from the same group of nuclei. |
| Pct nuclei in Trekker library with valid spatial barcodes | % nuclei from single-nuclei pipeline analysis found in the Trekker library with R2 matched to the Trekker bead barcode file | If <50%, check your samplesheet.csv to see if a wrong tile ID was used for the Trekker pipeline analysis. |
| Nuclei positioned | Nuclei from single-nuclei pipeline analysis with at least 1 spatial location assigned | If low, check for noticeable bead loss during nuclei counting. Alternatively, check if your Trekker library has been sufficiently sequenced to reach saturation, using metric 'Median useful reads per nuclei' (10x Chromium 3' v3.1, 3' v4, Multiome, 5', FLEX; Parse Evercode WT v3; Illumina 3' RNA Prep ≥ 400, BD Rhapsody WTA, ATAC+WTA, VDJ+WTA ≥ 250). |

| Metric name | Definition | Interpretation and suggested actions |
|---|---|---|
| Nuclei confidently positioned | Nuclei from single-nuclei pipeline analysis with 1 spatial location assigned (singlets) | If low, check for noticeable bead loss or multiplets during nuclei counting. Alternatively, check if your Trekker library has reached saturation, using metric 'Median useful reads per nuclei' (10x Chromium 3' v3.1, 3' v4, Multiome, 5', FLEX; Parse Evercode WT v3; Illumina 3' RNA Prep ≥ 400, BD Rhapsody WTA, ATAC+WTA, VDJ+WTA ≥ 250). |
| Total readpairs in Trekker library | Total raw FASTQ read pairs in the Trekker library | If notably lower than expected, check sequencing run quality and if demultiplexing was performed appropriately. |
| Pct readpairs with proper structure | % Trekker spatial barcode reads with expected linker sequence (hamming dist ≤1) | If <80%, check your samplesheet.csv for a potential swap between Trekker R1 and R2 FASTQs or if a wrong primer was used during Trekker library preparation. |
| Pct readpairs matched to single nuclei barcodes with valid spatial barcodes | % properly structured, single-nuclei matched read pairs that matched to the Trekker bead barcode file | If <50%, check your samplesheet to see if a wrong tile ID was used for the Trekker pipeline analysis. |
| Median useful reads per nuclei | Median reads per nuclei that matched to nuclei barcodes, properly structured, and matched to the Trekker bead barcode file | If <400 for 10x Chromium 3' v3.1, 3' v4, Multiome, 5', FLEX; Parse Evercode WT v3; Illumina 3' RNA Prep or <250 for BD Rhapsody WTA, ATAC+WTA, VDJ+WTA and no severe bead loss was observed, consider performing additional sequencing to obtain more reads and reach saturation. |

### 2. Metrics in `${Sample_ID}_summary_metrics.csv`

**Table 9. Sample metadata and analysis parameters.**

| Metric | Definition | Expected range or values |
|---|---|---|
| Sample ID | Sample ID used for naming pipeline output files | User-defined |
| Single cell assay | Single-nuclei assay platform selected for pipeline analysis | TrekkerU_C, TrekkerU_CX, TrekkerU_R, TrekkerU_M, TrekkerU_IL, TrekkerU_PIP, TrekkerQ_P TrekkerU_RVDJ, TrekkerU_RATAC Trekker5C_CX, |

| Metric | Definition | Expected range or values |
|---|---|---|
| | | TrekkerFX_FLEX |
| Tile ID | Tile ID used for pipeline analysis | UXXXX_XXX |
| eps | Max radius allowed in a nucleus-associated cluster (defined by the pipeline) | 50 |
| Min spatial barcodes required to locate a nucleus centroid | Minimum number of spatial barcodes required to assign 1 spatial location. This is the parameter minPts used for DBSCAN. For each sample, it is dynamically chosen within an expected range (10x Chromium 3' v3.1, 3' v4 [4,40]; 10x Chromium Multiome [4,26]; BD Rhapsody WTA, ATAC+WTA, VDJ+WTA [4,80]; Parse Evercode WT v3, Illumina 3' RNA Prep, 10x Chromium 5' [4,60]; 10x Chromium Flex [4,80]). The chosen number maximizes the metric 'Pct nuclei positioned with 1 spatial location' | TrekkerU_C, TrekkerU_CX: [4,40]<br>TrekkerU_R,<br>TrekkerU_RVDJ,<br>TrekkerU_RATAC: every other number in [4,80]<br>TrekkerU_M: every other number in [4,26]<br>TrekkerU_PIP,<br>TrekkerU_IL,<br>TrekkerQ_P,<br>Trekker5C_CX: [4,60]<br>TrekkerFX_FLEX: every other number in [4,80] |
| Maximum UMI cutoff | Max number of transcripts allowed for a valid spatial barcode. Spatial barcodes with more transcripts than the cutoff will be removed from spatial positioning as they likely are from dislodged Trekker beads. | 256 |

**Table 10. Nuclei positioning metrics.**

| Metric | Definition | Expected range or values |
|---|---|---|
| Total nuclei from single-nuclei sequencing library | Total high-quality nuclei retained from single-nuclei pipeline analysis | Dependent on single-nuclei platform and experimental design |
| Nuclei from single-nuclei sequencing library found in Trekker library | Number of nuclei from single-nuclei pipeline analysis found in Trekker library | Dependent on total nuclei identified from single-nuclei pipeline analysis |
| Pct nuclei in Trekker library | % nuclei from single-nuclei pipeline analysis found in the Trekker library | >95% |
| Nuclei from single-nuclei sequencing library found in Trekker library with valid spatial barcodes | Number of nuclei from single-nuclei pipeline analysis both found in Trekker library and matched to the Trekker bead barcode file | Dependent on total nuclei identified from single-nuclei pipeline analysis |
| Pct nuclei in Trekker library with valid spatial barcodes | % nuclei from single-nuclei pipeline analysis found in the Trekker library with R2 matched to the Trekker bead barcode file | >95% |
| Nuclei positioned | Number of nuclei from single-nuclei pipeline analysis with at least one spatial location assigned | Dependent on total nuclei identified from single-nuclei pipeline analysis |

| Metric | Definition | Expected range or values |
|---|---|---|
| Pct nuclei positioned | % nuclei from single-nuclei pipeline analysis with at least 1 spatial location assigned | >60% |
| Nuclei confidently positioned | Nuclei from single-nuclei pipeline analysis with 1 spatial location assigned (singlets) | Dependent on total nuclei identified from single-nuclei pipeline analysis |
| Pct nuclei confidently positioned | % nuclei from single-nuclei pipeline analysis with 1 spatial location assigned (singlets) | >40% |

**Table 11. Spatial barcode library quality.**

| Metric | Definition | Expected range or values |
|---|---|---|
| Total readpairs in Trekker library | Total raw FASTQ read pairs in the Trekker library | Dependent on single-nuclei platform and experimental design |
| Readpairs with proper structure | Trekker spatial barcode reads with expected linker sequence (hamming dist ≤1) | Dependent on total input reads |
| Pct readpairs with proper structure | % Trekker spatial barcode reads with expected linker sequence (hamming dist ≤1) | >80% |
| Readpairs used for matching to single nuclei barcodes | Properly structured read pairs used for barcode matching with single nuclei barcodes | Dependent on total input reads |
| Readpairs matched single nuclei barcodes | Properly structured read pairs containing barcodes matched to single nuclei barcodes (hamming dist = 0) | Dependent on total input reads |
| Pct readpairs matched to single nuclei barcodes | % Trekker read pairs properly structured and matched to nuclei barcodes from single-nuclei pipeline analysis | >7% |
| Readpairs matched to single nuclei barcodes with valid spatial barcodes | Properly structured, single-nuclei matched read pairs that matched to the Trekker bead barcode file | Dependent on total input reads |
| Pct readpairs matched to single nuclei barcodes with valid spatial barcodes | % properly structured, single-nuclei matched read pairs that matched to the Trekker bead barcode file | >75% |
| Pct useful reads | % raw read pairs that are properly structured, matched to both nuclei barcodes from single-nuclei pipeline analysis and the Trekker bead barcode file | >6% |

**Table 12. Spatial barcode matching quality.**

| Metric | Definition | Expected range or values |
|---|---|---|
| Total sequenced spatial barcodes | Number of unique spatial barcodes identified | Dependent on total input reads |
| Sequenced spatial barcodes perfectly matched to whitelist | Number of spatial barcodes with exact match in the bead barcode file (hamming dist = 0) | Dependent on total input reads |

| Metric | Definition | Expected range or values |
|---|---|---|
| Sequenced spatial barcodes approximately matched to whitelist | Number of spatial barcodes matched to the bead barcode file (hamming dist = 1) | Dependent on total input reads |
| Pct valid spatial barcodes | % unique spatial barcodes matched to the bead barcode file (hamming dist ≤1) | >60% |

**Table 13. Spatial barcode library depth.** All expected range or values in this table are dependent on total input reads, tissue type, and single-nuclei platform.

| Metric | Definition |
|---|---|
| Median spatial barcodes per nuclei | Median valid spatial barcodes per nuclei |
| Mean spatial barcodes per nuclei | Mean valid spatial barcodes per nuclei |
| Median spatial barcodes UMI per nuclei | Median number of molecules of valid spatial barcodes per nuclei |
| Mean spatial barcodes UMI per nuclei | Mean number of molecules of valid spatial barcodes per nuclei |
| Median reads per nuclei | Median reads per nuclei that matched to nuclei barcodes and are properly structured |
| Mean reads per nuclei | Mean reads per nuclei that matched to nuclei barcodes and are properly structured |
| Median useful reads per nuclei | Median reads per nuclei that matched to nuclei barcodes, properly structured, and matched to the Trekker bead barcode file |
| Mean useful reads per nuclei | Mean reads per nuclei that matched to nuclei barcodes, properly structured, and matched to the Trekker bead barcode file |

**Table 14. Signal to noise.**

| Metric | Definition | Expected range or values |
|---|---|---|
| Median proportion unique SB top cluster total | Median proportion of total spatial barcodes that contributed to spatial coordinate assignment per nucleus for all positioned nuclei | >0.02 |
| Mean proportion unique SB top cluster total | Mean proportion of total spatial barcodes that contributed to spatial coordinate assignment per nucleus for all positioned nuclei | >0.04 |
| Median proportion SB UMI top cluster total | Median proportion of total spatial barcode transcripts that contributed to spatial coordinate assignment per nucleus for all positioned nuclei | >0.04 |
| Mean proportion SB UMI top cluster total | Mean proportion of total spatial barcode transcripts that contributed to spatial coordinate assignment per nucleus for all positioned nuclei | >0.07 |
| Median mapped cells proportion unique SB top cluster | Median proportion of total spatial barcodes that contributed to spatial coordinate assignment per nucleus for confidently positioned nuclei | >0.05 |

| Metric | Definition | Expected range or values |
|---|---|---|
| Mean mapped cells proportion unique SB top cluster | Mean proportion of total spatial barcodes that contributed to spatial coordinate assignment per nucleus for confidently positioned nuclei | >0.06 |
| Median mapped cells proportion SB UMI top cluster | Median proportion of total spatial barcode transcripts that contributed to spatial coordinate assignment per nucleus for confidently positioned nuclei | >0.07 |
| Mean mapped cells proportion SB UMI top cluster | Mean proportion of total spatial barcode transcripts that contributed to spatial coordinate assignment per nucleus for confidently positioned nuclei | >0.1 |

## C. How Positioning Works

### 1. How Are the Nuclei Positioned Bioinformatically?

After nuclei are bioinformatically associated with their spatial barcodes, DBSCAN (Ester et al. 1996) is used to filter out the noise from the spatial barcodes before the nuclei are spatially positioned.

For each nucleus, DBSCAN clusters the spatial barcodes in their spatial coordinates and assigns a Cluster ID to each spatial barcode. DBSCAN identifies clusters based on density and assigns two types of Cluster IDs:

1. **Cluster ID >0**

   Spatial barcodes assigned have discrete spatial clustering. They appear in high-density regions in the spatial coordinates and are signal spatial barcodes denoting the nucleus' true spatial location.

2. **Cluster ID = 0**

   Spatial barcodes assigned have no clear spatial distribution. They appear in low-density regions in the spatial coordinates and are denoted as noise.

For nuclei with non-noise DBSCAN cluster (Cluster ID >0) identified, the pipeline filters out the noise spatial barcodes (Cluster ID = 0) and computes a UMI-weighted centroid for spatial barcodes in the cluster with Cluster ID = 1. This centroid is the nucleus' spatial coordinate.

### 2. Categories in Nuclei Positioning

The Trekker Primary Analysis Pipeline categorizes nuclei positioning based on the number of non-noise DBSCAN clusters (Cluster ID >0) assigned.

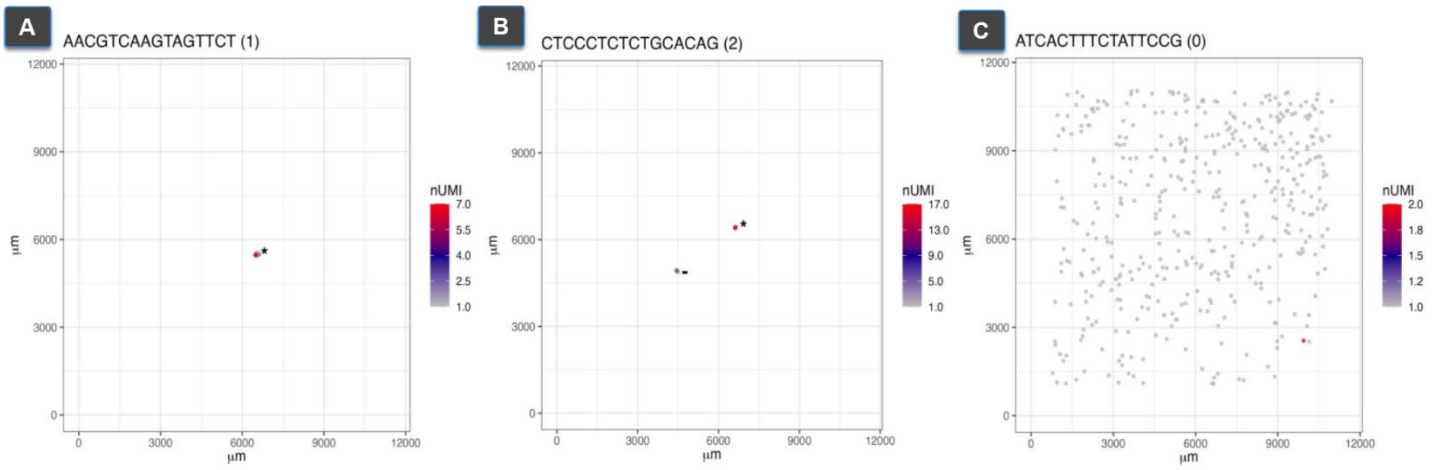


**Figure 3. Examples of nuclei positioned with 1, 2, or no spatial locations.** Each plot represents the spatial barcodes bioinformatically associated with a single nucleus. Each nucleus was assigned to one (1) spatial location (**Panel A**), two (2) spatial locations (**Panel B**), or no spatial location since only noise spatial barcodes were identified (**Panel C**). Each dot is a spatial barcode colored by the number of unique molecular identifiers (nUMI). Spatial barcodes are plotted in the spatial coordinates in μm. "*" points to the cluster with Cluster ID = 1, and "–" points to the cluster with Cluster ID >1.

1. "Nuclei positioned with 1 spatial location": Nuclei with 1 non-noise cluster assigned (Figure 3, Panel A)
2. "Nuclei positioned with 2+ spatial locations": Nuclei with >1 non-noise cluster assigned (Figure 3, Panel B). These nuclei are usually from one of the three scenarios. Nuclei under scenarios b. and c. below are salvageable for spatial positioning.
   a. Two or more nuclei were captured in the same droplet or well
   b. A nucleus and a Trekker bead were captured in the same droplet or well
   c. A DBSCAN cluster was called as two clusters due to parameterization artifacts
3. "Nuclei positioned" = Nuclei in category 1 + Nuclei in category 2
4. We do not assign spatial coordinates to nuclei with only noise cluster (Cluster ID = 0) identified (Figure 3, Panel C)

DBSCAN requires two parameters as input: minPts and eps. To determine the optimal parameter set for each Trekker run, the pipeline iterates through different minPts within a range (e.g., 4–15) under a constant eps (50) and chooses the minPts that leads to the highest proportion of nuclei that are positioned with one spatial location (with one non-noise DBSCAN cluster assigned).

## IV. References

Ester, M., Kriegel, H., Sander, J., et al. A density-based algorithm for discovering clusters in large spatial databases with noise. in *KDD* **96,** 226–231 (1996).

# Appendix A. Preprocessing

## A. Preprocessing: Partition Trekker FX FASTQ Pair

### 1. Overview

If you are processing data generated using the Trekker FX Single-Cell Spatial Mapping Kit or other Trekker kits paired with GEM-X Flex Gene Expression Reagent Kits for Multiplexed Samples, you will need to split your Trekker FASTQ pairs into partitions, if cells in a single 10x well were split into multiple partitions. Each partition is associated with a multiplexing barcode (e.g., AB001 for FLEX v1 and A-A01 for FLEX APEX).

Follow the instructions below to split your Trekker FASTQ pairs into partitions using the *trekkerFX_demux-v1.2.0.py* script. This script parses paired Trekker Read 1 and Read 2 FASTQ files and splits reads to individual partitions based on the partition-specific feature barcode sequences in Read 2. The output consists of per-partition Trekker FASTQ pairs, which serve as the required input for the Trekker Primary Analysis Pipeline.

After successful partitioning, use the partition-specific FASTQ files as input to the Trekker Primary Analysis Pipeline, as outlined in Section II, "Trekker Pipeline Local Installation".

**IMPORTANT:** Use **"TrekkerFX_FLEX"** as the value for **sc_platform** in your samplesheet.csv when running the main Trekker pipeline.

**IMPORTANT:** You also need to split the GEX output from a single 10x well into partitions. For details, refer to *"*Trekker FX Primary Pipeline Analysis Quick Start Guide*"* (Step 1) and see in this manual Appendix C, Section C (How to Run Cell Ranger in Partitioned Mode).

### 2. System Requirements

Python >= 3.6, >= 128 GB RAM, >= 2 CPU cores

### 3. Access the partitioning script

You can find the script in the trekker-v1.4.11 codebase, as below.

Navigate to the script directory:

```
cd trekker-v1.4.11/common/TrekkerFX_FLEX
```

Script:

```
trekkerFX_demux-v1.3.py
```

### 4. Input

Prepare the following inputs before triggering the partitioner

#### *a. Multiplexed Trekker FASTQ files*

```
Multiplexed_Trekker_R1.fastq.gz
Multiplexed_Trekker_R2.fastq.gz
```

#### *b. Sample sheet* partitions.csv

```
partitions.csv
```

Use this example partitions.csv as a starting point. Also see below for an explanation of its content.

```
Sample1, AB005
Sample2, AB006
Sample3, AB007
Sample4, AB008
```

**Notes on partitions.csv**

- **Format**
  - This file contains the sample name to multiplex label translation for your multiplexed experiment
  - This file should be in comma-delimited format (.csv)
  - This file should **NOT** contain a header
  - This file should contain two columns
- **Content**
  - In the example above, there are 4 samples (Sample1, Sample2, Sample3, Sample4), each paired with a multiplex label (AB005, AB006, AB007, AB008), respectively.
  - Column 1: Sample name for each sample in the multiplexed library, one row per sample.
    - Use only alphanumeric, underscore, or hyphen characters. Any other unusual special characters will result in demultiplexing failures.
  - Column 2: The multiplex label for each sample name in column 1.
    - For FLEX v1: choose from one of the 16 labels: AB001 to AB016.
    - For FLEX APEX: choose from one of the 384 labels: e.g., A-A01, where A is the Set ID and A01 is the well ID.

## 5. Trigger the partitioner

Follow the example below to trigger the partitioner.

**NOTE:** Before processing your own data, we recommend using our example input to test if the partitioner is properly setup.

If you used FLEX v1 chemistry:

```
python trekkerFX_demux-v1.3.py /path/to/Multiplexed_Trekker_R1.fastq.gz
/path/to/Multiplexed_Trekker_R2.fastq.gz /path/to/partitions.csv v1
```

If you used FLEX APEX chemistry:

```
python trekkerFX_demux-v1.3.py /path/to/Multiplexed_Trekker_R1.fastq.gz
/path/to/Multiplexed_Trekker_R2.fastq.gz /path/to/partitions.csv v2
```

**IMPORTANT:** Please use **ABSOLUTE PATHS** for all files. Using relative paths may lead to error or premature script termination.

**IMPORTANT:** If a *TrekkerFX_FLEX_DEMUX_FASTQS* folder already exists from a previous demultiplexing run, a new run in the same directory will overwrite all files with the same name, including `trekkerFX_demux_v1.3.log`, `metrics.csv` and any `fastq.gz` files.

**NOTE:** Go to this section to see additional notes on triggering the partitioner.

## 6. Monitor the run

STDOUT of a successful run will look similar to the example below.

STDOUT will also be saved in a log file (see 'Output' section).

```
Total_sequences:1000000
Exact_matches:876551
Percent_exact_matches:87.66
No_matches:123449
Percent_no_matches:12.34
Sample_ID,Barcode_ID,Reads,Pct_reads
AB001,AB001,2,0.00
AB002,AB002,12,0.00
AB003,AB003,3,0.00
AB004,AB004,2,0.00
Sample1,AB005,80111,9.14
Sample2,AB006,130000,14.83
Sample3,AB007,408051,46.55
Sample4,AB008,258346,29.47
AB009,AB009,2,0.00
AB010,AB010,6,0.00
AB011,AB011,7,0.00
AB012,AB012,2,0.00
AB013,AB013,1,0.00
AB014,AB014,2,0.00
AB015,AB015,0,0.00
AB016,AB016,4,0.00
```

## 7. Output

A *TrekkerFX_FLEX_DEMUX_FASTQS* folder will be created in the directory where you run the partitioner. This folder contains the following output files:

```
Demultiplexed sample-specific Trekker FASTQ pairs
metrics.csv
trekkerFX_demux_v1.3.log
```

## 8. Testing and example datasets

To test partitioner configuration, use our example inputs to trigger the partitioner and compare the results to our example output.

### *a. Download example inputs*

```
wget https://www.takarabio.com/resourcedocument/x357118 -O - |
tar -xzf -
```

### *b. Download example outputs*

```
wget https://www.takarabio.com/resourcedocument/x355186 -O - |
tar -xzf -
```

## 9. Output definition

Detailed descriptions of the output files are provided below:

### *a. Demultiplexed partition-specific Trekker FASTQ pairs*

The trekkerFX_demux.py script generates a pair of R1/R2 .fastq.gz files for each partition associated with a multiplex label. For the example partitions.csv file shown earlier, this results in eight FASTQ files:

```
Sample1_R1.fastq.gz
Sample1_R2.fastq.gz
Sample2_R1.fastq.gz
Sample2_R2.fastq.gz
Sample3_R1.fastq.gz
Sample3_R2.fastq.gz
Sample4_R1.fastq.gz
Sample4_R2.fastq.gz
```

The trekkerFX_demux.py script outputs demultiplexed FASTQ pairs only for samples that have a specified sample name in the partitions.csv file or for multiplex labels that account for at least 1% of total reads. The read percentage for each of the 16 multiplex labels is reported in the metrics.csv file.

**IMPORTANT:** If a *`TrekkerFX_FLEX_DEMUX_FASTQS`* folder already exists from a previous demultiplexing run, a new run in the same directory will overwrite all files with the same name, including `trekkerFX_demux_v1.3.log`, `metrics.csv` and any `fastq.gz` files.

### *b. metrics.csv*

| Metric | Definition |
|---|---|
| Total_sequences | Total read pairs parsed from the input multiplexed Trekker FASTQs |
| Exact_matches | Read pairs containing a barcode that exactly matches an expected multiplex barcode |
| Percent_exact_matches | % Read pairs that are exact matches |
| No_matches | Read pairs that do not match any expected multiplex barcode |
| Percent_no_matches | % Read pairs that have no matches |
| Sample_ID | User defined sample names from the partitions.csv file |
| Barcode_ID | Multiplex barcode ID (i.e., AB001–AB016) |
| Reads | Number of read pairs that matched to the multiplex barcode shown in that row |

| Metric | Definition |
|---|---|
| Pct_reads | % Total read pairs matched to the multiplex barcode shown in that row |

### Example metrics.csv file:

In this metrics.csv example, there are assorted read and match statistics, along with detailed metrics for each of the multiplex labels (AB001–AB016 for FLEX v1, and one of the 384 possible labels for FLEX APEX). Four primary samples were included in this demultiplexing run, and the table lists both the read counts and the percentage of total reads assigned to each sample.

```
Total_sequences:1000000
Exact_matches:876551
Percent_exact_matches:87.66
No_matches:123449
Percent_no_matches:12.34
Sample_ID,Barcode_ID,Reads,Pct_reads
AB001,AB001,2,0.00
AB002,AB002,12,0.00
AB003,AB003,3,0.00
AB004,AB004,2,0.00
Sample1,AB005,80111,9.14
Sample2,AB006,130000,14.83
Sample3,AB007,408051,46.55
Sample4,AB008,258346,29.47
AB009,AB009,2,0.00
AB010,AB010,6,0.00
AB011,AB011,7,0.00
AB012,AB012,2,0.00
AB013,AB013,1,0.00
AB014,AB014,2,0.00
AB015,AB015,0,0.00
AB016,AB016,4,0.00
```

### *c. trekkerFX_demux_v1.3.log*

The trekkerFX_demux_v1.3.log file contains assorted information printed during the demultiplexing process (identical to STDOUT). In most cases, you will not need to review this file. However, if you encounter issues while running trekkerFX_demux.py, the log may provide useful details to help you troubleshoot.

## 10. Evaluate partitioning success

Once partitioning completes, it is important to review the metrics.csv file to verify your Trekker reads were partitioned correctly.

Below is an example of the demultiplexing metrics generated from the sample inputs.

### *a) Example metrics.csv of a successful partitioning run*

Note how each labeled sample has a high Pct_reads value (Sample1: 9.14%, Sample2: 14.83%, Sample3: 46.55%, Sample4: 29.47%) and each unlabeled sample (AB001, AB002) has a low Pct_reads value (0.00%).

```
Total_sequences:1000000
Exact_matches:876551
Percent_exact_matches:87.66
No_matches:123449
Percent_no_matches:12.34
Sample_ID,Barcode_ID,Reads,Pct_reads
AB001,AB001,2,0.00
AB002,AB002,12,0.00
AB003,AB003,3,0.00
AB004,AB004,2,0.00
Sample1,AB005,80111,9.14
Sample2,AB006,130000,14.83
Sample3,AB007,408051,46.55
Sample4,AB008,258346,29.47
AB009,AB009,2,0.00
AB010,AB010,6,0.00
AB011,AB011,7,0.00
AB012,AB012,2,0.00
AB013,AB013,1,0.00
AB014,AB014,2,0.00
AB015,AB015,0,0.00
AB016,AB016,4,0.00
```

### *b) Example metrics.csv of a partitioning run with an improperly configured partitions.csv file*

In this example, there were no matches for Sample2 (AB006), however, there was a large proportion of matches for AB015. In this case, it is likely that Sample2 should have been paired with AB015 in the partitions.csv file (instead of AB006).

```
Total_sequences:1000000
Exact_matches:876551
Percent_exact_matches:87.66
No_matches:123449
Percent_no_matches:12.34
Sample_ID,Barcode_ID,Reads,Pct_reads
AB001,AB001,2,0.00
AB002,AB002,12,0.00
AB003,AB003,3,0.00
AB004,AB004,2,0.00
Sample1,AB005,80111,9.14
Sample2,AB006,0,0.00
Sample3,AB007,408051,46.55
Sample4,AB008,258346,29.47
AB009,AB009,2,0.00
AB010,AB010,6,0.00
AB011,AB011,7,0.00
AB012,AB012,2,0.00
AB013,AB013,1,0.00
AB014,AB014,2,0.00
AB015,AB015,130000,14.83
AB016,AB016,4,0.00
```

## 11. Troubleshooting

For most issues involving invalid inputs or execution errors, the following message will be printed to both STDOUT and the log file:

```
Usage: python trekkerFX_demux.py <R1.fastq.gz> <R2.fastq.gz>
<sample_file.csv>

Usage: if you want to run the script without a sample file, enter 'NA'
as the third argument
```

The following is a list of the most commonly encountered problems.

### a. Invalid input file path

This error occurs when the specified fastq.gz file(s) or partitions.csv cannot be found. The error message will indicate which file is missing.

**Example:**

```
Invalid input file: <file_name>
```

### b. Invalid fastq.gz file name

This error occurs when the fastq.gz file names do not include “R1” or “R2,” or when they do not end with “fastq.gz.” This safeguard prevents triggering the script with R1 and R2 reversed.

**Example:**

```
First fastq.gz file must contain "R1" or "r1" the name

Second fastq.gz file must contain "R2" or "r2" the name
```

### c. Invalid sample name or multiplex label in partitions.csv

This error occurs when the partitions.csv file contains an empty sample name or multiplex label. The example error below illustrates both cases.

**Example:**

```
Empty barcode label or sample name in row: 'Sample1_rep1,'

Empty barcode label or sample name in row: ',AB001'
```

### d. Duplicate barcode label in partitions.csv

This error occurs when the partitions.csv file contains duplicate barcode labels.

```
Sample1_rep1,AB001
Sample2_rep1,AB001
```

**Example:**

```
Duplicate barcode label AB001 found in sample file: partitions.csv
```

### e. Duplicate sample name in partitions.csv

This error occurs when the partitions.csv file contains duplicate sample names.

**Example:**

```
Duplicate sample name Sample1 found in sample file: partitions.csv
```

### f. Unexpected multiples label in partitions.csv

This error occurs when the partitions.csv file contains a multiplex label that is not one of the 16 expected labels for FLEX v1 (AB001–AB016) or one of the 384 labels for FLEX APEX.

**Example:**

```
Barcode label AA001 not expected in sample file: partitions.csv
```

### g. Improperly formatted partitions.csv

This error occurs when a row in the partitions.csv file contains unexpected or unrecognized formatting. As shown in the example below, this may result from a missing comma between the sample name and barcode label, empty rows, or other invalid formats. In these cases, the script skips the affected rows and applies the default barcode labels (e.g., AB001–AB016 for FLEX v1) when naming the associated demultiplexed FASTQ files.

**Example:**

```
initialize_samples|WARNING: Incorrect formatting in row:
'Sample1AB001'
```

## 12. Additional notes on triggering the partitioner

**a. If the required information for partitions.csv is not available.**

If you lost the association between sample names and multiplex labels, and cannot supply a partitions.csv, use 'NA' as the 3rd argument, as shown below.

```
python trekkerFX_demux_v3.py
/path/to/Multiplexed_Trekker_R1.fastq.gz
/path/to/Multiplexed_Trekker_R2.fastq.gz NA
```

## B. Preprocessing: Parse Evercode WT v3 Kit (TrekkerQ_P)

### 1. Overview

If you are processing data generated from the Parse Evercode WT v3 Kit (TrekkerQ_P), you will need to demultiplex your Trekker FASTQ pairs into sample-specific FASTQ files (Figure 4). These sample-specific FASTQ pairs are the required input for the Trekker Primary Analysis Pipeline.

This script demutiplexes paired Read 1 and Read 2 from a Trekker library, using a "sample" barcode defined by a sample's well coordinates in the first plate of Parse assay. The script outputs sample-specific Trekker FASTQ files.

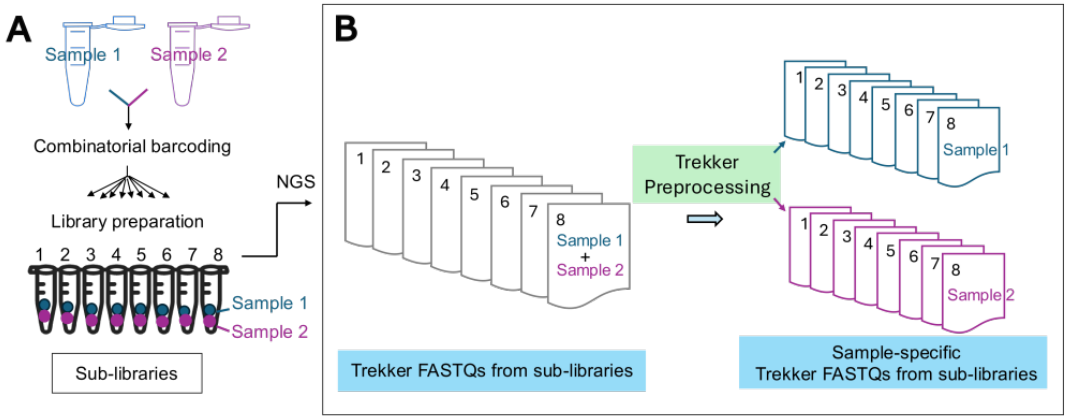


**Figure 4. Illustration of how Trekker preprocessing step demultiplexes Trekker FASTQs to sample-specific Trekker FASTQs. Panel A.** Single-cell workflow. **Panel B.** Trekker FASTQ preprocessing schematics.

After successful demultiplexing, use sample-specific FASTQ files as input for running the Trekker Primary Analysis Pipeline, as outlined in Section II, "Trekker Pipeline Local Installation".

Follow the instructions below to demultiplex your Trekker FASTQ pairs using the fastq_sep_groups_v0.5.py script.

### 2. System Requirements

- A Linux workstation with Python >= 3.10 or greater installed.
- Python module "pandas". If you don't have this module installed, you can run pip install pandas.

Read more on the Parse support page.

### 3. Access the script

You can find the script in the trekker-v1.4.11 codebase, as below.

Navigate to the script directory:

```
cd trekker-v1.4.11/common/TrekkerQ_P
```

Script:

`fastq_sep_groups_v0.5.py`

## 4. Input

Prepare the following inputs before triggering the demultiplexer.

1. Multiplexed Trekker paired FASTQ files, for each sub-library.

**IMPORTANT:** For each Trekker+Parse run, you will likely have multiple Trekker sub-libraries, leading to multiple Trekker FASTQ pairs (defined by different Illumina sample indices). If so, you must demultiplex each pair individually.

Example file names:

```
Multiplexed_TrekkerQ_P_HumanBreastCancer_SL1_R1_001.fastq.gz
Multiplexed_TrekkerQ_P_HumanBreastCancer_SL1_R2_001.fastq.gz
Multiplexed_TrekkerQ_P_HumanBreastCancer_SL2_R1_001.fastq.gz
Multiplexed_TrekkerQ_P_HumanBreastCancer_SL2_R2_001.fastq.gz
```

## 5. Trigger the Demultiplexer

Follow the example below to trigger the demultiplexer.

**NOTE:** Before processing your own data, we recommend using our example inputs to test if the converter is properly set up.

```
python fastq_sep_groups_v0.5.py \
   --chemistry v3 \
   --fq1
/home/TrekkerQ_P_DEMUX_example_input/Multiplexed_TrekkerQ_P_HumanBrea
stCancer_SL1_R1_001.fastq.gz \
   --fq2
/home/TrekkerQ_P_DEMUX_example_input/Multiplexed_TrekkerQ_P_HumanBrea
stCancer_SL1_R2_001.fastq.gz \
   --opath /home/TrekkerQ_P_DEMUX_example_output/ \
   --group HBC1 A1-A12,B1-B12 \
   --group HBC2 C1-C12,D1-D12


python fastq_sep_groups_v0.5.py \
   --chemistry v3 \
   --fq1
/home/TrekkerQ_P_DEMUX_example_input/Multiplexed_TrekkerQ_P_HumanBrea
stCancer_SL2_R1_001.fastq.gz \
   --fq2
/home/TrekkerQ_P_DEMUX_example_input/Multiplexed_TrekkerQ_P_HumanBrea
stCancer_SL2_R2_001.fastq.gz \
   --opath /home/TrekkerQ_P_DEMUX_example_output/ \
   --group HBC1 A1-A12,B1-B12 \
   --group HBC2 C1-C12,D1-D12
```

**NOTES:** See below for further details on command arguments:

Read more about each input argument on the Parse support page.

- `chemistry`: Chemistry used for the Parse assay.
- `fq1`: Absolute path to Trekker FASTQ Read 1 without sample demultiplexing.
- `fq2`: Absolute path to Trekker FASTQ Read 2 without sample demultiplexing.
- `opath`: Absolute path to the desired output directory.
- `group`: Each `group` defines a sample by a desired name (in this example, we have used `HBC1` and `HBC2`) and the sample's well coordinates in the first plate of the Parse assay. Well coordinates can be separated by a comma (,).

## 6. Monitor the Run

The STDOUT of a successful run will look similar to the example output below (next page):

```
# Set amplicon sequence for chemistry {args.chemistry}
# Start time 2025-07-11 13:33:18.878248
# Initializing...
# Loaded barcode data (9 bc sets)
# Sampling 500000 reads from
/home/TrekkerQ_P_DEMUX_example_input/Multiplexed_TrekkerQ_P_HumanBreastCancer_SL1_R2_001
.fastq.gz
# Collected 500000 data from 600000 fastq records
# Loaded fastq R2 sample
# Scoring fastq data against 3 kits, chemistry v3
#   WT_mini (n26_R1_v3_3) = 0.134
#   WT (n107_R1_v3_3) = 0.694
#   WT_mega (n222_R1_v3_3) = 0.431
# Best scoring kit = WT, 0.694
# Fasta R1
/home/TrekkerQ_P_DEMUX_example_input/Multiplexed_TrekkerQ_P_HumanBreastCancer_SL1_R1_001
.fastq.gz
#       R2
/home/TrekkerQ_P_DEMUX_example_input/Multiplexed_TrekkerQ_P_HumanBreastCancer_SL1_R2_001
.fastq.gz
# Read range    No restrictions
# Kit           WT
# Barcodes      ['n107_R1_v3_3', 'v1', 'R3_v3']
# Amplicon      NNNNNNNNNN33333333ATGAGGGGTCAG22222222TCCAACCACCTC11111111
# Max edit dist 2
# Matching      Permissive first barcode matches
# Out path      ./Demux_out/
# Out base      Multiplexed_TrekkerQ_P_HumanBreastCancer_SL1_group
#
# Subsets       2 groups (by wells)
#   HBC1   A1-A12,B1-B12      (24): [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15,
16, 17, 18, 19, 20, 21, 22, 23, 24]
#   HBC2   C1-C12,D1-D12      (24): [25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37,
38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48]
#
```

## 7. Output

The demultiplexed Trekker FASTQ pairs are written to directory specified by the `--opath` parameter when executing the demux command.

```
/home/TrekkerQ_P_DEMUX_example_output/
```

## 8. Pairing Demuxed Trekker FASTQ with Parse Output for Trekker Pipeline Run

In Parse's Trailmaker output, each sample has multiple versions of three files below with varying multiplex levels. For Trekker, use the versions demultiplexed by both sequencing index and sample (Figure 5).

```
cell_metadata.csv.gz, all_genes.csv.gz, count_matrix.mtx.gz
```

**NOTE:** DO NOT use the files from `output_combined` and `all-samples`, as this reduces positioning rate.

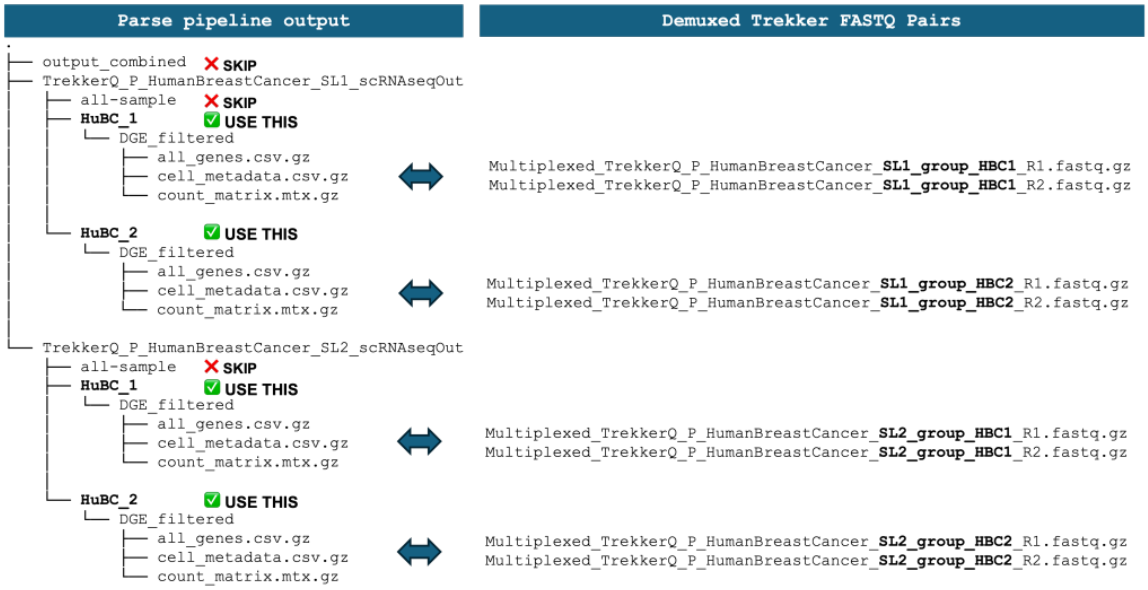


**Figure 5. Illustration of how to pair demultiplexed Trekker FASTQs with Parse' pipeline output.**

After running Trekker pipeline on the demuxed data, use the merging script from Appendix B, "Trekker Output Merger" to combine outputs for samples tagged with the same Trekker tile.

## 9. Testing and Example Datasets (TrekkerQ_P)

To test the demultiplexer configuration, use our example inputs to trigger the demultiplexer and compare the results to our example output.

**NOTE:** For best results, use Adobe Acrobat Reader to view this user manual. If you are copying and pasting these commands, ensure that the commands include the correct punctuation (e.g., hyphens) before execution.

1. Download example inputs.

```
wget https://www.takarabio.com/resourcedocument/x353772 -O - |
tar -xzf -
```

2. Download example outputs.

```
wget https://www.takarabio.com/resourcedocument/x353773 -O - |
tar -xzf -
```

# Appendix B: Trekker Output Merger

If your tissue section was processed on a single Trekker tile but split into multiple single-nuclei reactions (e.g. processed in multiple 10x Chromium wells or BD Rhapsody lanes with different sample indices during sequencing), you can process the Trekker FASTQ pairs for each reaction and combine their pipeline outputs.

**NOTE:** The scenario above is different from sequencing a reaction in multiple sequencing lanes. In this case, concatenate FASTQ files from multiple lanes into a single Read l and a single Read 2, and process the concatenated FASTQ pair through the Trekker pipeline. Refer to Appendix C, "How to" for instructions.

## A. Download trekker_merger-v1.4.11

**NOTE:** For best results, use Adobe Acrobat Reader to view this user manual. If you are copying and pasting these commands, ensure that the commands include the correct punctuation (e.g., hyphens) before execution.

Use the following command to download trekker_merger-v1.4.11.

```
wget https://takarabio.com/resourcedocument/x357435 -O - | tar -xzf -
```

Use the following command to navigate into the appropriate folder.

```
cd trekker_merger*
```

## B. Set Up Dependencies

The merger can be executed through Docker, Singularity, or Conda.

### 1. Docker

- If you have already pulled the trekker docker image (tbusaspatial/trekker:v1.4.11) for your Trekker pipeline, you don't need to do anything. (You pulled it here.)
- If you have not pulled the docker image, follow steps below to pull it.
  1. Install docker if not already. Visit the docker page for installation instructions.
  2. Pull the trekker docker image, using the command below.

```
docker pull tbusaspatial/trekker:v1.4.11
```

### 2. Singularity

- If you used Singularity to execute dependencies for your Trekker pipeline, you don't need to do anything.
- If you did not use Singularity for your Trekker pipeline dependencies, make sure you have the singularity software installed.

  Install singularity if you have not already. Visit the singularity page for installation instructions.

**NOTE:** You DO NOT need to download the singularity image `trekker-v1.4.11.sif`. It is in the folder `trekker_merger-v1.4.11`.

### 3. Conda

**NOTE:** Step 2 is required, regardless of whether you skipped Step 1.

- If you used Conda to execute dependencies for your Trekker pipeline, you can use the same environment for the merger and skip Step 1. You created the environment here.
- If you did not use Conda for your Trekker pipeline dependencies, follow Step 1 to create the environment.
  1. You can create and activate a Conda environment trekker using the `environment.yml` (in the `trekker-v1.4.11` script folder) we created for the pipeline.

```
cd  trekker-*
conda env create -f environment.yml
conda activate trekker
```

  2. Modify the bolded values in the two lines shown below (next page) in `trekker_merger.sh` (in the `trekker_merger-v1.4.11` script folder) to match the paths on your platform.

```
#==================== Define Conda Paths ======================
PROFILE_CONDA_PATH=/home/tools/miniconda3/envs/trekker/
CONDA_SH_PATH=/home/tools/miniconda3/etc/profile.d/conda.sh
```

**NOTES:** See below for further details on the values you are modifying.

- **/home/tools/miniconda3/envs/trekker/**: Replace this value with the absolute path to the trekker conda environment you created in Step l. To find the path, type:

  ```
  conda info --envs
  ```

  Copy and paste the path displayed next to trekker, as bolded below.

  ```
  trekker                  /home/tools/miniconda3/envs/trekker
  ```

- **/home/tools/miniconda3/:** Replace this value with the path to your miniconda installation. To find the path, type:

  ```
  conda info | grep -i 'base environment'
  ```

## C. Check Input File Completeness

Selected trekker-v1.4.11 outputs for all samples to be merged are required and must conform to the following requirement:

For each output, the following two files should be present and in the following folder structure.

```
/path_to/output/intermediates/${SAMPLE_ID}_seurat_spatial.rds
/path_to/output/${SAMPLE_ID}_summary_metrics.csv
```

## D. Input

Prepare the following inputs before triggering the merger.

**NOTE:** Before processing your own data, we recommend using example inputs to test if the pipeline is properly set up and if your files have the proper format.

1. Selected trekker-v1.4.11 pipeline outputs for all outputs to be merged

   **NOTE:** Don't forget to check the completeness of the outputs. Missing files will lead to merger failures.

2. Merger sample sheet

   ```
   merger_samplesheet.csv
   ```

   Use this example merger_samplesheet.csv as a starting point.

   

   **IMPORTANT:** DO NOT use Excel to edit the sample sheet. Doing so will lead to unwanted modification to the 'new line' token and merger failure. Use a text editor (e.g., Vim or equivalent).

Columns in a sample sheet include:

```
sample,out_dir
TrekkerR_MouseEmbryo_LaneA,/path/to/20241020_TrekkerR_MouseEmbryo_LaneA/trekker_TrekkerR_
MouseEmbryo_LaneA/output/
TrekkerR_MouseEmbryo_LaneB,/path/to/20241020_TrekkerR_MouseEmbryo_LaneB/trekker_TrekkerR_
MouseEmbryo_LaneB/output/
TrekkerR_MouseEmbryo_LaneC,/path/to/20241020_TrekkerR_MouseEmbryo_LaneC/trekker_TrekkerR_
MouseEmbryo_LaneC/output/
```

**NOTES:** See below for further details on columns in a sample sheet.

- `sample`: Prefixes used for naming the Trekker pipeline output for each sample.

  This value is used to locate the two required input files (above). If you named the file like

  ```
  /output/${SAMPLE_ID}_summary_metrics.csv
  ```

  you should use

  ```
  ${SAMPLE_ID}
  ```

  for the value of `sample`.
  This is also the value for `sample` you used in Section II.F.4, "Sample Sheet".
- `out_dir`: Absolute path to the trekker-v1.4.11 pipeline folder `output` for each sample.
- DO NOT swap the column orders in the sample sheet. Re-ordered columns will lead to merger failure.

## E. Trigger the Merger

Follow the example below to trigger the merger. Modify the bolded values below to match the paths and files on your platform. Before processing your own data, we recommend using our example inputs to test if the merger is properly set up.

```
$ cd trekker_merger-v1.4.11
bash trekker_merger.sh \
/path/to/samplesheet.csv \
${output_dir} \
${sample_ID} \
${profile}
```

**IMPORTANT:** Please replace `/path/to/samplesheet.csv` with the absolute PATH to `samplesheet.csv` on your platform.

**NOTES:** See below for further details on input arguments.

- `/path/to/samplesheet.csv`: Absolute path to the samplesheet.csv.
- `${output_dir}`: Absolute path to a folder where the merger output should be directed to. If you used a value like this:

  ```
  /home/merger_out/MouseEmbryo
  ```

the base directory `/home/merger_out/` should exist. The merger will create folder `MouseEmbryo`.

- `${sample_ID}`: Prefix for naming the final Trekker merged output. Avoid using '.' or space.
- `${profile}`: How to execute dependencies. Choose among Docker, Singularity, or Conda based on what you set up in Section II.D, "Set Up Dependencies". (Do not use quotes. Case does not matter.)

Example command:

```
$ cd trekker_merger-v1.4.11
bash trekker_merger.sh \
     /home/trekker/samplesheet.csv \
     /home/trekker/merger_out/MouseEmbryo/ \
     MouseEmbryo \
     singularity
```

## F. Monitor the Run

The STDOUT of a successful pipeline run will look similar to the example below (next page):

```
Checking samplesheet... '/home/trekker_merger_example_input/samplesheet.csv' exists

Checking samplesheet format

Samplesheet is in Linux format.

Checking Output directory path...
'/home/merger_out exists'

Checking Output Sample ID prefix...

Using SAMPLE_ID: MouseEmbryo

TrekkerR_MouseEmbryo_LaneA

seurat_file and metrics_file for TrekkerR_MouseEmbryo_LaneA are copied to a temporary
input directory

TrekkerR_MouseEmbryo_LaneB

seurat_file and metrics_file for TrekkerR_MouseEmbryo_LaneB are copied to a temporary
input directory

TrekkerR_MouseEmbryo_LaneC

seurat_file and metrics_file for TrekkerR_MouseEmbryo_LaneC are copied to a temporary
input directory

All samples in the samplesheet have the same Trekker Tile ID: LTTag0050_015 and are from
the same Single_cell_assay platform: TrekkerR

Starting trekker_merger

Running the trekker_merger using 'docker'

Start mergeSeurats

mergeSeurats Done

Start mergeMetrics

mergeMetrics Done

Start genreport

genreport Done

Your trekker output has been merged. Please find the results in
/home/merger_out/MouseEmbryo
```

A log folder containing stepwise logs can be found in the folder where you stored your `samplesheet.csv`. Use the command below to access detailed logs, modifying the bolded values to match the paths and files on your platform.

```
$ cd /path/to/where-samplesheet.csv-is-stored
cd log/${sample_ID}
ls
```

## G. Output

The merger outputs are in `${output_dir}`. It contains key output files for downstream analysis, as shown below. Detailed descriptions of each key output file can be found in Section III, "Interpreting Trekker Pipeline Output".

```
├─── intermediates
├─── ${sample_ID}_Report.html
├─── ${sample_ID}_summary_metrics_merged.csv
├─── ${sample_ID}_ConfPositioned_anndata_merged.h5ad
├─── ${sample_ID}_ConfPositioned_seurat_spatial_merged.rds
├─── ${sample_ID}_MoleculesPer_ConfPositionedNuclei_merged.mtx
├─── ${sample_ID}_barcodes_ConfPositionedNuclei_merged.tsv
├─── ${sample_ID}_genes_ConfPositionedNuclei_merged.tsv
├─── ${sample_ID}_Location_ConfPositionedNuclei_merged.csv
├─── ${sample_ID}_variable_features_clusters_merged.csv
└─── ${sample_ID}_variable_features_spatial_moransi_merged.txt
```

## H. Testing and Example Datasets (Merger)

To test pipeline configuration, use our example inputs to trigger the merger and compare the results to our example outputs.

**NOTE:** For best results, use Adobe Acrobat Reader to view this user manual. If you are copying and pasting these commands, ensure that the commands include the correct punctuation (e.g., hyphens) before execution.

1. Download example inputs

   ```
   wget https://www.takarabio.com/resourcedocument/x357117 -O - |
   tar -xzf -
   ```

2. Download example outputs

   ```
   wget https://www.takarabio.com/resourcedocument/x357436 -O - |
   tar -xzf -
   ```

# Appendix C. How to

## A. How to Concatenate Multiple FASTQ Files into One FASTQ

In a terminal, type the following command:

```
cat ${s1_1}_R1.fastq.gz ${s1_2}_R1.fastq.gz > ${sample1}_R1.fastq.gz
```

**NOTE:** Don't forget to replace `${s1_*}` and `${sample1}` with the prefixes of your own FASTQs

In the command above, three `R1.fastq.gz` files are concatenated into one FASTQ `${sample1}_R1.fastq.gz`. You can concatenate as many FASTQs as needed by appending file names to the cat command before >.

## B. How to Compress a FASTQ File into .gz Format

In a terminal, type the following command:

```
gzip ${sample1}.fastq
```

Replace **`${sample1}`** with the prefix of your own FASTQ.

## C. How to Run Cell Ranger in Partitioned Mode

To run Cell Ranger in partitioned mode, configure the **[samples]** section in config.csv so each row represents a multiplexing barcode-defined partition, as shown in the example below.

```
[gene-expression],,
reference,/mnt/references/cellranger_references/refdata-gex-GRCh38-2020-A,
probe-set,/mnt/tools/cellranger-x.x.x/probe_sets/Chromium_Human_Transcriptome_Probe_Set_v1.0.1_GRCh38-2020-A.csv,
create-bam,FALSE,
,,
[libraries],,
fastq_id,fastqs,feature_types
Pool1_GEX,/mnt/data/20260401_Pool1,Gene Expression
,,
[samples],,
sample_id,probe_barcode_ids,
Tile1_BC001_GEX,BC001,
Tile1_BC002_GEX,BC002,
Tile1_BC003_GEX,BC003,
Tile2_BC004_GEX,BC004,
Tile2_BC005_GEX,BC005,
Tile3_BC016_GEX,BC016,
```

**Figure 6. An example of correctly configured config.csv file to run Cell Ranger in partitioned mode.**

You should NOT combine partitions from the same Trekker tile / tissue section into a single row. Below represents an incorrect config.csv file.

```
[gene-expression],,
reference,/mnt/references/cellranger_references/refdata-gex-GRCh38-2020-A,
probe-set,/mnt/tools/cellranger-x.x.x/probe_sets/Chromium_Human_Transcriptome_Probe_Set_v1.0.1_GRCh38-2020-A.csv,
create-bam,FALSE,
,,
[libraries],,
fastq_id,fastqs,feature_types
Pool1_GEX,/mnt/data/20260401_Pool1,Gene Expression
,,
[samples],,
sample_id,probe_barcode_ids,
Tile1_GEX,BC001|BC002|BC003,
Tile2_GEX,BC004|BC005,
Tile3_GEX,BC016,
```

Figure 7. An example of incorrectly configured config.csv file to run Cell Ranger in partitioned mode.

## D. How to Locate Partitioned Cell Ranger Outputs for Trekker Pipeline?

For running the Trekker Pipeline in step 3, *filtered* and *partitioned* cell count matrix should be used, for filling 'sc_outdir' value in the samplesheet.csv.

*Filtered* and *partitioned* cell count matrix can be located in the following path:

/outs/per_sample_outs/${partition_id}/count/sample_filtered_feature_bc_matrix

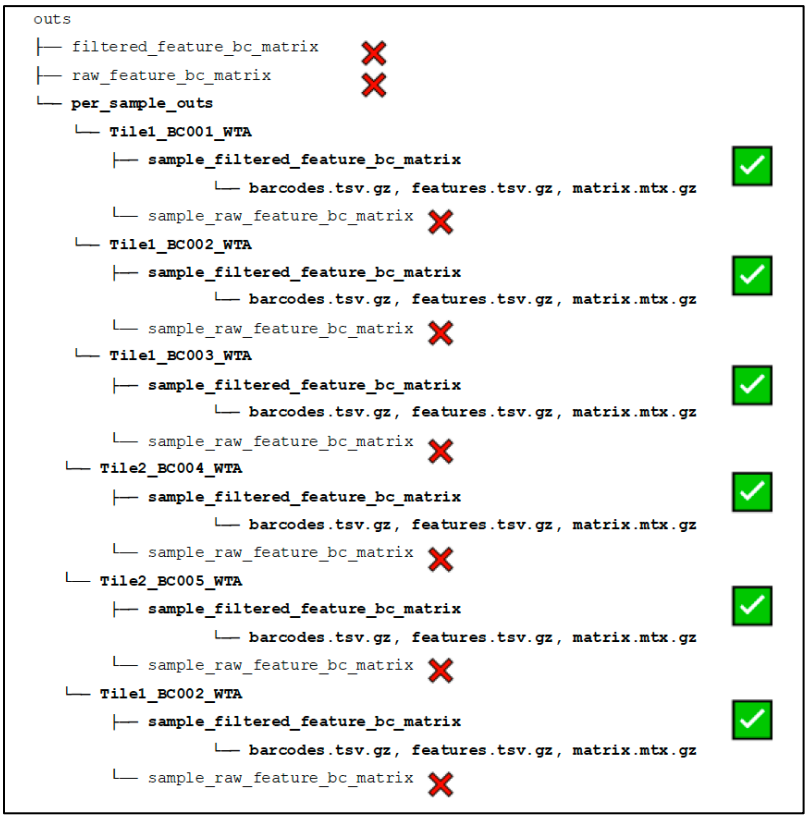


Figure 8. Illustration of where partitioned Cell Ranger outputs are.

# Appendix D. Alternative Trekker Pipeline Input Preparation

## A. Input Preparation: BD Rhapsody Single-Cell TCR/BCR Next + WTA Kit (TrekkerU_RVDJ)

### 1. Overview

If you are processing data generated from the BD Rhapsody Single-Cell TCR/BCR Next + WTA Kit, additional input and a slightly different samplesheet.csv are required by the trekker pipeline.

After you have all the required input, return to Section II.G, "Trigger the Pipeline" and proceed with running the Trekker Primary Analysis Pipeline.

Follow the instructions below to prepare your input.

### 2. Input

Prepare the following inputs before triggering the pipeline.

**NOTE:** Before processing your own data, we recommend using example inputs for TrekkerU_RVDJ to test if the pipeline is properly set up and if your files have the proper format.

1. Bead barcode file

   `${Tile_ID}_BeadBarcodes.txt`

   Download the bead barcode file (e.g., `U0001_001_BeadBarcodes.txt`) using our Tile bead barcode file retrieval tool.

   **NOTE:** Don't forget to unzip the downloaded file before use.

2. Trekker paired FASTQ files

   Example file names:

   ```
   TrekkerU_RVDJ_HumanBreastCancer_R1_001.fastq.gz
   TrekkerU_RVDJ_HumanBreastCancer_R2_001.fastq.gz
   ```

   **NOTE:** Don't forget to check the format conformity of these files (Section II.E, "Check Input Format"). Improper format will lead to pipeline failures.

3. Selected outputs from the bioinformatics pipeline of the single-nuclei sequencing library

   ```
   barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz
   ```

   Follow the steps below to ensure the Trekker pipeline locate these files.

   - For each snRNAseq reaction, generate the three output files using the associated bioinformatics pipeline.
   - Identify the 'quality-filtered' version of these three files and place them in a single folder.
   - Update the column `sc_outdir` in samplesheet.csv such that these three files can be found within the folder defined by the `sc_outdir`.

   **NOTE:** Don't forget to check the format conformity of these files (Section II.E, "Check Input Format"). Improper format will lead to pipeline failures.

4. VDJ-associated output from the bioinformatics pipeline of the single-nuclei sequencing library.

   ```
   TrekkerU_RVDJ_HumanBreastCancer_VDJ_Seurat.rds
   ```

5. Sample sheet

   ```
   samplesheet.csv
   ```

   Use this example `TrekkerU_RVDJ_samplesheet.csv` as a starting point to enter information on your sample to be analyzed.

   **NOTES:**

   – Prepare a separate `samplesheet.csv` for each reaction (i.e., a pair of Trekker FASTQs). The Trekker Primary Analysis Pipeline does NOT support multiple rows (reactions) in a sample sheet.

   – DO NOT swap the column orders in the sample sheet. Re-ordered columns will lead to pipeline failure.

Columns in a sample sheet include:

```
sample,sc_sample,experiment_date,barcode_file,fastq_1,fastq_2,sc_outdir,sc_platform,profile,subsample,cores,scmulti_outdir

TrekkerU_RVDJ_HumanBreastCancer,TrekkerU_RVDJ_HumanBreastCancer_VDJ,20250331,/path/to/U00043_014_BeadBarcodes.txt,/path/to/TrekkerU_RVDJ_HumanBreastCancer_R1_001.fastq.gz,/path/to/TrekkerU_RVDJ_HumanBreastCancer_R2_001.fastq.gz,/path/to/TrekkerU_RVDJ_ExampleInput_HumanBreastCancer/TrekkerU_RVDJ_HumanBreastCancer_scRNAseqOut/TrekkerU_RVDJ_HumanBreastCancer_RSEC_MolsPerCell_MEX/,TrekkerU_RVDJ,singularity,No,8,/path/to/TrekkerU_RVDJ_ExampleInput_HumanBreastCancer/TrekkerU_RVDJ_HumanBreastCancer_scRNAseqOut
```

**NOTES:** See below for further details on columns in a sample sheet.

- `sample`: Prefix for naming the Trekker pipeline output. Avoid using '.' or space.
- `sc_sample`: Prefix for the following VDJ-associated single-nuclei pipeline analysis output. (Use a desired string with no space or special characters.)

  `${sc_sample}_Seurat.rds`
- `experiment_date`: Date of analysis in format YYYYMMDD. This will be used as part of the output folder name.
- `barcode_file`: Absolute path to the Trekker bead barcode file.
- `fastq_1`: Absolute path to the Trekker FASTQ Read 1.
- `fastq_2`: Absolute path to the Trekker FASTQ Read 2.
- `sc_outdir`: Absolute path to the folder containing the three following required single-nuclei pipeline analysis output.

  `barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`
- `sc_platform`: Select a value from the second column of the table below (e.g., TrekkerU_RVDJ), based on the single-cell assay platform you are using from the first column:

  **IMPORTANT:** Make sure to choose the correct value. An incorrect value will lead to poor positioning results.

| Single-cell assay platform | `sc_platform` value |
|---|---|
| BD Rhapsody Single-Cell TCR/BCR Next + WTA, polyT capture | TrekkerU_RVDJ |

- `profile`: How to execute dependencies. Choose among Docker, Singularity, or Conda based on what you set up in Section II.D, "Set Up Dependencies". (Do not use quotes. Case does not matter.)
- `subsample`: If subsampling should be performed for spatial positioning parameterization. Use "no" unless you have >5 x $10^6$ nuclei to be positioned. (Do not use quotes. Case does not matter.)

- `cores`: Number of cores to use for the spatial positioning step. We recommend 8, which is sufficient for most cases.
- `scmulti_outdir`: Absolute path to the folder containing the following VDJ-associated single-nuclei pipeline analysis output.

  `${sc_sample}_Seurat.rds`

### 3. Testing and example datasets

To test pipeline configuration, use our example inputs to trigger the pipeline and compare the results to our example output.

**NOTE:** For best results, use Adobe Acrobat Reader to view this user manual. If you are copying and pasting these commands, ensure that the commands include the correct punctuation (e.g., hyphens) before execution.

1. Download example inputs

   Single-nuclei assay platform: TrekkerU_RVDJ

   ```
   wget https://www.takarabio.com/resourcedocument/x353761 -O - |
   tar -xzf -
   ```

2. Download example outputs

   Single-nuclei assay platform: TrekkerU_RVDJ

   ```
   wget https://www.takarabio.com/resourcedocument/x357444 -O - |
   tar -xzf -
   ```

For troubleshooting instructions, refer to Appendix E, Section D, "Troubleshooting: Input Preparation for BD Rhapsody Single-Cell TCR/BCR Next".

## B. Input Preparation: BD Rhapsody Single-Cell ATAC-Seq + mRNA WTA Kit (TrekkerU_RATAC)

### 1. Overview

If you are processing data generated from the BD Rhapsody Single-Cell ATAC-Seq + mRNA WTA Kit (TrekkerU_RATAC), additional input and a slightly different `samplesheet.csv` are required by the Trekker Primary Analysis Pipeline.

After you have all the required input, return to Section II.G, "Trigger the Pipeline" and proceed with running the Trekker Primary Analysis Pipeline.

Follow the instructions below to prepare your input.

### 2. Input

Prepare the following inputs before triggering the pipeline.

**NOTE:** Before processing your own data, we recommend using example inputs for TrekkerU_RATAC to test if the pipeline is properly set up and if your files have the proper format.

1. Bead barcode file

`${Tile_ID}_BeadBarcodes.txt`

Download the bead barcode file (e.g., `U0001_001_BeadBarcodes.txt`) using our Tile bead barcode file retrieval tool.

**NOTE:** Don't forget to unzip the downloaded file before use.

2. Trekker paired FASTQ files

   Example file names:

   `TrekkerU_RATAC_MouseKidney_R1_001.fastq.gz`

   `TrekkerU_RATAC_MouseKidney_R2_001.fastq.gz`

   **NOTE:** Don't forget to check the format conformity of these files (Section II.E, "Check Input Format"). Improper format will lead to pipeline failures.

3. Selected outputs from the bioinformatics pipeline of the single-nuclei sequencing library

   `barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`

   Follow the steps below to ensure the Trekker pipeline locate these files.

   - For each snRNAseq reaction, generate the three output files using the associated bioinformatics pipeline.
   - Identify the 'quality-filtered' version of these three files and place them in a single folder.
   - Update the column `sc_outdir` in samplesheet.csv such that these three files can be found within the folder defined by the `sc_outdir`.

   **NOTE:** Don't forget to check the format conformity of these files (Section II.E, "Check Input Format"). Improper format will lead to pipeline failures.

4. ATAC-associated output from the bioinformatics pipeline of the single-nuclei sequencing library.

   `atac-barcodes.tsv.gz, atac-features.tsv.gz, atac-matrix.mtx.gz`

5. Sample sheet

   `samplesheet.csv`

   Use this example `TrekkerU_RATAC_samplesheet.csv` as a starting point to enter information on your sample to be analyzed.

**NOTES:**

- Prepare a separate `samplesheet.csv` for each reaction (i.e., a pair of Trekker FASTQs). The Trekker Primary Analysis Pipeline does NOT support multiple rows (reactions) in a sample sheet.
- DO NOT swap the column orders in the sample sheet. Re-ordered columns will lead to pipeline failure.

Columns in a sample sheet include:

```
sample,sc_sample,experiment_date,barcode_file,fastq_1,fastq_2,sc_outdir,sc_platform,profi
le,subsample,cores,scmulti_outdir
TrekkerU_RATAC_MouseKidney,TrekkerU_RATAC_MouseKidney_scRNAseq,20250624,/path/to/U0027_01
6_BeadBarcodes.txt,/path/to/TrekkerU_RATAC_MouseKidney_R1_001.fastq.gz,/path/to/TrekkerU_
RATAC_MouseKidney_R2_001.fastq.gz,/path/to/TrekkerU_RATAC_MouseKidney_scseqOut/TrekkerU_R
ATAC_MouseKidney_RSEC_MolsPerCell_MEX/,TrekkerU_RATAC,singularity,no,8,/path/to/TrekkerU_
RATAC_MouseKidney_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/
```

**NOTES:** See below for further details on columns in a sample sheet.

- `sample`: Prefix for naming the Trekker pipeline output. Avoid using '.' or space.
- `sc_sample`: Prefix for single-nuclei pipeline analysis output. (For book-keeping purposes only. Use a desired string with no space or special characters.)
- `experiment_date`: Date of analysis in format YYYYMMDD. This will be used as part of the output folder name.
- `barcode_file`: Absolute path to the Trekker bead barcode file.
- `fastq_1`: Absolute path to the Trekker FASTQ Read 1.
- `fastq_2`: Absolute path to the Trekker FASTQ Read 2.
- `sc_outdir`: Absolute path to the folder containing the three following required single-nuclei pipeline analysis output.

  `barcodes.tsv.gz, features.tsv.gz, matrix.mtx.gz`
- `sc_platform`: Select a value from the second column of the table below (e.g., TrekkerU_RVDJ), based on the single-cell assay platform you are using from the first column:

**IMPORTANT:** Make sure to choose the correct value. An incorrect value will lead to poor positioning results.

| Single-cell assay platform | `sc_platform` value |
|---|---|
| BD RhapsodyTM Single-Cell ATAC-Seq + mRNA Whole Transcriptome Analysis Kit | TrekkerU_RATAC |

- `profile`: How to execute dependencies. Choose among Docker, Singularity, or Conda based on what you set up in Section II.D, "Set Up Dependencies". (Do not use quotes. Case does not matter.)
- `subsample`: If subsampling should be performed for spatial positioning parameterization. Use "no" unless you have >5 x $10^6$ nuclei to be positioned. (Do not use quotes. Case does not matter.)
- `cores`: Number of cores to use for the spatial positioning step. We recommend 8, which is sufficient for most cases.
- `scmulti_outdir`: Absolute path to the folder containing the following ATAC-associated single-nuclei pipeline analysis output.
  `atac-barcodes.tsv.gz, atac-features.tsv.gz, atac-matrix.mtx.gz`

### 3. Testing and example datasets

To test pipeline configuration, use our example inputs to trigger the pipeline and compare the results to our example output.

**NOTE:** For best results, use Adobe Acrobat Reader to view this user manual. If you are copying and pasting these commands, ensure that the commands include the correct punctuation (e.g., hyphens) before execution.

1. Download example inputs

   Single-nuclei assay platform: TrekkerU_RATAC

   ```
   wget https://www.takarabio.com/resourcedocument/x353760 -O - |
   tar -xzf -
   ```

2. Download example outputs

   Single-nuclei assay platform: TrekkerU_RATAC

   ```
   wget https://www.takarabio.com/resourcedocument/x357443 -O - |
   tar -xzf -
   ```

For troubleshooting instructions, refer to Appendix E, Section E, "Troubleshooting: Input Preparation for BD Rhapsody Single-Cell ATAC-Seq".

# Appendix E. Troubleshooting

## A. Troubleshooting: Local Installation

- **A required input is missing.**

  You will see an error message like the one below. If this happens, check your `samplesheet.csv` to ensure the path for the missing input is correct.

```
Checking samplesheet...
 '/home/TrekkerU_C_ExampleInput_MouseBrain/samplesheet.csv' exists.

Checking Trekker FASTQ R1...
'/home/TrekkerU_C_ExampleInput_MouseBrain/TrekkerU_C_Mouse_brain_R1_001
.fastq.gz' exists.

Checking Trekker FASTQ R2...
'/home/TrekkerU_C_ExampleInput_MouseBrain/TrekkerU_C_Mouse_brain_R2_001
.fastq.gz' exists.

Checking Trekker tile spatial barcode whitelist...
'/home/TrekkerU_C_ExampleInput_MouseBrain/LTTag0053_003_BeadBarcodes.tx
t' exists.

Checking snRNAseq outputs...
Abort the Pipeline. Error: Unable to locate single-nuclei RNAseq
pipeline output folder. Please ensure the path specified by 'sc_outdir'
in your samplesheet exists.
```

- **A step failed.**

  The error message will direct you to the log of the failed step. If this happens, review the step-log for details or share it with technical_support@takarabio.com.

```
...
Checking snRNAseq outputs...
snRNAseq outputs located.
Starting Trekker Analysis (TrekkerU_C) using singularity
Start fastq_parser
Abort the Pipeline. Error: fastq_parser failed, Please refer to
'/home/TrekkerU_C_ExampleInput_MouseBrain/log/TrekkerU_C_MouseBrain/fastq_
parser.log' for details.
```

- (TrekkerU_RVDJ only) **The required VDJ-associated input is missing.**

  You will see error messages like below. If this happens, check your `samplesheet.csv` to ensure `sc_sample` and `scmulti_outdir` values are correct such that the path to the VDJ-associated file follows this pattern:

  `${scmulti_outdir}/${sc_sample}_Seurat.rds`

  - Error due to missing `scmulti_outdir` column in the `samplesheet.csv`

```
...
Checking scmulti_outdir...
Abort the Pipeline. Error: The scmulti_outdir value in the samplesheet
is empty. Please check your samplesheet and provide a valid value.
```

- Error due to incorrect `scmulti_outdir` value in the `samplesheet.csv`

```
...
Checking scmulti_outdir...
Error message:
Abort the Pipeline. Error: Unable to locate folder
'/home/TrekkerU_RVDJ_ExampleInput_HumanBreastCancer/TrekkerU_RVDJ_Human
BreastCancer_scRNAseqOut/'.
Please ensure the path specified by 'scmulti_outdir' in your
samplesheet exists.
```

- Error due to incorrect `sc_sample` value in the `samplesheet.csv`

```
... Checking scmulti_outdir...
'/home/TrekkerU_RVDJ_ExampleInput_HumanBreastCancer/TrekkerU_RVDJ_Hum
anBreastCancer_scRNAseqOut/' exists.
Checking scmulti_outdir/vdj_seurat file...
Error message:
Abort the Pipeline. Error: File
'/home/TrekkerU_RVDJ_ExampleInput_HumanBreastCancer/TrekkerU_RVDJ_Hum
anBreastCancer_scRNAseqOut/TrekkerU_RVDJ_HumanBreastCancer_VDJ_Seurat
.rds' does not exist.
Please check your samplesheet and provide a valid value to
'sc_sample' (prefix for vdj seurat file) or ensure that file
'TrekkerU_RVDJ_HumanBreastCancer_VDJ_Seurat.rds' exists in the path
specified by 'scmulti_outdir' in your samplesheet.
```

- (TrekkerU_RATAC only) **The required ATAC-associated input is missing.**

  You will see error messages like below. If this happens, check your `samplesheet.csv` to ensure the `scmulti_outdir` value is correct and it contains these three ATAC-associated files:

  `${scmulti_outdir}/atac-barcodes.tsv.gz`

  `${scmulti_outdir}/atac-features.tsv.gz`

  `${scmulti_outdir}/atac-matrix.mtx.gz`

- Error due to missing `scmulti_outdir` column in the `samplesheet.csv`

```
...
Checking scmulti_outdir...
Abort the Pipeline. Error: The scmulti_outdir value in the
samplesheet is empty. Please check your samplesheet and provide a
valid value.
```

- Error due to incorrect `scmulti_outdir` value in the `samplesheet.csv`

```
...
Checking scmulti_outdir...
Abort the Pipeline. Error: Unable to locate the multiomic output folder
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseKidn
ey_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/' from the
single-nuclei pipeline. Please ensure the path specified by
'scmulti_outdir' in your samplesheet exists.
```

- Error due to missing files in folder defined by `scmulti_outdir`

```
...
Checking scmulti_outdir...
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseK
idney_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/'
exists.
Checking scmulti_outdir/atac files...
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseK
idney_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/atac
-barcodes.tsv.gz' exists.
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseK
idney_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/atac
-features.tsv.gz' exists.
Abort the Pipeline. Error: File
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseK
idney_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/atac
-matrix.mtx.gz' does not exist. Please check your samplesheet and
provide a valid path to the 'scmulti_outdir' value or ensure that
the 'atac-matrix' can be located in the path specified by
'scmulti_outdir' of your samplesheet.
```

- (TrekkerU_RATAC only) **The numbers of nuclei in RNA-Seq and ATAC-Seq matrices do not match.**

  You will see error messages like below. If this happens, check your `samplesheet.csv` to ensure `sc_outdir` and `scmulti_outdir` point to the RNA-seq and ATAC-seq matrices that came from the same run of the single-cell bioinformatics pipeline.

```
...
Checking the number of nuclei in snRNAseq and snATACseq count matrices
from the single-nuclei pipeline...
Abort the Pipeline. Error: Different number of nuclei found. Please
ensure snRNAseq and snATACseq count matrices have the same number of
nuclei.
```

- (TrekkerFX_FLEX only) **Trekker FX input has not been partitioned based on multiplexing barcodes.**

  You will see errors as below.

  If this occurs, it indicates that the single-cell count matrix used has not been partitioned. Please refer to '[Trekker FX Data Analysis Quick Start Guide]' in the trekker-v1.4.11 folder for instructions on splitting GEX data into partitions based on multiplexing barcodes.

  You will see error message in your stdout like below:

```
...
Checking snRNAseq outputs...
snRNAseq outputs located.
Starting Trekker Analysis (TrekkerFX_FLEX) using conda
Start check_mtx_status
Abort the Pipeline. Error: check_mtx_status failed. Please refer to
'/home/samplesheets/log/TrekkerFX_FLEX_Example/check_mtx_status.log' for
details.
```

  `check_mtx_status.log` will have error messages like below:

```
...
ERROR: Multiple multiplexed partitions detected in
/home/TrekkerFX_FLEX_ExampleInput_Human_Tonsil/outs/per_sample_outs/Examp
le/count/sample_filtered_feature_bc_matrix/barcodes.tsv.gz.
barcodes.tsv.gz contains barcodes from more than one partition
with counts: {'BC011': 38229, 'BC010': 33248, 'BC009': 31084, 'BC008':
29791}
Please refer to 'Trekker FX Primary Pipeline Analysis Quick Start Guide'
in folder trekker-v1.4.11
to prepare single-cell count matrix in partitioned format.
```

## B. Troubleshooting: Merging Pipeline Output

- **The path to the script folder is not defined.**

  You will see an error message like the one below. If this happens, define path following instructions in Appendix B, Section B, "Set Up Paths".

  STDOUT:

```
...
Abort the trekker_merger. Error: mergeSeurats failed, Please refer to
/trekker_merger_example_input/log/MouseEmbryo/mergeSeurats.log for
details.
```

  mergeSeurats.log:

```
...
Fatal error: cannot open file '/home/trekker_merger-
v1.4.11//mergeSeurats.R': No such file or directoryERROR
conda.cli.main_run:execute(125): `...` failed. (See above for error)
```

- **A required input is missing**.

  You will see an error message like the one below. If this happens, check your `samplesheet.csv` to ensure the path for the missing input is correct.

```
...
Seurat file not found for TrekkerR_MouseEmbryo_LaneD. Please check your
samplesheet and provide a valid path to the output folder for sample
TrekkerR_MouseEmbryo_LaneD.
Metrics file not found for TrekkerR_MouseEmbryo_LaneD. Please check
your samplesheet and provide a valid path to the output folder for
sample TrekkerR_MouseEmbryo_LaneD.
```

- **Intermediate folder `input` exists**.

  If this happens, remove the `input` folder and re-trigger the run.

```
...
/home/TrekkerR_ExampleInput_MouseEmbryo/input/ exists. Please delete the
input folder or update its name and start the trekker_merger again.
```

- **Outputs being merged are not from the same Trekker tile or single-nuclei platform**.

  If this happens, modify your `samplesheet.csv` so only outputs from the same tile and single-nuclei platform are included.

```
...
Error: Trekker Tile ID must be the same for all samples. Please check
your samplesheet and provide the output folders for samples from the
same Trekker tile.
Error: The Single_cell_assay must be the same for all samples. Please
check your samplesheet and provide the output folders for samples
processed by the same single cell assay platform.
```

- **A step failed**.

  The error message will direct you to the log of the failed step. If this happens, review the step-log for details or share it with technical_support@takarabio.com.

```
...
Running the trekker_merger using 'conda'
Start mergeSeurats
Abort the trekker_merger. Error: mergeSeurats failed. Please refer to
/home/TrekkerR_ExampleInput_MouseEmbryo/log/TrekkerR_MouseEmbryo/mergeSe
urats.log for details
```

## C. Troubleshooting: Preprocessing for Illumina Single Cell 3′ RNA Prep Kit (TrekkerU_PIP)

- **A required input is missing.**

  You will see an error message like the one below. If this happens, check your `convertmeta.csv` to ensure the path for the missing input is correct.

```
Checking samplesheet...

'/home/trekkerU_PIP_converter_example_input/convertmeta.csv' exists

Checking samplesheet format...

samplesheet is in Linux format.

Checking FASTQ R1...

Abort the trekkerU_PIP_converter. Error: File
'/home/trekkerU_PIP_converter_example_input/TrekkerU_PIP_MouseBrain_R1_0
01.fastq.gz' does not exist. Please provide a valid path to the
'fastq_R1' in
'/home/trekkerU_PIP_converter_example_input/convertmeta.csv'.
```

- **Incorrect chemistry argument used in the sample sheet.**

  If this happens, modify your `convertmeta.csv` to ensure that the value for `chem` is one of the following PIPseeker chemistry: v3, v4, or V.

```
...

Checking snRNAseq output files in sc_outdir...

snRNAseq output files located.

Checking PIPseq chemistry...

Chemistry used: v5

Abort the trekkerU_PIP_converter. Error: The PIPseq chemistry must be
v3, v4 or V. Please provide a valid value to the 'chem' in
'/home/trekkerU_PIP_converter_example_input/convertmeta.csv'
```

- **Validation failed for updated FASTQ R1**

  FASTQ R1 retained by pipseeker and updated FASTQ R1 generated does not have same number of reads. If this happens, please delete all the intermediate files and retrigger a clean trekkerU_PIP_converter run.

```
...

Starting the trekkerU_PIP_converter...

Start convertBarcode

convertBarcode Done

Start concatR1

concatR1 Done

Validating FASTQ R1...

Abort the trekkerU_PIP_converter. Error: Total reads mismatch after
concatenation. Found 900000 reads in
'/home/trekkerU_PIP_converter_example_input/barcoded_fastqs/TrekkerU_PIP
_MouseBrain_converted_R1.fastq.gz', but expected 979647 reads based on
PIPseeker output file
''/home/trekkerU_PIP_converter_example_input/metrics/barcode_stats.csv'.
Please delete all the intermediate files and retrigger
trekkerU_PIP_converter.
```

- **A step failed.**

  The error message will direct you to the log of the failed step. If this happens, review the step-log for details or share it with technical_support@takarabio.com.

```
...

Starting the trekkerU_PIP_converter...

Start convertBarcode

Abort the TrekkerU_PIP_converter. Error: convertBarcode failed. Please
refer to
'/home/trekkerU_PIP_converter_example_input/log/TrekkerU_PIP_MouseBrain/c
onvertBarcode.log' for details.
```

## D. Troubleshooting: Input Preparation for BD Rhapsody Single-Cell TCR/BCR Next + WTA Kit (TrekkerU_RVDJ)

- **A required input is missing.**

  You will see an error message like the one below. If this happens, check your `samplesheet.csv` to ensure the path for the missing input is correct.

```
Checking samplesheet...
 '/home/TrekkerU_C_ExampleInput_MouseBrain/samplesheet.csv' exists.
Checking Trekker FASTQ R1...

'/home/TrekkerU_C_ExampleInput_MouseBrain/TrekkerU_C_Mouse_brain_R1_001
.fastq.gz' exists.
Checking Trekker FASTQ R2...

'/home/TrekkerU_C_ExampleInput_MouseBrain/TrekkerU_C_Mouse_brain_R2_001
.fastq.gz' exists.
Checking Trekker tile spatial barcode whitelist...

'/home/TrekkerU_C_ExampleInput_MouseBrain/LTTag0053_003_BeadBarcodes.tx
t' exists.
Checking snRNAseq outputs...
Abort the Pipeline. Error: Unable to locate single-nuclei RNAseq
pipeline output folder. Please ensure the path specified by 'sc_outdir'
in your samplesheet exists.
```

- **A step failed.**

  The error message will direct you to the log of the failed step. If this happens, review the step-log for details or share it with technical_support@takarabio.com.

```
...
Checking snRNAseq outputs...
snRNAseq outputs located.
Starting Trekker Analysis (TrekkerU_C) using singularity
Start fastq_parser
Abort the Pipeline. Error: fastq_parser failed, Please refer to
'/home/TrekkerU_C_ExampleInput_MouseBrain/log/TrekkerU_C_MouseBrain/fast
q_parser.log' for details.
```

- **The required VDJ-associated input is missing.**

  You will see error messages like below. If this happens, check your `samplesheet.csv` to ensure `sc_sample` and `scmulti_outdir` values are correct such that the path to the VDJ-associated file follows this pattern:

  `${scmulti_outdir}/${sc_sample}_Seurat.rds`

  - Due to missing `scmulti_outdir` column in the `samplesheet.csv`

```
...

Checking scmulti_outdir...

Abort the Pipeline. Error: The scmulti_outdir value in the
samplesheet is empty. Please check your samplesheet and provide a
valid value.
```

  - Due to incorrect `scmulti_outdir` value in the `samplesheet.csv`

```
...

Checking scmulti_outdir...

Abort the Pipeline. Error: Unable to locate the multiomic output
folder
'/home/TrekkerU_RVDJ_ExampleInput_HumanBreastCancer/TrekkerU_RVDJ_Hum
anBreastCancer_scRNAseqOut/' from the single-nuclei pipeline. Please
ensure the path specified by 'scmulti_outdir' in your samplesheet
exists.
```

  - Due to incorrect `sc_sample` value in the `samplesheet.csv`

```
...

Checking scmulti_outdir...

'/home/TrekkerU_RVDJ_ExampleInput_HumanBreastCancer/TrekkerU_RVDJ_Hum
anBreastCancer_scRNAseqOut/' exists.

Checking scmulti_outdir/vdj_seurat file...

Abort the Pipeline. Error: File

'/home/TrekkerU_RVDJ_ExampleInput_HumanBreastCancer/TrekkerU_RVDJ_Hum
anBreastCancer_scRNAseqOut/TrekkerU_RVDJ_HumanBreastCancer_VDJ_Seurat
.rds' does not exist.

Please check your samplesheet and provide a valid value to
'sc_sample' (prefix for vdj seurat file) or ensure that file
'TrekkerU_RVDJ_HumanBreastCancer_VDJ_Seurat.rds' exists in the path
specified by 'scmulti_outdir' in your samplesheet.
```

## E. Troubleshooting: Input Preparation for BD Rhapsody Single-Cell ATAC-Seq + mRNA Whole Transcriptome Analysis Kit (TrekkerU_RATAC)

- **A required input is missing.**

  You will see an error message like the one below. If this happens, check your `samplesheet.csv` to ensure the path for the missing input is correct.

```
Checking samplesheet...
 '/home/TrekkerU_C_ExampleInput_MouseBrain/samplesheet.csv' exists.
Checking Trekker FASTQ R1...

'/home/TrekkerU_C_ExampleInput_MouseBrain/TrekkerU_C_Mouse_brain_R1_001.
fastq.gz' exists.
Checking Trekker FASTQ R2...

'/home/TrekkerU_C_ExampleInput_MouseBrain/TrekkerU_C_Mouse_brain_R2_001.
fastq.gz' exists.
Checking Trekker tile spatial barcode whitelist...

'/home/TrekkerU_C_ExampleInput_MouseBrain/LTTag0053_003_BeadBarcodes.txt
' exists.
Checking snRNAseq outputs...
Abort the Pipeline. Error: Unable to locate single-nuclei RNAseq
pipeline output folder. Please ensure the path specified by 'sc_outdir'
in your samplesheet exists.
```

- **A step failed.**

  The error message will direct you to the log of the failed step. If this happens, review the step-log for details or share it with technical_support@takarabio.com.

```
...
Checking snRNAseq outputs...
snRNAseq outputs located.
Starting Trekker Analysis (TrekkerU_C) using singularity
Start fastq_parser
Abort the Pipeline. Error: fastq_parser failed, Please refer to
'/home/TrekkerU_C_ExampleInput_MouseBrain/log/TrekkerU_C_MouseBrain/fast
q_parser.log' for details.
```

- **The required ATAC-associated input is missing.**

  You will see error messages like below. If this happens, check your samplesheet.csv to ensure 'scmulti_outdir' value is correct and it contains these three ATAC-associated files.

  `${scmulti_outdir}/atac-barcodes.tsv.gz`

  `${scmulti_outdir}/atac-features.tsv.gz`

  `${scmulti_outdir}/atac-matrix.mtx.gz`

  - Due to missing `scmulti_outdir` column in the `samplesheet.csv`

```
...
Checking scmulti_outdir...
Abort the Pipeline. Error: The scmulti_outdir value in the samplesheet is
empty. Please check your samplesheet and provide a valid value.
```

  - Due to incorrect `scmulti_outdir` value in the `samplesheet.csv`

```
...
Checking scmulti_outdir...
Abort the Pipeline. Error: Unable to locate the multiomic output folder
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseKidney
_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/' from the
single-nuclei pipeline. Please ensure the path specified by
'scmulti_outdir' in your samplesheet exists.
```

  - Due to missing files in folder defined by `scmulti_outdir`

```
...
Checking scmulti_outdir...
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseKidney
_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/' exists.
Checking scmulti_outdir/atac files...
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseKidney
_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/atac-
barcodes.tsv.gz' exists.
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseKidney
_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/atac-
features.tsv.gz' exists.
Abort the Pipeline. Error: File
'/home/TrekkerU_RATAC_ExampleInput_MouseKidney/TrekkerU_RATAC_MouseKidney
_scseqOut/TrekkerU_RATAC_MouseKidney_ATAC_Cell_by_Peak_MEX/atac-
matrix.mtx.gz' does not exist. Please check your samplesheet and provide
a valid path to the 'scmulti_outdir' value or ensure that the 'atac-
matrix' can be located in the path specified by 'scmulti_outdir' of your
samplesheet.
```

- **The numbers of nuclei in RNA-Seq and ATAC-Seq matrices do not match.**

  You will see error messages like below. If this happens, check your `samplesheet.csv` to ensure `sc_outdir` and `scmulti_outdir` point to the RNA-Seq and ATAC-Seq matrices that came from the same run of the single-cell bioinformatics pipeline.

```
...
Checking the number of nuclei in snRNAseq and snATACseq count matrices
from the single-nuclei pipeline...
Abort the Pipeline. Error: Different number of nuclei found. Please
ensure snRNAseq and snATACseq count matrices have the same number of
nuclei.
```

| Contact Us | |
|---|---|
| **Customer Service/Ordering** | **Technical Support** |
| tel: 800.662.2566 (toll-free) | tel: 800.662.2566 (toll-free) |
| fax: 800.424.1350 (toll-free) | fax: 800.424.1350 (toll-free) |
| web: takarabio.com/service | web: takarabio.com/support |
| e-mail: ordersUS@takarabio.com | e-mail: technical_support@takarabio.com |

## Notice to Purchaser

Our products are to be used for **Research Use Only**. They may not be used for any other purpose, including, but not limited to, use in humans, therapeutic or diagnostic use, or commercial use of any kind. Our products may not be transferred to third parties, resold, modified for resale, or used to manufacture commercial products or to provide a service to third parties without our prior written approval.

Your use of this product is also subject to compliance with any applicable licensing requirements described on the product's web page at takarabio.com. It is your responsibility to review, understand and adhere to any restrictions imposed by such statements.



This document has been reviewed and approved by the Quality Department.