ScaleBio Seq Suite: RNA

Data Analysis Handbook

For Research Use Only.

Legal Notices

Document 1020803, Rev E, Jan 2025

5601 Oberlin Dr., Suite 110
San Diego, CA 92121
https://scale.bio/
support@scale.bio

Disclaimer of Warranty:
TO THE EXTENT PERMITTED BY APPLICABLE LAW, SCALEBIO PROVIDES THIS DOCUMENT "AS IS" WITHOUT WARRANTY OF ANY KIND, INCLUDING WITHOUT LIMITATION, ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NONINFRINGEMENT. IN NO EVENT WILL SCALEBIO BE LIABLE TO YOU OR ANY THIRD PARTY FOR ANY LOSS OR DAMAGE, DIRECT OR INDIRECT, FROM THE USE OF THIS DOCUMENT, INCLUDING WITHOUT LIMITATION, LOST PROFITS, LOST INVESTMENT, BUSINESS INTERRUPTION, GOODWILL, OR LOST DATA, EVEN IF SCALEBIO IS EXPRESSLY ADVISED IN ADVANCE OF THE POSSIBILITY OF SUCH LOSS OR DAMAGE. Any warranties applicable to the ScaleBio products are set forth in the Terms and Conditions accompanying such product and such Terms and Conditions are not modified in any way by the terms of this notice.

Trademark Information:
ScaleBio may make reference to products or services provided by other companies using their brand names or company names solely for the purpose of clarity, and does not assert any ownership rights over those third-party marks or names. Images were created with BioRender.com

Patent Information:
ScaleBio products may be covered by one or more patents as indicated at: https://scale.bio/legal-notice/

Terms and Conditions:
The use of the ScaleBio products described herein is subject to ScaleBio's Terms and Conditions that accompany the product, or such other terms as have been agreed to in writing between ScaleBio and the user.

Intended Use:
All products and services described herein are intended FOR RESEARCH USE ONLY and NOT FOR USE IN DIAGNOSTIC PROCEDURES.

Table of Contents

Introduction

The ScaleBio™ Single Cell RNA Sequencing Kit uses a 3-level combinatorial indexing strategy to resolve gene expression data at single cell resolution by using a combination of unique RT Barcodes, Ligation Barcodes and PCR Barcodes. The ScaleBio Seq Suite: RNA Data Analysis Pipeline is designed as an end-to-end workflow that takes users from raw sequencing output of their ScaleBio Single Cell RNA Sequencing Kit library to a thorough assessment of that library's performance, cell calling, and ultimately gene expression matrices.

This handbook serves as a high-level guide for setting up, running, and understanding the outputs of the ScaleBio Seq Suite: RNA Data Analysis Pipeline. For specific step-by-step instructions on installing and running the pipeline please refer to our GitHub repository. The introductory readme markdown file (README.md) provides an overview of the workflow. Additionally, there is a series of markdown files (*.md) within ScaleRna/docs to help guide users in more detail at each major step.

Compatibility

Please confirm that you are using the correct version of the Seq Suite: RNA Data Analysis Pipeline compatible with the chemistry version of your Single Cell RNA Sequencing Kit (Table 1) Kit.

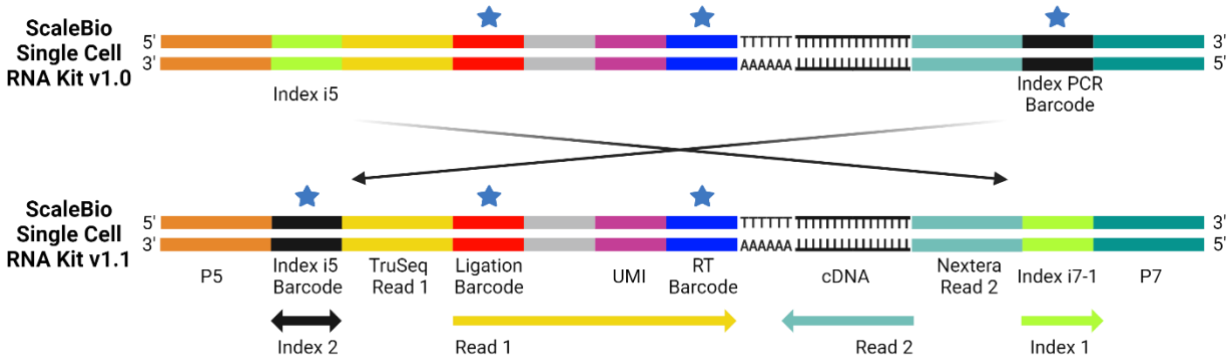
Table 1: ScaleBio Pipeline and Assay Chemistry Compatibility

Single Cell RNA Sequencing Kit Version	Seq Suite: RNA Data Analysis Pipeline Version	Note
v1.1	v1.4 or later	• i5 or i7 cell indexing • Multi-plate merging support for Extended Throughput
v1.0	v1.3.3 or later	• Only i7 cell indexing support • No Extended Throughput support

This compatibility requirement is due to different cell indexing configurations between the two kit versions. Cells are still indexed via unique RT Barcodes and Ligation Barcodes, but the position of the third barcode used for combinatorial indexing is different as illustrated Figure 1:

- **Single Cell RNA Sequencing Kit v1.0**: the third cell indexing barcode is Index PCR Barcode at the position of Index 1.
- **Single Cell RNA Sequencing Kit v1.1**: the third cell indexing barcode is Index i5 Barcode at the position of Index 2.

Figure 1: RNA Library Design Comparison Highlighting the Barcodes Used for Combinatorial Indexing (Blue Star)



In addition, changing the cell indexing configuration enables increasing the number of recovered cells from the assay by the usage of multiple Final Distribution Plates from the ScaleBio Single Cell RNA Extended Throughput v1.1 Kit. Version 1.4 of the Seq Suite: RNA Data Analysis Pipeline supports merging of multiple Final Distribution Plates.

Please review the changelog on GitHub, and *Appendix B: Single Cell RNA Extended Throughput Kit v1.1* on how to run multiple Final Distribution Plates through the pipeline.

Quick Start Guide

System Requirements:

- Linux system with GLIBC >= 2.17 (such as CentOS 7 or later)
- Java 11 or later
- 64GB of RAM and 12 CPU cores
    - Smaller datasets can be run with 32GB of RAM and 6 CPUs

Instructions:

1. Install Nextflow (23.10 or later).
2. Download the latest ScaleRna workflow to your machine.
3. Determine your system's optimal method to install dependencies.
4. Launch the ScaleBio down sampled pipeline test run.
5. Download one of the ScaleBio pre-built genomes or generate a new custom genome reference
6. Create a sample barcode table (samples.csv) for ScaleBio sample demultiplexing.
7. Specify all analysis parameters in the runParams.yml.
    7.1. To specify the fastq or bcl runFolder directory:
        7.1.1. *fastqDir : path/to/fastqDir*
        7.1.2. *runFolder : path/to/bcl/runFolder*
    7.2. To specify the library structure (v1.0 or v1.1)
        7.2.1. *libStructure : libV1.1json*
8. Setup your final nextflow command and launch the workflow!

Chapter 1: Pipeline Setup, Installation and Testing

1.1. Requirements

The workflow can be launched either on a macOS or on a Linux system, such as CentOS 7 or later (GLIBC ≥2.17) with Java 11 (or later). For running the pipeline on either system, the recommended minimal configuration is 64 GB of RAM and 12 CPU cores. For smaller datasets, 32 GB of RAM and 6 CPU cores can be sufficient. In addition, the workflow requires temporary storage for intermediate files, ranging up to 3 TB for large (e.g. NovaSeq S4) sequencing runs.

1.2. Install Nextflow

To install the pipeline on a new system, first install Nextflow, following the instructions at https://www.nextflow.io/. The installation requires Java version 11 or higher. Once downloaded to your system, you can confirm that the Nextflow executable works as expected by running the following command:

```
nextflow run hello
```

Note that Nextflow can be installed in your user directory without admin rights on the system. Once you have the Nextflow command running, it is recommended to add the install location to the path variable in the users bash startup file (i.e., .bash_profile, .bashrc, .profile).

1.3. Install the ScaleBio Seq Suite: RNA Pipeline

The pipeline can be downloaded by two methods:

1. By going to the ScaleBio Seq Suite: RNA GitHub page, clicking the green “Code” button and then “Download ZIP”. Unpack this file on your system directly in the directory in which you want to install the pipeline. To make sure the download is complete, make sure the executable commands (PY files) in the ScaleRna/bin directory have the appropriate read/write/execute privileges on your server.
2. The GitHub repository can be cloned to your machine:

   ```
   git clone https://github.com/ScaleBio/ScaleRna.git
   ```

1.4. Determine a dependency install method

The workflow requires several dependencies (see *Appendix C: Software Dependencies*). With the `-profile` setting, users can select the method of dependency installment. For Nextflow there are multiple methods for installing dependencies. This allows users flexibility depending on their systems requirements or limitations. These dependency profiles can be selected in one of three options:

1. Docker (RECOMMENDED): Using containers (‘docker’).
2. Singularity: Alternative container option, usually available as a module on most high-performance computers.
3. Conda: Using the conda package manager so NexfFlow can generate conda environments.

If none of the above options are successful, there is a final option to manually download and install all necessary dependencies, as well as placing them on path. This optino is prone to error and will usually require a lot more setup and installation time than the above option.

1.4.1. Using Containers

Containers have all the necessary code, libraries, and configurations for the workflow to operate on most systems. If your system supports execution of a container (either using *Docker* or *Singularity*) this is likely the easiest way to handle dependencies, especially if you are familiar with running containerized workflows.

1.4.2. Docker vs. Singularity Container

Docker support is enabled with the Nextflow command-line option `-profile docker`. Note that setting up *Docker* support for the first time on a new system typically requires admin (root) access and some familiarity with the system.

If your system access does not support the use of *Docker*, then *Singularity* is an alternative for container execution that is available on many HPC clusters. We require Singularity 2.3 or newer. Setting `-profile docker,singularity` (no space) will use the Singularity engine for all dependencies.

Container usage can require some extra setup to ensure all relevant file paths are accessible from inside the containers:

- For *Docker*, Nextflow will set the relevant options automatically at runtime to mount (`bind`) input and output paths to the container.
- For *Singularity*, this requires `USER BIND CONTROL` to be enabled in the system-wide configuration (see The Singularity Config File and the notes in the Nextflow singularity documentation).
- The environment variable `NXF_SINGULARITY_CACHEDIR` can be used to control where *Singularity* images are stored. This should be a writable location that is available on all compute nodes.
- Similarly, `TMPDIR` should be changed from the default `/tmp` to a location writable from the container if necessary.

1.4.3. Using conda

Another option is using the conda package manager. Nextflow can automatically create conda environments with most of the needed dependencies. This mode is selected by setting `-profile conda`. In this case, the following additional steps need to be completed:

- Install ScaleBio Tools
  - These are ScaleBio specific programs that are currently not available through conda.
  - Run `/PATH/TO/ScaleRNA/envs/download-scale-tools.sh`

    - This will install the pre-compiled binaries in `ScaleRNA/bin` (inside the Nextflow workflow directory), from where they will be available during workflow execution.
- If running from a sequencer runFolder (BCL) Illumina BCL Convert 3.9.3 is required to be installed (and available on `$PATH`).

See the Nextflow documentation for additional details of conda support in Nextflow.

1.4.4. Manual Dependency Installation

As a final alternative, the required dependencies can be installed directly, either by hand or using conda. A list of all requirements can be found in Chapter 5 or in the environment conda.yml files in `ScaleRNA/envs`. Once you have installed the dependencies, and made sure they are available on `$PATH`, the workflow can be run without any `-profile`.

1.4.5. Run the Workflow Test

A down-sampled dataset including usage instructions can be found here. Running this small sample dataset will allow you to rapidly test that you have installed the workflow correctly and your system requirements are met. This may also give you the opportunity to check compatibility of processed files with any downstream/secondary analysis.
NOTE: The raw sample data is stored on Amazon Web Services (AWS S3) and will be downloaded to your system automatically once you execute the Nextflow command (see here). Review your systems documentation for any additional steps that are required to download from AWS. Our test workflow reference genome and fastq files can be downloaded without an aws account or aws login credentials using the following commands here.

1.4.6. Workflow Parallelism

Individual steps of the pipeline are run as separate Nextflow tasks. Nextflow will launch as many parallel tasks concurrently as possible given available resources. Additionally, for many tasks, such as STAR alignment, the number of parallel threads inside a single task is configured dynamically to match available resources.
For large runs (> 100GB), the `--splitFastq` option enables an extra level of parallelism, based on the inputs provided. When starting from a bcl runFolder, the fastq files are split into individual PCR barcodes (96 for v1.1 3L RNA) and the majority of processes downstream and subsequently processed by barcode. When starting from fastq files, the workflow will execute barcode parsing in the same manner as starting with bcls with the resulting sample demultiplexed fastqs being split by each individual PCR barcode. All resulting read groups are merged in the end to produce the same outputs per sample as in the default case.
NOTE: When using `--splitFastq` and *–fastqOut* the resulting fastq files will be saved in the PCR barcode split format and seperated by samples. If desired, these files can be merged.

1.4.7. Running the ScaleRna nextflow workflow as a job with an executor/scheduler

The dynamic allocation of resources in Nextflow can be selected which “executor” configuration scope (see Executors — Nextflow 23.04.1 documentation). Nextflow also provides extensive documentation for hpc users such as SLURM or Moab.

https://www.nextflow.io/blog/2021/5_tips_for_hpc_users.html
https://www.nextflow.io/blog/2021/5-more-tips-for-nextflow-user-on-hpc.html

NOTE: By default (‘*local*’ executor), all tasks are launched on the same machine that Nextflow itself was started on, limiting the parallelism to the resources available on that computer (CPU cores and memory). Users with the minimum required resources should anticipate fold differences in runtime without use of executor function.

We recommend contacting your system's IT help desk to check if there is nextflow module (nf-modules) available for testing. These modules are rigorously tested and should provide guidance for the final nextflow.config setting and parameters. On AWS, Azure, or Google Cloud it can use *Batch* to achieve massive parallelism. The Nextflow documentation contains details about how each of these use-cases can be configured.

Chapter 2: Input Files

To run the pipeline the user will need to provide 3 inputs:

1. Sequencing reads (either bcl run folders OR fastq files R1, R2, and I1, I2)
2. Reference genome (STAR index)
3. Sample barcode table (samples.csv)

2.1. Sequencing Reads

NOTE: Illumina i7 and i5 index reads are required to be exported in fastq format. See below on how to generate index reads.

There are two formats reads can be used as input for the pipeline (ScaleRna/Fastq Generation):

2.1.1. BCL files as input – *preferred pathway*

If the ScaleBio RNA library was sequenced alone in a sequencing run or on a flow cell lane, the easiest way to run the analysis is to start directly from the sequencer RunFolder [`--runFolder`] (this is the sequencer output folder containing the *RunInfo.xml* file). In this case, the ScaleBio RNA workflow will internally generate FASTQ files appropriate for the pipeline input using Illumina *bcl-convert* (BCL Convert Support (illumina.com)).

2.1.2. FASTQ files as input [Read 1, Read 2, Index 1 (i7), Index 2 (i5)]

If the raw sequencer output is not available as BCL files, or the ScaleBio RNA library was multiplexed with other libraries during sequencing, the analysis can be started from FASTQ files generated ahead of time. In this case, please note the following:

- INDEX READS: Generating index reads in fastq format can be done using the *CreateFastqForIndexReads* option in Illumina *bcl-convert* OR by using ScaleBio provided samplesheets (see below)
- Each ScaleBio 3L RNA v1.1 library has 384 unique i7/i5 index combinations. Using ScaleBio samplesheets users should expect 96 X 4 fastq files (384 R1, R2, I1 and I2). Operating on a large number of files is optimal for ScaleRna and will result in decreased runtimes. However, concatenation of these files into a single 4 file faste set is also acceptable as input in case users would like to store less files per run. Simply use “SampleID” column to contain only “ScaleRNA”. Repeating names in the Sample_ID column is acceptable.
- Premade samplesheets (located here: [docs/examples/fastq-generation] are included in the ScaleRna pipeline and can be used for FASTQ generation with *bcl-convert*. Using these will create the FASTQ files with names as shown below for a full ScaleBio run. The resulting files will have the libName prefix and PCR well position.

```
ScaleRNA_1A_L001_R1_001.fastq.gz
ScaleRNA_1A_L001_R2_001.fastq.gz
ScaleRNA_1A_L001_I1_001.fastq.gz
```

```
ScaleRNA_1A_L001_I2_001.fastq.gz
```

- When generating fastq for extended throughput (ET) plates, the same logic above applies with each ET plate yielding 96 X 4 fastq files (384 R1, R2, I1 and I2) per ET plate sequenced. The resulting files will have the libName prefix, adapter primer set (plate ID), PCR well position.

```
ScaleRNA-AP1_1A_L001_R1_001.fastq.gz
ScaleRNA-AP1_1A_L001_R2_001.fastq.gz
ScaleRNA-AP1_1A_L001_I1_001.fastq.gz
ScaleRNA-AP1_1A_L001_I2_001.fastq.gz

ScaleRNA-AP2_1A_L001_R1_001.fastq.gz
ScaleRNA-AP2_1A_L001_R2_001.fastq.gz
ScaleRNA-AP2_1A_L001_I1_001.fastq.gz
ScaleRNA-AP2_1A_L001_I2_001.fastq.gz
```

2.2. Reference Genome

To generate a STAR (v2.7 or higher) reference genome index users require their respective genome fasta (.fa) file and gene annotation file (.gtf). Once the reference genome index is generated, these files are specified using a genome.json file, which contains the file paths and other parameters. Further information can be found here: Scale Reference Genomes.
ScaleBio has pre-built genomes for human (hg38), mouse (mm39), and a mixed human/mouse barnyard genome are available here:

- http://scale.pub.s3.amazonaws.com/genomes/rna/grch38.tgz
- http://scale.pub.s3.amazonaws.com/genomes/rna/mm39.tgz
- http://scale.pub.s3.amazonaws.com/genomes/rna/grch38_mm39.tgz

Download the appropriate reference file to your system, unpack (`tar -xzf GENOME.tgz`), and then specify the JSON file inside the directory (e.g. grch38/grch38.json) for the analysis [with the command line: `--genome`; or in the runParams.yml].

NOTE: You must download and unpack these files locally first. Do not specify the URLs to these TGZ files directly in the `--genome` option. Similarly, DO NOT use the example genome.json (`docs/examples/genome.json`) for real analysis beyond the test run. This test run aligns to a truncated version of the genome that is in place to ensure the pipeline is properly working in the user's environment and should not be used for analysis.

Building a new/custom STAR reference index for a different species or adding addition sequences/annotations requires the genome sequence (FASTA) and gene annotation (GTF). Adding additional sequences or modifying our pre-built genomes still requires a new reference genome build.

NOTE: When adding additional annotations to our pre-built genomes, please abide by the pre-built genome .gtf formatting including case-matching to ensure proper report generation.

Please see below the commands for generating a STAR index (v2.7 or higher) and new JSON file:

1. Build a STAR index, using at least STAR version 2.7:

```
STAR --runMode genomeGenerate --runThreadN 16 --genomeDir star.ref --genomeFastaFiles
GENOME_SEQUENCE.fa --sjdbGTFfile GENE_ANNOTATION.gtf
```

2. Generate a genome.json file with the following as a template:

```
{
  "name": "GRCh38",
  "speciesName": "Homo sapiens",
  "star_index": "path/to/star.ref",
  "gtf": path/to/GENE_ANNOTATION.gtf",
}
```

NOTE: When creating a custom genome, all features included in the GTF file will be used for gene expression analysis by the workflow. To exclude certain annotations (e.g. pseudo-genes), filter the GTF file before generating the STAR index.

2.3. Sample Barcode Table (samples.csv)

The ScaleRna library is pooled and needs to be further separated into sample specific files for analysis. A sample barcode table (e.g. samples.csv) file is used to list the samples included in an analysis run, their sample barcode (RT) sequences and optional sample-specific analysis parameters.

It is a comma separated file (CSV) with a header line (column names), followed by one sample per line. Please see below the minimum requirements:

Table 2: Contents of the sample barcode table entries

sample	barcodes	libName
Enter your desired sample name here	Enter the ScaleBio RT barcodes for the sample row	Fastq file prefix in order for ScaleRna to identify fastq files

Notes on the “barcodes” column used for demultiplexing samples

During analysis, sequencing data is first converted into library FASTQ files (libName column). If multiple samples were included in one sequencing library, the pipeline with further demultiplex based on the sample (RT) barcodes. These sample barcodes are defined by the wells on the RT Barcode Plate, for example see the table below.

Table 3: Example sample barcode table with 2 samples

sample	barcodes	libName
Jurkat-DMSO	1A-6H	ScaleRNA
Jurkat-500nM	7A-12H	ScaleRNA

Rules to abide by for sample naming

- "sample" and "libName" should consist ONLY of letters, numbers, dash (-) and dot (.).
- Underscores are not valid.
- "barcodes" is required if more than one sample is processed on the RT Barcode Plate.
- When running from pre-existing FASTQ file input, "libName" should match the first part of the FASTQ file name for this sample, e.g.: `ScaleRNA` for `ScaleRNA_*.fastq.gz`.

Rules to abide by for sample barcodes

1. The RT wells used for each sample are given as either:
   - An individual value: 1A
   - A range of wells: 1A-2H
   - A list of values or ranges, separated by semicolon (;): 1A;2A-2D
2. All ranges are read in column-wise order,
   a. e.g. 1A-2C, refers to 1A-1H (all of column 1) plus 2A, 2B and 2C.
3. Samples plated by rows they should input each RT well separated by a semi-colon
   a. e.g. Row 1 = 1A;2A; 3A;4A;5A;6A;7A;8A;9A;10A;11A;12A
   b. e.g. Row 4 = 1D;2D; 3D;4D;5D;6D;7D;8D;9D;10D;11D;12D

Chapter 3: Step-by-Step Overview of the Pipeline

In this chapter, we list the steps performed by the sequencing analysis pipeline, showing the order in which they are performed, and providing high-level information about the analyses performed at each step.

3.1. FASTQ Generation with ScaleRna

- This is performed only if a RunFolder (BCL files) is provided as input, rather than FASTQ files that were pre-generated using Illumina *bcl-convert*.
- All necessary FASTQ files are generated, including the i7 and i5 index reads (I1 and I2, respectively)
  - One set of FASTQ files, containing the raw unaligned reads, is generated for the entire sequencing run/lane.
  - Reads not matching one of the 96 expected ScaleBio RNA i5 PCR barcode sequences are filtered (into the *Undetermined* FASTQ files).
  - The i5 barcode is not typically needed at the library resolution for demultiplexing. If it is being used to separate the ScaleBio Single Cell RNA library from other

(non-ScaleBio) libraries sequenced on the same sequencing run, the user will need to generate FASTQ files before running the pipeline and use FASTQ as input (see *Sequencing Reads* for an example samplesheet).
    - If --splitFastq is enabled, a separate FASTQ file is produced for each PCR index resulting in 96 sets of files, that can be processed in parallel and merged at the end of the workflow.
- The *bcl-convert* step produces the standard Illumina reports on the number of reads per library and related metrics: Output Files (illumina.com)

3.2. FastQC

- This is an optional step to run QC reports on the input FASTQ files [--fastqc] using *FastQC (*Babraham Bioinformatics - FastQC A Quality Control tool for High Throughput Sequence Data.).
- The FastQC reports include information on output such as Q30 scores, adapter content, and base distribution for Read 1 & 2.
- The index reads are excluded from the report.

3.3. Barcode Parsing

- In this step, the three single-cell barcodes are extracted from the reads and error corrected using ScaleBio *bcParser* executable.
- This is included in the Docker/Singularity container or can be downloaded with `ScaleRNA/env/download-scale-tools.sh`
- Error correction is done against the list of expected barcode sequences, allowing up to 1 mismatch.
- If --splitFastq is enabled, each set of FASTQ files (e.g. each sequencing lane) is processed in parallel, otherwise they are pooled before this step.

3.4. Sample Demultiplexing

- If different samples were loaded into different wells of the RT Barcode plate during the ScaleBio Single Cell RNA Sequencing workflow, they are separated at this point for independent analysis.
- Implemented in *bcParser* together with Barcode Parsing.
- Uses the *samples.csv* sheet (see *Chapter 2: Input Files* for more information) to define RT barcodes for each sample.
- Outputs a pair of FASTQ files (barcode read and transcript read) for each sample.
- If --splitFastq is enabled and the reads were not already split on PCR barcode during *bcl-convert*, reads within each input FASTQ will be split based on RT barcode here. These subsets can then also be processed in parallel. RT barcodes corresponding to the same sample will be merged at the end of the workflow.

3.5. Generation of Library QC Report

- Produces an HTML report with QC metrics for the whole ScaleBio RNA library. This report focuses on barcode matching rates, read distribution across samples, and data quality across all barcodes (RT, Ligation, and Index PCR).

3.6. Read Trimming

- Trims any transcript reads that run into a Poly-A or Poly-T stretch using *cutadapt* (Cutadapt — Cutadapt 4.2 documentation).
- Cuts any read at the first occurrence of a stretch of 7 As within 8 bp.
- Transcript reads shorter than 16 bp after trimming are discarded.

3.7. Genome Alignment

- Aligns transcript reads to the genome using *STAR* (STAR/STARsolo.md at master · alexdobin/STAR (github.com)).
- STAR performs local sequence alignment, including spliced alignments to both known and novel transcripts / isoforms.
- Optionally outputs a sorted BAM file with alignments for custom downstream analysis or visualization [*--bamOut*].

3.8. Transcript Match

- Matches aligned reads to genes and transcripts for expression quantification.
- Included in *STARsolo* (STAR/STARsolo.md at master · alexdobin/STAR (github.com)).
- Matches reads to exons and introns, in sense direction only (relative to transcript annotation).
- Exon matches are prioritized over intron matches (50% read to exon overlap).
- Reads that match an exon in antisense direction are filtered, i.e., not counted even if they overlap an intron in sense direction.
- The gene annotation used is the one built into the STAR index (from the GTF used during index generation).

3.9. Unique Transcript Counts

- The number of unique transcripts for each cell and gene combination is counted.
- Included in *STARsolo* (STAR/STARsolo.md at master · alexdobin/STAR (github.com)).
- Transcripts are unique if they differ in Cell Barcode, mapped gene, or UMI sequence.
- If a read matches multiple genes, the count is divided among them in proportion to the unique counts of the genes. See STAR/STARsolo.md at master · alexdobin/STAR · GitHub for details ("PropUnique" method). This behavior can be controlled with the --starMulti option.

3.10. Cell Filtering

- Calculates per cell metrics and a unique transcript threshold to filter cell barcodes belonging to real cells from background.
- The cell threshold is determined following the method in *STARsolo*:
    - A preliminary list of possible cells is set (either based on the number of expected cells set in samples.csv, or the number of cell-barcodes with over 100 [*--minReads*] unique transcripts.
    - TopCount is set to the unique transcript count of the top 1 percentile [*--topCellPercent*] of preliminary cells.
    - The threshold is set to TopCount / 10 [*--minCellRatio*].
- CellFinder
    - CellFinder is an EmptyDrops-like cell calling approach. This option is enabled when the parameter *cellFinder* is true. Cells that are above the *minUTC* but below

the provided or calculated *UTC threshold* can potentially be "rescued" based on expression differences from the ambient RNA.

- Setting a minUTC threshold
    - A unique transcript count threshold is used when the parameter *UTC* is greater than 0 and *cellFinder* is not enabled. In this case every barcode with a total count >= *UTC* is called as a cell.

3.11. Generation of Gene Expression Matrix

- This output file is a sparse matrix (MTX file format) of the gene expression per cell (unique transcript counts).
- Both an unfiltered (all cell-barcodes) and a filtered (cell barcodes above the cell threshold) matrix are produced.
- Can be read into Seurat, ScanPy and similar.
- For more information on the specific metrics found in this report please see the links in *Chapter 4: Overview of Analysis Output Files*.

3.12. Generation of Sample QC Report

- A summary report with metrics for each sample.
- Includes mapping metrics, cell-counts, and sensitivity metrics.
- Includes distribution of RT barcodes for the specific sample.
- Produces a HTML document and a CSV file with metrics in text format.
- Note: for more information on the specific metrics found in this report please see the links in *Chapter 4: Overview of Analysis Output Files*.

Chapter 4: Overview of Analysis Output Files

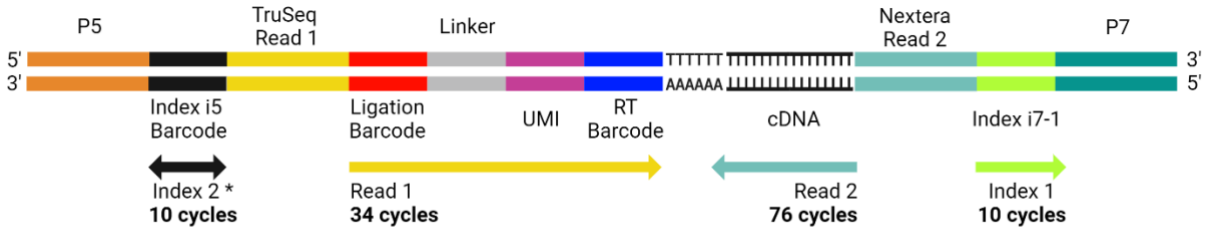
For more information about the specific metrics found in any of the output files please see the links below which contain the most up-to-date information.

- Outputs Overview
- QC Report Overview

Appendix A: RNA Library Structure and List of Barcode Sequences

The overall library structure for Single Cell RNA Sequencing Kit v1.1 is shown in Figure 2.

Figure 2: Library Structure for the Single Cell RNA Sequencing Kit v1.1 Assay



* orientation depends on sequencer and sequencing chemistry

Component	Size	Description
P5	–	Illumina P5 sequence (AATGATACGGCGACCACCGAGATCTACAC)
Index i5 Barcode	10 bp	Combinatorial Indexing
TruSeq Read 1	–	Illumina sequencing primer
Ligation Barcode	9 bp	Combinatorial Indexing. Half of the Ligation Barcodes are 9 bp, while the other half are 8 bp followed by a 'T' nucleotide for total length of 9 bp
Linker	7 bp	Fixed sequence (TCAGAGC) that is used to anchor UMI and RT Barcodes
UMI	8 bp	Unique Molecular Identifier
RT Barcode	10 bp	Combinatorial Indexing
Nextera Read 2	–	Illumina sequencing primer
I7	10 bp	Index i7, pool of 4 oligos for base balancing
P7	–	Illumina P7 sequence (ATCTCGTATGCCGTCTTCTGCTTG)

The full list of all barcode sequences can be found in the references directory of the workflow. The overall combinatorial cell-barcode is made up of a combination of RT Barcode, Ligation Barcode and the Index i5 Barcode, combined with the i7 index pool (Table 4) when using the Single Cell RNA Extended Throughput Kit v1.1.

Table 4: Index i7 Sequences present in a Base Balanced Pool per Index PCR Plate. Sequence directionality corresponds to bcl-convert expectation.

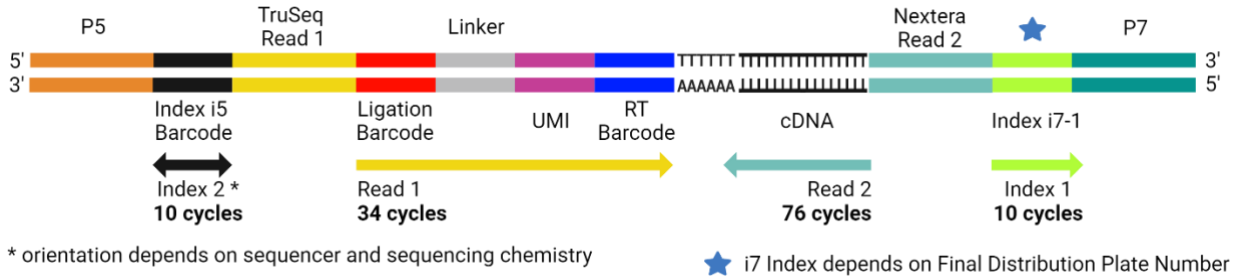
Barcode Name	Reagent Name	Sequences of i7 Oligo Pools			
RNA-A-AP1	Adaptor Primer i7-1	ATCTGCAGTC	CAGAAGCTAG	GCTCTCGCCT	TGAGCATAGA
RNA-A-AP2	Adaptor Primer i7-2	ACTATCTTGA	CTGGAACGCT	TGCCGTACAG	GAATCGGATC
RNA-A-AP3	Adaptor Primer i7-3	AGGTCAATTA	TACAACGCGT	GTAGTTCACG	CCTCGGTGAC
RNA-A-AP4	Adaptor Primer i7-4	AAGCCGGCTG	GTCGTTAAGC	CGAAGCTTCA	TCTTAACGAT

Appendix B: Single Cell RNA Extended Throughput Kit v1.1

B.1. Overview

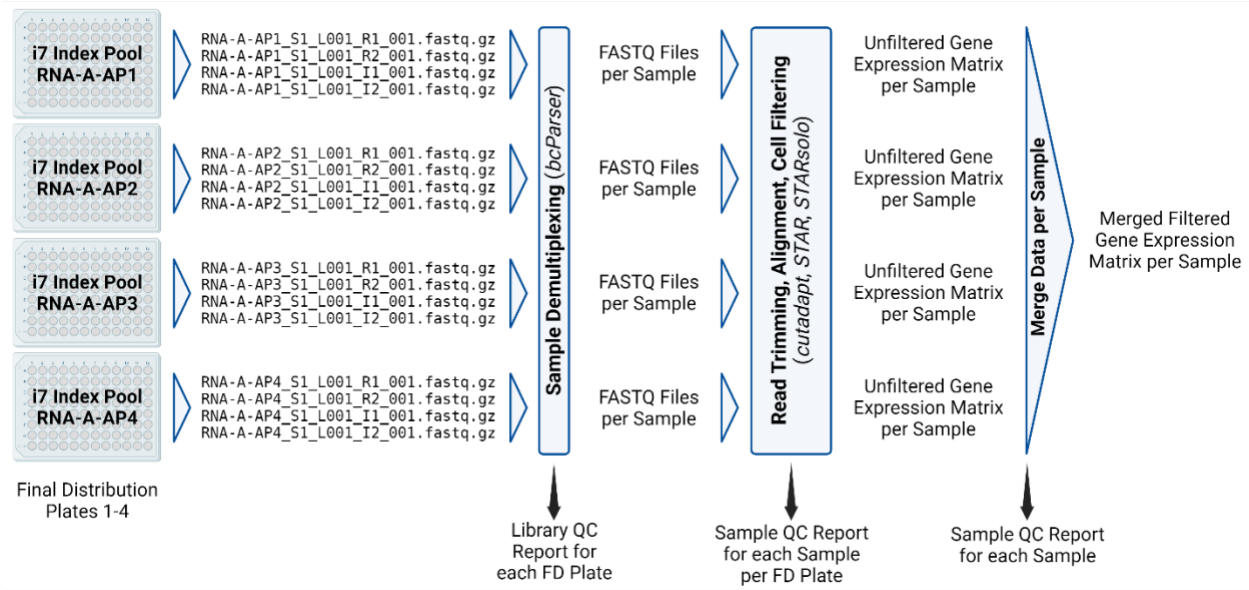
The ScaleBio Seq Suite: RNA Data Analysis Pipeline v1.4 (or later) supports the usage of the ScaleBio Single Cell RNA Extended Throughput Kit v1.1. This kit employs three additional Final Distribution Plates to recover more cells from the same experiment. The RT Barcodes, Ligation Barcodes and Index i5 Barcodes remain the same for each Final Distribution Plate, but the i7 index (position Index 1, Figure 3) receives a different set of i7 index sequences from the i7 primer pool used during library preparation (Table 4).

Figure 3: Adapted Library Structure for Single Cell RNA Extended Throughput Kit v1.1



The data is analyzed by generating separate "FASTQ libraries" for each Final Distribution Plate, analyzing each plate separately though the core workflow (e.g., *bcParser*, *STARSolo*) and then combining the outputs at the single-cell level (gene expression matrix and metrics). Cells originating from the same sample coming from different Final Distribution Plates are distinguished by appending the plate-name to the cell barcode in the gene expression matrix outputs.

Figure 4: Overview of Merging Multiple Final Distribution Plates



There are certain tweaks to the input files of the workflow required to ensure that the sample data across the various Final Distribution Plates is properly handled and merged for analysis.

B.2. Input Files: FASTQ

Please review chapter *Sequencing Reads*. If starting the workflow from BCL files, the FASTQ files required for the correct processing of the Single Cell RNA Extended Throughput v1.1 Kit are created automatically. A separate set of FASTQ files is created for each Final Distribution Plate (i7 index pool) as their own library, example:

```
ScaleRNA-AP1_S1_L001_R1_001.fastq.gz
ScaleRNA-AP1_S1_L001_R2_001.fastq.gz
ScaleRNA-AP1_S1_L001_I1_001.fastq.gz
ScaleRNA-AP1_S1_L001_I2_001.fastq.gz
```

If BCL files are not available, this set of FASTQ files needs to be created for each Final Distribution Plate to be merged in the downstream data processing.

B.3. Single Sequencing Run

If all Final Distribution Plates are sequenced together in the same sequencing run, the Nextflow analysis run can be setup as such:

- Create one samples.csv file for all plates and samples according to the instructions in chapter *Sample Barcode Table (samples.*csv).
- List each sample on each plate (repeating the sample name for each plate used).
- Set the libIndex column to the name of the used i7 index pool (e.g. RNA-A-AP1, see Table 4 for an overview of all pool options).
- Launch the workflow with the `--merge` option.
    - This will produce outputs for each individual Final Distribution Plate, as well as merged outputs combining all cells for each sample across all Final Distribution Plates.

The workflow will first split the data into the libraries belonging to each Final Distribution Plate, then process each sample per plate independently before merging outputs downstream (see Figure 4). By assigning a non-unique sample name that is present with multiple libIndex, we are telling the workflow to look in both Final Distribution Plates for cells belonging to that sample. Table 5 shows a samples.csv file example involving two samples (PBMC-1 and PBMC-2) processed on two Final Distribution Plates as indicated by the two different libIndex used (RNA-A-AP1 and RNA-A-AP2).

Table 5: samples.csv file Example with two Samples run on two Final Distribution Plates

sample	barcodes	libIndex
PBMC-1	1A-6H	RNA-A-AP1
PBMC-2	7A-12H	RNA-A-AP1
PBMC-1	1A-6H	RNA-A-AP2
PBMC-2	7A-12H	RNA-A-AP2

B.4. Merging Across Multiple Sequencing Runs

If the Final Distribution Plates were sequenced on separate sequencing runs, each plate first needs to be analyzed individually, and the results need to be later merged in a separate Nextflow run.

- Create a separate samples.csv file for each Final Distribution Plate, including the libIndex column with the i7 index pool used for each Final Distribution Plate (e.g. RNA-A-AP1, see Table 4 for an overview of all pool options).
- Run the Nextflow analysis workflow separately for each Final Distribution Plate from BCL files.
- Create another samples.csv that lists all samples and plates to be merged.
    - Add a new resultDir column that points to the workflow output directories for the individual plate runs.
    - See samples.merge.csv.
- Run the workflow with the --reporting and --merge option.
    - --reporting uses the previous alignment results instead of re-processing all reads.
    - --merge creates merged outputs for each sample across all plates as before.
    - Do not specify any input reads at this step (no --runFolder or --fastqDir).

Appendix C: Software Dependencies

- Java (11 or later)
- Nextflow (23.10 or later)
- Fastq generation:
    - bcl-convert 3.9.3
- For the full complement of dependencies, please visit our GitHub page: ScaleBio Seq Suite: RNA

Appendix D: ScalePlex

D.1. Overview

ScaleBio Seq Suite: RNA v1.6 (or later) provides support for the ScalePlex product within the ScaleRNA workflow. This is inclusive for extended throughput as well, and supports the full suite of processing and merging described in Appendix B. ScalePlex enables a higher degree of sample multiplexing by labeling material in fixation prior to loading onto the RT plate. Each well of the ScalePlex Oligo Plate is loaded with two ScalePlex oligos that, in combination, uniquely mark cells from that well. Each column and row have their own specific ScalePlex oligo associated, and their intersect per well dictates the expected combination which are used for counting and final assignment (represented in Figure D.1.). These oligo labels then propagate through the indexing strategy alongside the mRNA material per cell and are amplified after the workflow is completed into their own uniquely indexed library. The i7 sequence of the resultant library is specific to the ScalePlex fraction while the i5 index sequence is the same as the mRNA and corresponds to the well of the PCR plate the cell is from (Figure D.2.)

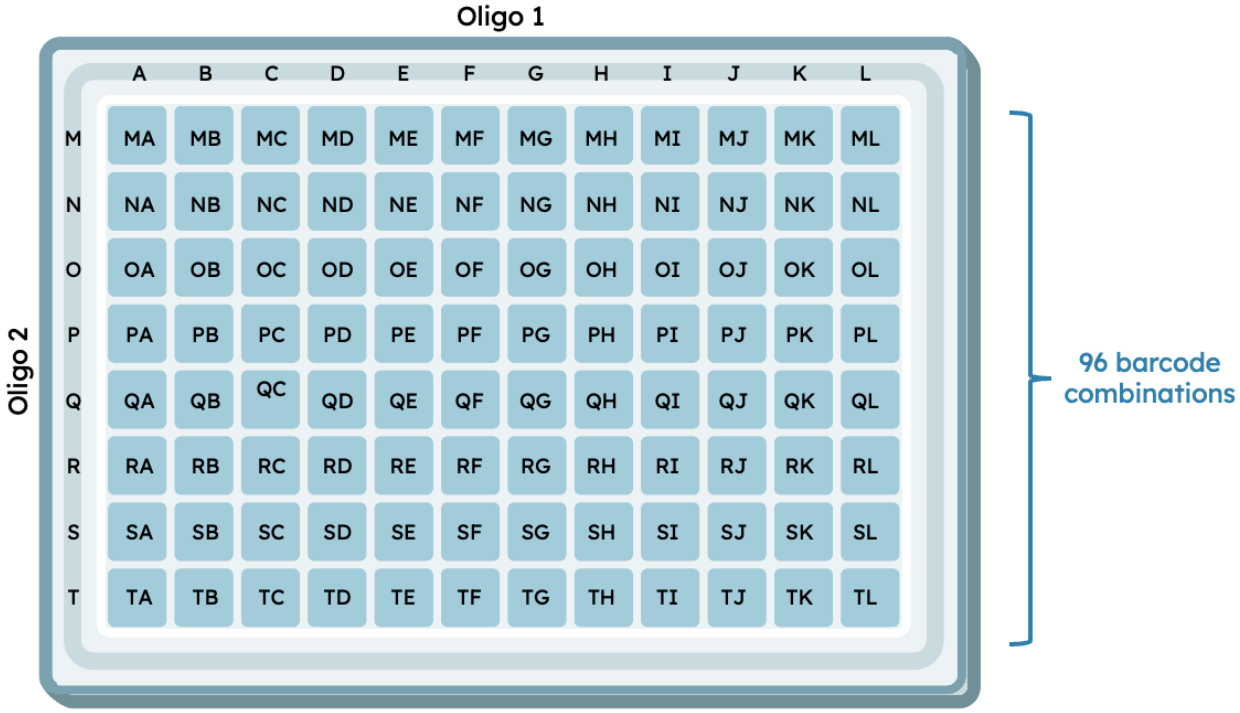


Figure D.1. ScalePlex Oligo Plate diagram, illustrating combinatorial ScalePlex oligo scheme

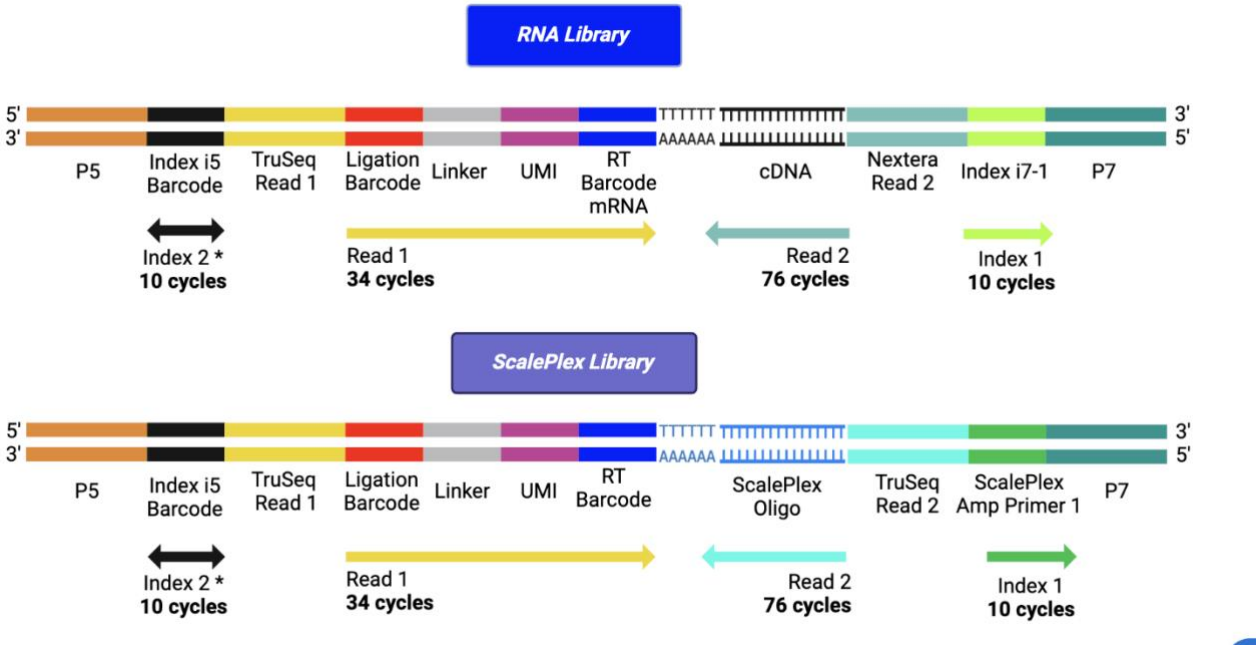


Figure D.2. RNA and ScalePlex Library Molecule Diagram

When processing the data, a handling of inputs splits sequencing reads into mRNA and ScalePlex library partitions that are then processed in parallel. The mRNA data is analyzed in the same way as our standard ScaleRNA workflow, but now with the processing of the ScalePlex library in tandem the final outputs and reports will reflect the presence of the ScalePlex assignment for samples as defined by their RT well position. The ScalePlex library reads undergo the same barcode sequence validation as do the mRNA, but with an added layer to ensure that the ScalePlex target sequences are also present in the reads. This is then used to generate a UMI counts matrix per individual oligo per cell, and finally we use these counts for ultimate assignment per cell back to their ScalePlex Oligo Plate position of origin.

D.2. Inputs

Firstly, a workflow level parameter of “scalePlex” must be set to true either in the in-line execution or runParams.yml to ensure the workflow proceeds with proper analysis. In addition, there are additional columns in the sample barcode table (samples.csv): scalePlexLibName, scalePlexLibIndex, and scalePlexBarcodes.

The same input requirements as discussed in Chapter 2 pertaining to the RNA portion of the analysis are also maintained in this configuration, but now with the additional sequencing read data that correspond to your ScalePlex library paired with your RNA library. We support two main input modes for your sequencing data, BCL files direct from the sequencing run folder or fastq files discussed below.

D.2.1. Changes to the Sample Barcode Table

The inclusion of the ScalePlex module for analysis in our ScaleRNA workflow has added several optional columns to the sample barcode table that can be engaged with to specify certain features regarding the ScalePlex Oligo Plate usage per sample as defined by RT, or for information regarding the sequencing reads used as input for the analysis. For the columns pertaining to the sequencing read information, we'll discuss in sections D.2.2. (scalePlexLibIndex) and D.2.3. (scalePlexLibName) of this handbook appendix. One additional optional column is independent of sequencing input mode, called "scalePlexBarcodes". This allows for the user to specify which wells or columns of the ScalePlex Oligo Plate were used when generating the pool of samples loaded into each sample as defined by their RT plate well position. The convention for specifying this is exactly as how we speciy the RT plate positions in the "barcodes" column, but critically here we are referring to the ScalePlex Oligo plate position rather than the RT plate position (Figure D.1.). See section 2.3 for further details regarding how that is encoded.

D.2.2. BCL file input

This configuration assumes that your ScaleRNA and ScalePlex libraries were sequenced together in the same sequencing run. Here, if the "scalePlex" parameter is enabled, and the samples barcode table (samples.csv) is configured to reflect the RNA library's RT sample designation, then with no additional modification the workflow will attempt to process the RNA and ScalePlex libraries. It does this by assuming that any of the potential ScalePlex library indices are present and will demultiplex reads for both RNA and ScalePlex and then subsequently split into samples as defined by RT for processing. Optionally, if the user supplies the libIndex as show in TableB.4. in the case of extended throughput, then the workflow will implicitly pair proper RNA library with its ScalePlex equivalent. Finally, the user may also supply a "scalePlexLibIndex" column to the file and explicitly supply the index sequence used in their study (references can be found here). There, the workflow will demultiplex based on the supplied sequence and proceed with downstream analysis.

sample	barcodes	libIndex	scalePlexLibIndex
PBMC-1	1A-6H	RNA-A-AP1	CAAGCGGAGC
PBMC-2	7A-12H	RNA-A-AP1	CAAGCGGAGC
PBMC-1	1A-6H	RNA-A-AP2	TATACCGAAG
PBMC-2	7A-12H	RNA-A-AP2	TATACCGAAG

The full list of i7 index sequences and their mapping to the corresponding RNA plate library can be found below:

Library	scalePlex lib Index Sequence
ScalePlex-A-AP1	CAAGCGGAGC
ScalePlex-A-AP2	TATACCGAAG
ScalePlex-A-AP3	GGCGAATCTC
ScalePlex-A-AP4	AACGGCCTAG

D.2.3. Fastq file input

This configuration supports using already generated fastq files as input, one set of fastq's for your ScaleRNA library, and a set of fastq's for your ScalePlex library associated with the RNA experiment. These fastq files are to be compliant with the naming and structure described in Chapter 2, with the "LibraryName" field unique to the RNA and ScalePlex fractions respectively. Once the files are organized, in the sample barcode table (samples.csv) document used for input, the user will add an additional column entitled "scalePlexLibName" whose values will correspond to the Library Name of the ScalePlex library Fastqs (Example scaleplex.samples.csv).

sample	barcodes	libName	scalePlexLibName
PBMC-1	1A-6H	ScaleRNA-AP1	ScalePlex-AP1
PBMC-2	7A-12H	ScaleRNA-AP1	ScalePlex-AP1
PBMC-1	1A-6H	ScaleRNA-AP2	ScalePlex-AP2
PBMC-2	7A-12H	ScaleRNA-AP2	ScalePlex-AP2

D.3: Amended Trimming SOP for RNA in ScalePlex

The RNA portion of the workflow described in Chapter 3 remains untouched with the inclusion of ScalePlex with one important distinction. In the absence of ScalePlex, it is our default to first trim sequencing reads using cutadapt to remove bases corresponding to PolyA or PolyT stretches in the read. With the inclusion of the "scalePlex" set to True, we will additionally trim bases that correspond to the PCR handle of the scalePlex oligo in the RNA library. The goal of this trimming is to deplete any scalePlex molecules that could exist in the RNA fraction that could contribute to artificial alignment and quantitation. The design and modifications of the scalePlex oligo limit this fraction substantially, but for the purposes of data sanitation the trimming is still included.

D.4. ScalePlex Assignment

D.4.1. Assignment Methodology

As previously mentioned in D.1. Overview, the inclusion of ScalePlex to your library generation workflow involves the addition of two ScalePlex oligos, one specific to each row of the ScalePlex Oligo Plate and another specific to each column. In the resultant sample's analysis, we employ the oligo's unique combination per cell and use this to map back to the original well of the ScalePlex Oligo Plate that the cell was subjected to the labeling.

Once the ScalePlex oligo UMI counts are generated for each sample, we constrain the barcodes analyzed to be only those that passed cell calling in the RNA workflow and perform assignment. There are two methods of assignment that are supported by the workflow, controlled by the "scalePlexAssignmentMethod" parameter.

The default approach, when the parameter is set to “bg”, we build an expected background count estimation for each ScalePlex UMI count across the whole sample’s cells (sample as defined by RT at this stage), and perform a statistical test of the observed counts against that background. If the top two detected ScalePlex labels both pass the background and correspond to an expected combination of labels that align back to a well of the ScalePlex Oligo Plate, then the cell is assigned. Optionally, there is a “scalePlexPercentFromTopTwo” parameter that controls a threshold based on the ratio of ScalePlex UMI’s for any cell coming from the top two detected (given that we are doing a combinatorial based assignment, the top two should both be very enriched over total). By default this is set to zero, but provides the user a means to improve their assignment confidence to remove any higher background ScalePlex barcodes if need be.
The second method of assignment is a simpler approach that leverages the fold change of a cell’s second highest ScalePlex count compared to the third highest.

When “scalePlexAssignmentMethod” is set to “fc”, the workflow check to see if the fold change of the second highest ScalePlex count to the third highest ScalePlex count is > scalePlexFCThreshold parameter value (default 2) and if so the assignment proceeds. The same validation of expected pairs is also enforced as in the background based method.

NOTE We recommend using the default background based assignment approach for most cases. The fold change based assignment is best suited with datasets with low overall ScalePlex counts per cell, where the background estimation can run into issues in the testing.

D.4.2. Failure Modes for Assignment
While the assignment logic was described above in section D.4.1., there are several failure modes for cells to fail assignment and thus are unable to be associated with the ScalePlex Oligo Plate well of origin and are detailed below:

1. Max_Fail
    a. A cell fails assignment and is labeled "Max_Fail” in the “bg” method when at least one of the top two ScalePlex oligos detected in that cell were not among those that passed the statistical background test for that cell
2. Indeterminate
    a. A cell fails assignment and is labeled “Indeterminate” in the “bg” method when there were no ScalePlex oligo counts per cell that passed when compared versus the estimated background
3. Enrich_Fail
    a. A cell fails assignment and is labeled “Enrich_Fail” in the “bg” method in the case that " scalePlexPercentFromTopTwo" is enabled and fails to pass the enrichment threshold set by the user supplied value.
4. A cell fails assignment and is labeled “Enrich_Fail” in the “fc” method in the case that the fold change of the second highest ScalePlex oligo counts over the third highest ScalePlex oligo count was < 2. Unexpected

a. A cell fails assignment and is labeled “Unexpected" when it passed all of the other assignment criteria for either the “bg” or “fc” method but the two passing ScalePlex oligos in combination do not match with a combination that either is:
    i. not on the plate in the first place such as the top two being from two different columns rather than a combination of a row and column
    ii. If the user supplied an entry for “scalePlexBarcodes" for the sample in questoin and the assignment falls outside the constraint specified

The result of the assignment per cell can be found in the “assigned_scaleplex” column of the SAMPLE.ScalePlex.allCells.csv in the samples directory, more details can be found in the next section D.5. Outputs

D.5. Outputs

The full list of outputs and descriptors can be found here: LINK

Briefly, engagement of the ScalePlex module for analysis will add an additional library qc report for the ScalePlex sequencing reads. Additionally, for each sample as defined by RT well designation (specified in the samples.csv) the sample specific report will feature a new “ScalePlex” tab that reports that summarize the performance of your ScalePlex sample material.

The main output that is relevant to downstream analysis is in the SAMPLE.ScalePlex.allCells.csv, which contains per-cell metadata for the ScalePlex fraction for each cell barcode in the analysis. A full description can be found in the link above, but it is worth mentioning specifically here that the “assigned_scaleplex” column is the place in which cells have their ultimate assignment indicated. When performing downstream analysis of ScalePlex data, the user should incorporate the SAMPLE.ScalePlex.allCells.csv into the metadata of their analysis object and can then leverage the labels in the "assigned_scaleplex” column to subset or summarize their analysis based on the true sample of origin based on the assignment.

Appendix E: Custom Reference Genome

E.1 Reference Genome

The workflow uses a genome reference and gene annotation for analysis. Reads are aligned to the full genome sequence and then matched to annotated transcripts using the RNA-seq aligner STAR. Hence specifically a STAR (v2.7.4 or higher) genome index is required, including both the sequence (fasta file) and gene annotation (GTF file).

E.2 Pre-built Genomes

ScaleBio pre-built reference genomes for human (hg38), mouse (mm39), and a mixed human/mouse barnyard genome are available here:

- http://scale.pub.s3.amazonaws.com/genomes/rna/grch38.tgz
- http://scale.pub.s3.amazonaws.com/genomes/rna/mm39.tgz
- http://scale.pub.s3.amazonaws.com/genomes/rna/grch38_mm39.tgz

All of these use ENSEMBL annotations, subset to only "`protein_coding`" and "`lncRNA`" and "`IG_?_gene`" biotypes.

Download the appropriate reference file to your system, unpack (`tar -xzf GENOME.tgz`), and then specify the JSON file inside the directory (e.g. grch38/grch38.json) for the analysis [with the command line: `--genome`; or in the runParams.yml].

NOTE: You must download and unpack these files locally first. Do not specify the URLs to these TGZ files directly in the `--genome` option. Similarly, DO NOT use the example genome.json (`docs/examples/genome.json`) for real analysis beyond the test run. This is a truncated version of the genome used only to ensure the pipeline is properly setup in the user's environment and should not be used for real data.

E.3 Creating a New Genome Index

An example command to build a STAR index:

```
STAR --runMode genomeGenerate --runThreadN 16 --genomeDir star.ref --
genomeFastaFiles Homo_sapiens.GRCh38.dna.primary_assembly.fa --sjdbGTFfile
Homo_sapiens.GRCh38.103.biotypeFiltered.gtf
```

- star.ref is the output directory which will contain the STAR index
  - The whole directory with all files is required during analysis
- `Homo_sapiens.GRCh38.dna.primary_assembly.fa` is the genome sequence in fasta format
  - Generally a reference genome with only 'primary' contigs (chromosomes) should be used, i.e. no alternative haplotypes or sequence patches (e.g. Ensemble 'primary')
- `Homo_sapiens.GRCh38.103.biotypeFiltered.gtf` is the gene annotation in GTF format.
  - This has to match the genome sequence (fasta), i.e. same genome version and chromosome names (watch for `chr1` vs. `1`)
  - All genes included in the GTF file will be used for gene expression analysis. To exclude certain annotations (e.g. pseudo-genes), filter the GTF file before generating the STAR index.
  - STAR specifically uses "exon" annotations in the file, linked into transcripts by "transcript_id" and grouped into genes by "gene_id". Optionally "gene_name" is used to output human-readable gene-names. All fields are case sensitive!
- See "Generating genome indexes" in the STAR manual (STAR/doc/STARmanual.pdf at master · alexdobin/STAR (github.com)) for additional details and advanced options.

The ScaleRNA analysis workflow uses a genome.json file, the specify the genome index location and other related parameters. To use a new reference genome, create a new json file with the path to the STAR index and GTF file and specify for the workflow with the `--genome`. See Scale Reference Genomes for additional details on the `genome.json` file format.

E.4 Adding Extra Sequences or Transgenes

If the sample being analyzed includes a trans-gene or similar additional transcript being captured, a new reference genome including this extra sequence is required. This is true whether the extra transcript is targeted through the Poly-A tail or with a custom RT primer. In general this 'alignment-based' approach will work with one or multiple extra transcripts, as long as they are sufficiently unique from each other and the rest of the genome to be uniquely mappable with the RNA reads. Standard alignment is not optimal for short barcodes or similar sequences such as CRISPR guides; See ScaleBio/ScaleCRISPR: ScaleBio Seq Suite: CRISPR Workflow (github.com) for these use-cases.

First the sequence for the trans-gene should be added as an extra contig to the end of the fasta file, e.g.

```
>chr1
      TAACCCTAACCCTAACC...
      >chr2
      ...
      >transgene1
      ATGGTGAGCAAGGGCGAGGAGGATAACAT
```

Second an entry in the GTF file is needed. Typically this can simply be a single 'exon' spanning the entire extra sequence

```
transgene1       custom exon  1     1000  .     +     .     gene_id
"transgene1"; transcript_id "transgene1";
```

This will count all reads mapping to the transgene sequence as expression.

These fasta and GTF files can then be used to build a new STAR index and genome.json (see above). The extra transcript will be included in all outputs together with regular annotated genes.

Document Revision History

Revision	Revision Date	Document ID	Changes
Rev A	Jun 2023	1020803	Initial release
Rev B	Dec 2023	1020803	Indexing update v1.4
Rev C	Mar 2024	1020803	Bug fixes and format update
Rev D	Sep 2024	1020803	ScalePlex update v1.6
Rev E	Jan 2025	1020803	Custom reference genome instructions