MiCV guide

Prepared by: Nigel Michki (MiOmics, Inc.)
MiCV version: 0.8.5
First draft: 2020-10-06
Last revised: 2021-01-01

Notices/disclaimers

Certain third party brand names and/or trademarks are referenced in this work solely for clarity, and MiOmics Inc. does not claim any rights in those third party marks or names.

THIS GUIDE AND ITS RELATED SOFTWARE (MiCV) ARE PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS, SERVICE PROVIDERS, OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE GUIDE OR SOFTWARE, OR THE USE OR OTHER DEALINGS IN THE GUIDE OR SOFTWARE.

This page intentionally left blank

Contents

Quick start

Register and login:

1) Navigate to https://pro.micv.works/register to make an account with your email address
2) Navigate to [Account] and subscribe to a monthly plan

Upload and preprocess your data:

1) Navigate to the [Load] tab in MiCV
2) Move your 10X/mtx-style counts matrix directory (output from CellRanger, usually called "filtered_feature_bc_matrix") into a zip file
    - Alternatively, if you have an hdf5 file from MiCV/scanpy, place that in the zip file
3) Drag the zip file into the dashed data upload box on the [Load] tab
4) Click [Analyze data] and go stretch your legs for a minute while MiCV analyzes your data

Generate a summary report:

1) Navigate to the [Summarize] tab in MiCV
2) Click [Generate report] and go grab some coffee - this will take 2-10 minutes, depending on the number of cells and number of automatically identified clusters in your dataset
3) Click [Download report] to download your summary report

Identify marker genes:

1) Navigate to the [Explore>Markers] tab
2) Select [leiden] in the selection box above the top graph
3) Scroll down and select [all] in the selection box on the bottom of the page
4) Click [Recalculate] and wait until a plot appears
5) Save the plot by right-clicking on it and selecting [Save image]
6) Save a ranked list of all marker genes by clicking the [Download table] button

Calculate pseudotime trajectory (optional)

1) Navigate to the [Pseudotime] tab
2) Select [# genes] in the selection box on the [Pseudotime] tab
3) In the top-right corner of the plot, select the [Lasso] selection tool (dashed curved line)
4) Using this tool, click-and-drag in a circle around a small group of cells with high #genes
    - These are often good candidate *starter cells* in differentiation pathways
4) Click the [Recalculate pseudotime] button
5) Observe pseudotime trajectory by selecting [pseudotime] in the selection box

Explore and manually annotate cells

1) Navigate to the [Annotate] tab
2) In the top-left box, select [user_0]
    - This is a blank data column that will store your manual cluster annotations
3) (optional) To plot pseudotime trajectories, select [pseudotime] in the top-right selection box
4) To plot single gene expression projections, find the single gene selection box in the bottom-left of the page and use it to search for a gene symbol. Make sure [single-gene] is selected
5) To view data about your selected gene, your organism from the organism selection box found below the gene expression projection plot
6) To plot multi gene expression projections, select up to 3 genes from the multi gene selection box in the bottom-left of the page. Make sure [multi-gene] is selected
    - This plots in cyan, magenta, and yellow, the expression of gene 1, 2, and 3 in your list, respectively. Expression for each gene is normalized to its own maximum, range [0,1]. It is meant to help visualize gene *co-expression* patterns, not absolute expression levels.
7) To plot the expression distribution of genes in your data, select your genes in the selection box on the bottom-right of the page. Plots are generated at the very bottom of the page
8) (optional) To plot the pseudotime expression trend of genes in your data, select genes as in step (7), but select a [pseudotime branch] in the selection box below the gene selection box.
9) To select cells, use the selection tools in *any* of the plots to select cells of interest
    - The logical AND/set INTERSECTION of cells across all plots will be highlighted in the top-left plot, where you selected [user_0] in step (1).
10) Once satisfied with your selection, click the [Define new cluster] button to update [user_0] with a new, manually annotated cluster
11) Repeat until you have clustered all of your cells based on your own expert analysis
12) Go back to the [Explore>Marker genes] tab and select [user_0] in the top selection box, then repeat the steps in the [Identify marker genes] section to calculate new marker genes

Save and export your data

1) Navigate to the [Save] tab
2) Choose to either keep all cells (default) or a subset of cells
3) Enter a name to save this analyzed dataset under in your MiCV account
4) Click the [Save dataset to MiCV] button
    - Validate by returning to the [Load] tab - a new dataset should be selectable in your list of saved datasets
5) Generate an export file containing all cells in your data by clicking the [Generate file] button
6) The [Download file] button should now be selectable - click it to download your data in hdf5 format (readable by scanpy, seurat, and MiCV)

Useful definitions

UMIs: unique molecular identifiers; DNA barcodes unique to one captured mRNA molecule. Often referred to as (gene) *counts*.

Counts matrix: a cell x gene matrix containing integer values representing the number of UMIs/counts for each gene in each cell.

Expression matrix: a filtered, scaled, and log-transformed (i.e. ln(1 + UMIs)) version of the counts matrix, fairly representing each gene's expression level in each cell.

Data upload

Accepted file formats

Users employ a variety of mapping and analysis tools to work with their data. To better accommodate as many users as possible, MiCV supports a wide range of file formats for importing (uploading) data into MiCV.

Currently, MiCV can recognize and attempt to parse the following data formats:

1) **mtx** (10Xv2/v3 **CellRanger**) -style counts matrix directories
2) **h5ad** files containing AnnData objects from **scanpy** and **MiCV**
3) hdf5 files in **loom**-format from **Seurat** and other workflows
4) **gzipped** output directories from **UMI-tools**

Most users who have generated data on the 10X Chromium scRNA-seq platform will have mtx-style counts matrices (option 1 above).

Users who have previously performed analysis using Seurat will need to save their data to the loom format (option 3 above). Information on that process can be found on the Seurat website: https://satijalab.org/seurat/v3.0/conversion_vignette.html

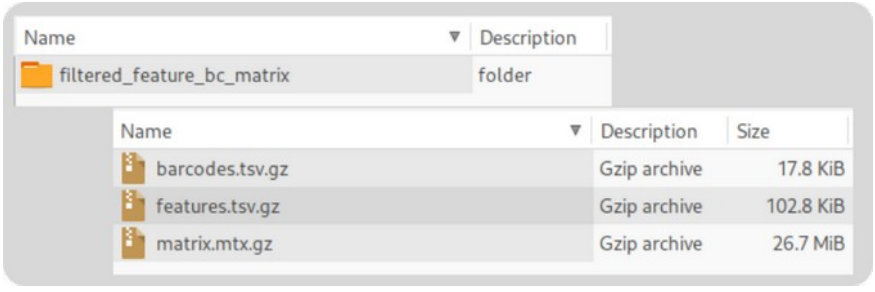
Preparing data for upload

MiCV can recognize and parse one or more datasets at a time. We import them into MiCV by placing them in a zip file, then uploading it on the [Load] tab in MiCV. Here, we will walk through 2 examples.

Example 1: A single, mtx/CellRanger-style counts matrix directory

Users working with data straight from CellRanger will be provided with a directory called "filtered_feature_bc_matrix" as an output. It is structured as follows:

- filtered_feature_bc_matrix
  - barcodes.tsv.gz
  - features.tsv.gz
  - matrix.mtx.gz



To upload this directory to MiCV, simply compress it into a zip file. Instructions on how to compress a folder vary by operating system. In general:

Windows:

https://support.microsoft.com/en-us/help/14200/windows-compress-uncompress-zip-files

1) Right-click the folder (“filtered_feature_bc_matrix”)
2) Select [Send to], then select [Compressed (zipped) folder]

Mac OSX:

https://support.apple.com/guide/mac-help/compress-uncompress-files-folders-mac-mchlp2528/mac

1) Control-click the folder (“filtered_feature_bc_matrix”)
2) Select [Compress]

Linux:

Varies by distribution. In general, right-click the folder and select [Compress]. Alternatively, in a terminal window, navigate to your folder’s parent directory and run:

```
$zip folder_name.zip folder_name
```

Example 2: Multiple datasets, representing different conditions, in different file formats

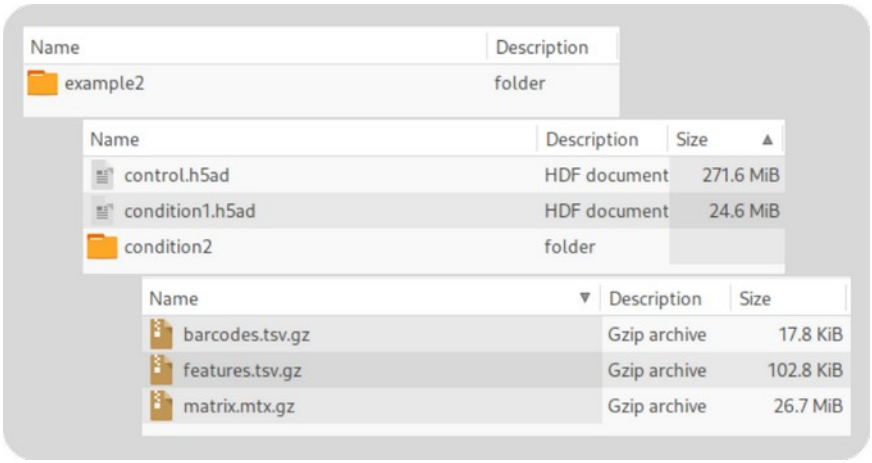
For the purposes of this example, consider the following experimental design/dataset list:

- Control, in h5ad format
- Condition 1, in h5ad format
- Condition 2, in mtx-style format

Assuming these samples are meant to be interpreted together, users can upload all 3 samples together at the same time and MiCV will concatenate them into a single data object, labelling cells from each file with separate ‘batch’ labels (so the datasets are never mixed up).

To prepare these datasets for upload, make a new folder on your computer and copy all 3 datasets into this folder. It should be structured as follows:

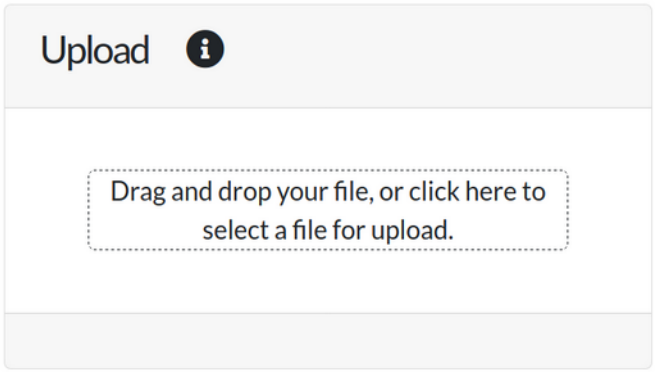
- example2
  - control.h5ad
  - condition1.h5ad
  - condition2
    - barcodes.tsv.gz
    - features.tsv.gz
    - matrix.mtx.gz



Once assembled, compress the folder you created (‘example2’, in this example) using the instructions for your specific operating system provided in example 1 above.

Uploading your data

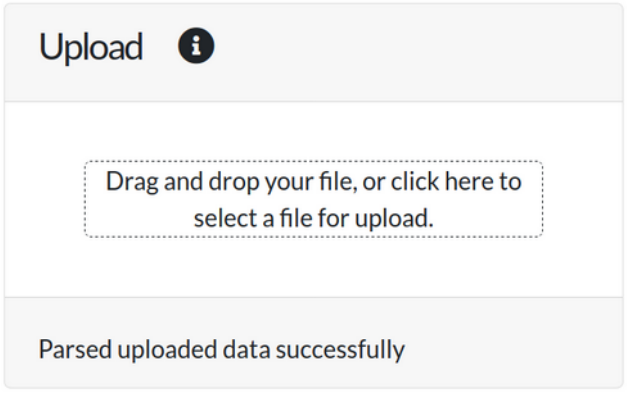
Once you have prepared a single zip file containing your dataset(s), navigate to the [Load] tab in MiCV. The following window should be shown on the right of the screen:

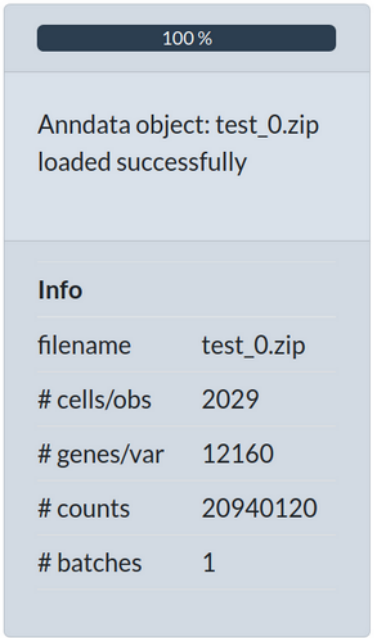


Drag and drop your zip file into the dashed box, or click on the dashed box itself to bring up a file selection menu. The upload process should begin, indicated by the presence of a spinner on-screen:

The upload process may take a few minutes, depending on the speed of your internet connection, the size of the data files, and the file formats provided.

After your data have been uploaded successfully, a few items on-screen will update:





Troubleshooting

Data integrity checks

- Verify that the number of batches reported in the colored status panel match the number of datasets you uploaded in your zip file.
- Verify that the number of cells reported in the colored status panel matches what you expect from previous analysis or a QC report from CellRanger/other software.
- Likewise, verify that the number of unique genes reported matches what you expect.

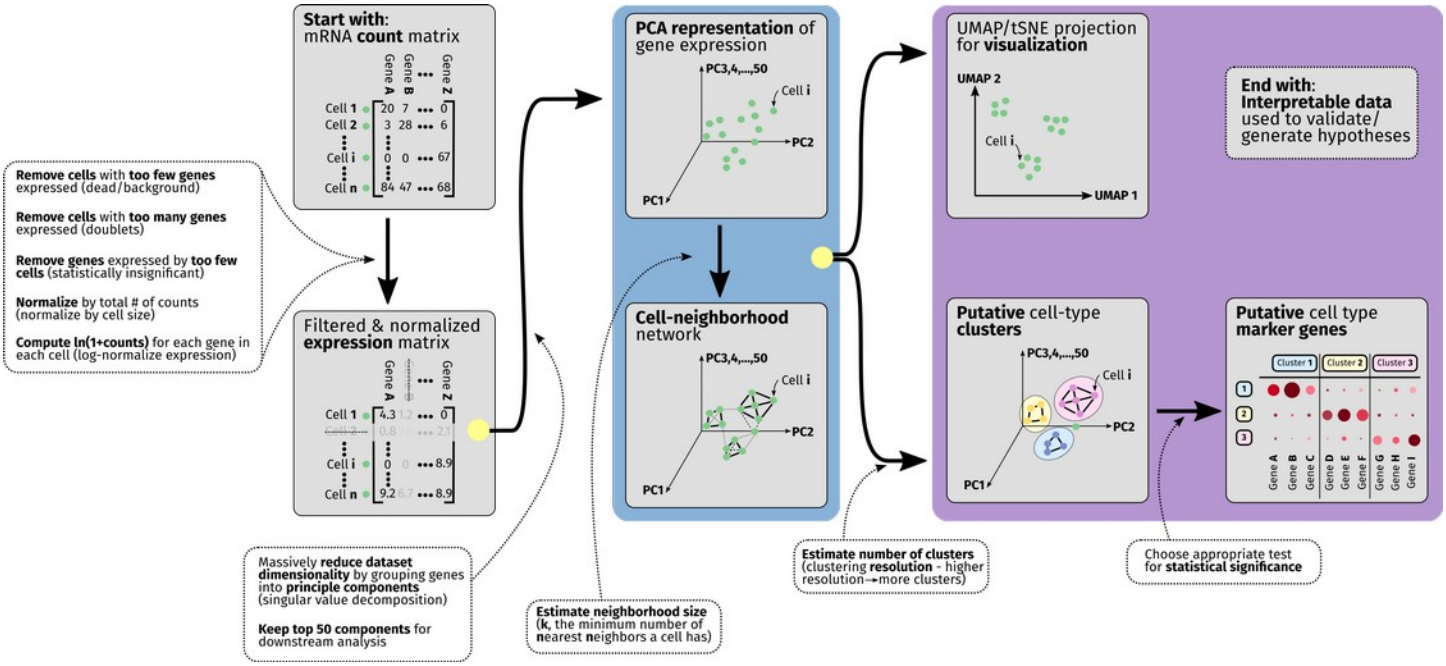
When things go wrong

- When uploading a single dataset, verify that you can open it correctly in other software (scanpy/seurat/Loupe Cell Browser/etc.) without needing any modifications
- When uploading multiple datasets, verify each individual dataset independently as above before compressing them together into your zip file
- MiCV attempts to identify and intelligently handle NaN values in uploaded datasets. However, they can still be a source of problems. Consider manually filtering out NaN values from your data, and make sure that every row and column has a label (when applicable).
- If a spinner never appears on-screen (or goes away immediately), there is likely an issue transferring the data. Firewall and web-browser security restrictions can sometimes block outgoing data transfers from your computer. Open your task manager/system monitor to view outgoing network traffic, and attempt to upload data to MiCV. If no data leaves your computer, it is likely being stopped by your computer. Consider trying a different web browser (Firefox is primarily what we use to develop MiCV), contacting your IT department, and/or letting us know about your issue! We will be happy to help.

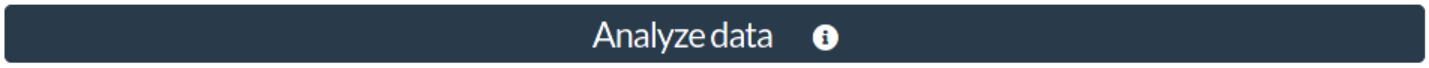
Data preprocessing

A typical scRNA-seq automated analysis pipeline

Performing analysis on raw counts matrices representing the results of an scRNA-seq experiment is not trivial, but is fairly standardized. Below is a diagram depicting all of the steps found in the MiCV automated analysis pipeline. This analysis pipeline is largely the same as that proposed by the authors of scanpy and seurat, 2 popular scRNA-seq analysis libraries written in Python and R, respectively.



Before we get into the details, it is important to point out that this pipeline usually does not need any modifications. In MiCV, after you have uploaded your data, simply press the [Analyze data] button, found underneath the upload box:



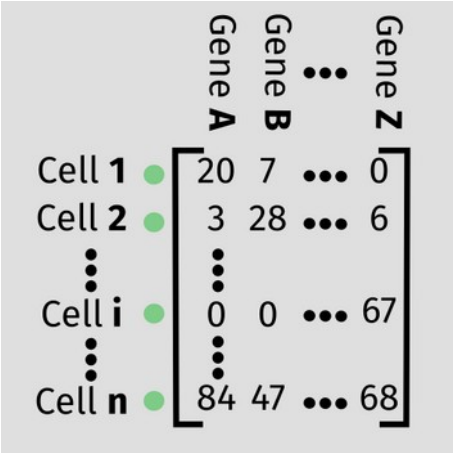
After pressing this button, all of the analysis steps outlined in the diagram above will be completed for you with default, well-tested parameter values. You will then be ready to view projection plots and identify marker genes for putative cellular subtype clusters.

However, it is still useful to understand what is going on 'under the hood', as your specific experimental design or questions may require some parameter fine-tuning for maximal clarity and robustness. We summarize here the major preprocessing steps and related parameters that MiCV implements here. We also recommend reading through both the scanpy and seurat tutorials for additional context and training.

Raw data - the counts matrix

After 'mapping' your raw sequencing reads using a tool like CellRanger, STAR, salmon, and many others, you should have in your possession what is known commonly as a 'counts matrix'. It is structured as follows:

- Unique cells make up the rows of the matrix
- Unique genes make up the columns of the matrix
- Each integer in the matrix represents the number of unique RNA molecules (UMIs) associated with a specific gene were captured within a specific cell



In this case, a single cell's transcriptome would be represented by a single, full-length row of integers, representing how many molecules of each gene's mRNA were present in the cell.

The rows/columns of this matrix can sometimes be swapped, but this makes no meaningful change to the data aside from changing how we parse the data on import.

Filtering the counts matrix

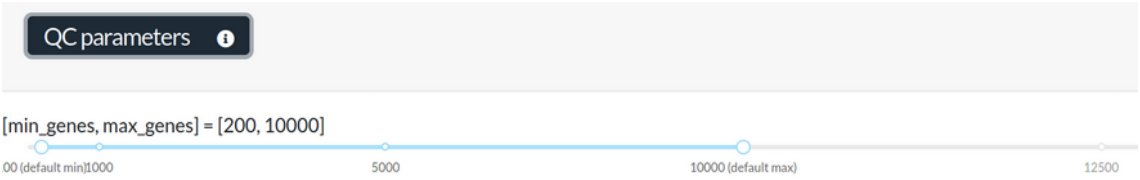
To get a sense of 'scale' for this data, remember that many organisms have upwards of 20000 unique genes in their genome that we might find expressed as mRNA. Thus, for a typical scRNA-seq experiment with, say, 1000 cells, we should have 1000 x 20000 = 20000000 = 20 million data points in this matrix. Not only is data this large computationally expensive to work with - it is also *noisy*, owing largely to the fact that the data is *high dimensional* (20000 dimensional, in this case).

One way to reduce dataset dimensionality is to remove outlier cells and genes. We typically call this the *filtering* step, and it comes first in most analysis pipelines.

In MiCV, we first filter out cells based on the number of unique genes they express:

- Cells that express very few genes relative to the median are likely not cells at all but background RNA captured in the mRNA capture process (or cells that were unhealthy and should likely be excluded)
- Cells that express very many genes relative to the median are *often* doublets: 2 cells that got stuck together during the mRNA capture process and are being treated as 1 cell mistakenly

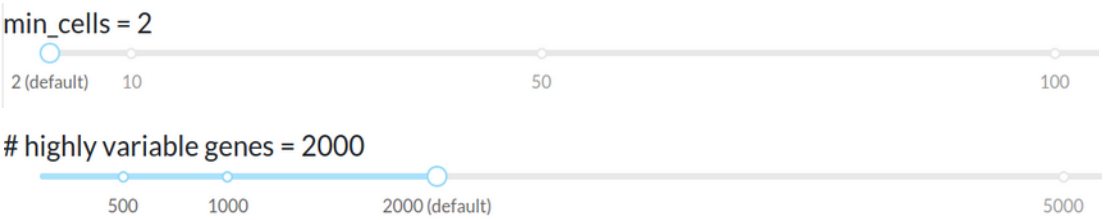
We can filter cells in this way using the slider tool in MiCV on the [Preprocess] tab, under the [QC parameters] dropdown:



Likewise, we can filter out genes based on how many cells they are expressed in:

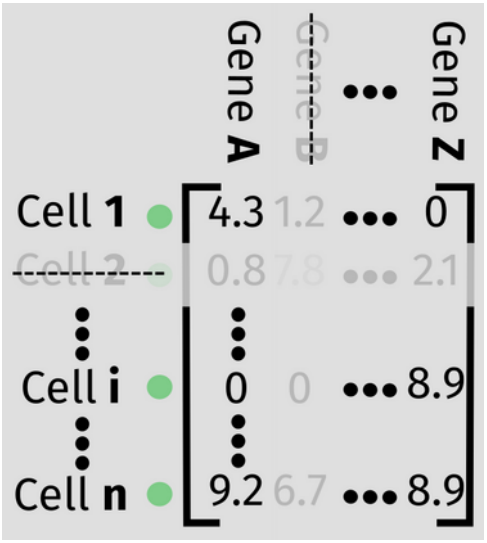
- Genes expressed in very few cells (fewer than 2, typically) are low-expressing and difficult to establish statistical significance with
- Unlike with cells, we do not typically remove genes expressed in all cells. Rather, we select a set of *highly variable genes* to focus in on for downstream analysis. This enables us to greatly reduce the dimensionality of our data while basing our analysis on the factors that make our data unique (the highly variable genes).

We can filter genes in this way using the slider tools in MiCV found on the [Preprocess] tab, under the [QC parameters] dropdown:



Normalizing the counts matrix

In MiCV, normalization and log-transformation are done in the background for you; this section is purely for your reference.

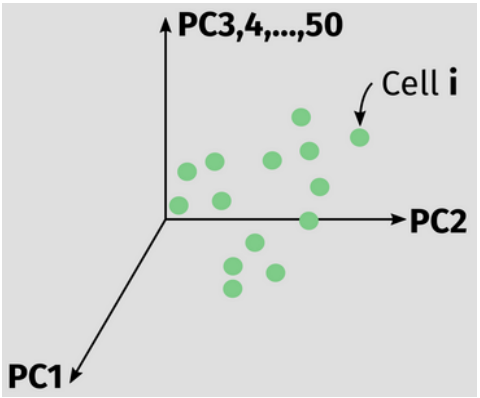


The unique molecule counts in this matrix represent how many mRNA molecules were present in a cell. However, these counts do not directly represent gene *expression* in the cell, as the biochemical reactions that drive gene expression levels are not linearly related to the number of mRNAs in the cell. Thus, we need to take some steps to correct for this and normalize/convert the counts matrix into a more interpretable *gene expression* matrix.

We first correct for *size factors* (differences in total UMI counts for each cell) by scaling the total number of counts in each cell to a constant factor (100,000 UMIs/cell in MiCV).

We then log-transform every value in the matrix, converting UMI counts to ln(1 + UMI counts). We now call this the *gene expression matrix* (seen in the figure here) which represents the expression levels of each gene in each cell, each set on equal footing with one another.

Identifying principal components (PCA)



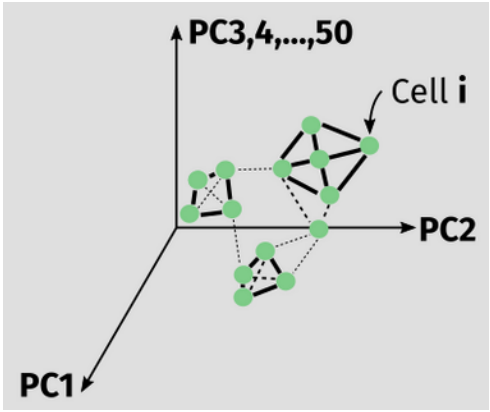
In MiCV, PCA is done in the background for you; this section is purely for your reference.

To begin to learn about the structure of this still high-dimensional gene expression dataset, we perform something called *principal component analysis (PCA)*. Similar to singular value decomposition, PCA attempts to identify sets of genes that, when grouped together, explain some of the variance in the data.

A practical example of this could be found in a single-cell dataset with an equal number of cells from male and female mice. Genes that play a role in regulating gene expression on the X-chromosome (such as the *Xist* RNA) could make up a principal component, as they describe a major amount of variation in the data.

Typically, PCA algorithms will generate 50 unique principal components to describe an scRNA-seq dataset. This effectively reduces the dimensionality of the data from 20000 to 50, while still accurately describing all of the nuances of individual cells' gene expression levels. The PCA representation of the data is used going forward in the analysis pipeline.

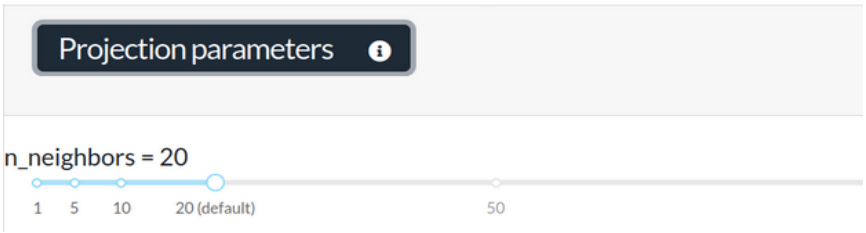
Neighborhood network/graph generation



Before clustering, we try to further understand the structure of the data by fitting a *neighborhood network* to the cells in the dataset. Network fitting algorithms take in a user's estimate of the number of similar cells (neighbors) any given cell might have, and use that information to connect similar cells in a network.

While a bit arcane, this network description of an scRNA-seq dataset makes identifying putative cell subtype clusters possible. It is used, along with the PCA representation of the data, in all analysis steps going forward.

In the [Projection parameters] dropdown on the [Preprocess] tab, MiCV exposes a slider bar for changing the expected number of neighbors (k) in your dataset. Users can rapidly test various values of k by selecting the [Recalculate projection] button at the bottom of the dropdown, without the need to repeat unnecessary steps from earlier in the pipeline.



Typically, k=20 (the default) is a good starting point, as it allows for relative small clusters to be identified while still identifying true large clusters in the data. However, datasets with particularly large cell counts (>20,000 cells), or very few expected cellular subtypes *might* benefit from larger values of k.