SCelVis

Sep 06, 2022

Tutorial

SCelVis is a web-based visualization tool for single-cell sequencing data.

CHAPTER 1

Getting Started

- you can find the **publically available web app** at https://scelvis-demo.bihealth.org.
- to get started with your analysis, watch the Movie or follow the *Analysis Tutorial*.
- to learn how to get your own data into SCelVis, read about *Source Format Details*.
- if you are a bioinformatician and interested in running your own SCelVis server, read the *Command Line Tutorial*.

CHAPTER 2

Running on your Computer

The recommended way to run SCelVis is using Docker:

```
$ docker run quay.io/biocontainers/scelvis:TAG scelvis --help
$ docker run -p 8050:8050 -v data:/data quay.io/biocontainers/scelvis:TAG scelvis run␣
↪--data-source /data
```

In the above, replace `TAG` with the latest version that you can find on the corresponding Quay.io project. For example, use `0.7.3--py_0`.

Alternatively, you can install it using `pip` ...

```
$ pip install scelvis
$ scelvis run --data-source ./data
```

... or using Bioconda:

```
$ conda install scelvis
$ scelvis run --data-source ./data
```

2.1 Analysis Tutorial

This tutorial describes the basics of performing the analysis of data using SCelVis. For this tutorial, we will use the public demo instance at https://scelvis-demo.bihealth.org.

Note: Data to be visualized can either be uploaded into a SCelVis server or it can be defined when the SCelVis server starts. When using a remote SCelVis server such as the public instance at scelvis-demo.bihealth.org, you will most likely upload your data as shown below. However, the server can also be started with the path or URL to the data. This way, computational staff can provide predefined datasets to non-computational staff. See *Command Line Tutorial* for more information.

First, download the file pbmc.h5ad, which is a published dataset of ~14000 IFN-beta treated and control PBMCs from 8 donors (GSE96583; see Kang et al.). For a simpler dataset, you could also use hgmm_1k.h5ad, containing data for 1000 cells from a 1:1 Mixture of Fresh Frozen Human (HEK293T) and Mouse (NIH3T3) Cells (10X v3 chemistry).

2.1.1 Upload Data

Then, use the menu on the top right to access the data upload screen *Go To –> Upload Data*.

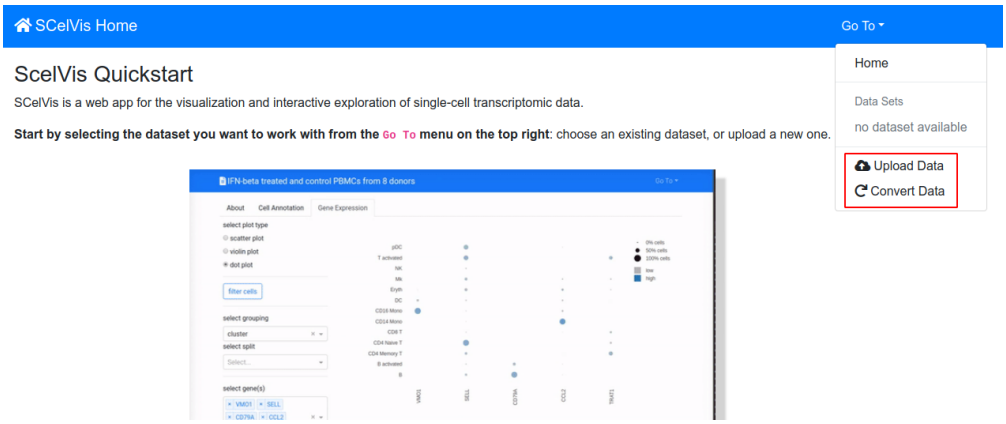

Fig. 1: Accessing the upload screen.

On the next screen click *Choose File*, select the *pbmc.h5ad* file, and click *Upload*. The file upload will take a while and return a link to the data analysis screen shown in the next section.

Note: Alternatively, you can use the example data set available from the top-right menu: *Go To –> Data Sets –> PBMC*.

2.1.2 Cell Annotation Analysis

In the beginning of each analysis, the cell annotation screen from the figure below is shown.

1. **Analysis selection tab.** This control allows to switch between

   About A textual description attached to the dataset.

   Cell Annotation The cell annotation analysis screen that you are looking at.

   Gene Expression The gene expression analysis screen.

2. **Plot type selection.** This allows to select the plot type for the cell-based analysis. Note that changing the plot type will change the subsequent controls.
3. **Cell filter button.** Using this control, you can filter cells based on various properties, see Section *Cell Filtering*.
4. **Axis and color control.** This allows you to select the dimensions to display along the horizontal and vertical axes as well as the colouring.
5. **Differential expression button.** This allows you to run a differential expression between two groups of cells, see Section *Differential Expression Analysis*.
6. **The cell scatter plot.** This is a dynamic scatter plot. Each dot corresponds to one cell, colored by what is selected from the *select coloring* list.

Fig. 2: The cell annotation scatter plot.

7. **Plot controls.** When you move your mouse cursor over the plot then some buttons will appear on the top right. These are explained in detail in Section *Plot Interface Controls* together with some other tricks.
8. **Download data for this plot.** Download a CSV file with the data for reproducing the plot outside of SCelVis.

Scatter Plot

Usually, you would choose embedding coordinates (e.g., tSNE_1 and tSNE_2) for *select x axis* and *select y axis* to create a standard tSNE or UMAP plot. *select coloring* allows to color cells by different cell annotation attributes, e.g., *cluster* for the cluster annotation or *n_counts* for the # of UMIs per cell. The available choices depend on how the dataset was preprocessed. Categorical variables will be shown with a discrete color scale, numerical variables with a gradient.

Alternatively, you could also plot, e.g., # of UMIs vs. # of genes for QC.

Violin and Box Plot

When selecting *violin plot* or *box plot* in the *select plot type* control you can draw violin or box plots. For example, selecting *n_counts* and *n_genes* in the *select variable(s)* and :*orig.ident* in *select coloring* will display the *n_genes* value distribution in the upper panel, and the *n_counts* value distribution in the lower panel for the individual samples of this dataset.

You can use the *select split* list to select whether you want to further split the grouping, e.g., by cluster identity. Hovering the mouse over the violin or box shapes will show you various statistical summaries of the distribution.

Bar Plots

With *bar plot*, you can display summary statistics, such as the number of cells per cluster by selecting *cluster* in *select grouping*. With *select split*, you can further investigate how clusters are populated in the different samples or the different donors. Checking *normalized* will switch from cell numbers to fractions, *stacked* will use stacked bars.

2.1.3 Gene Expression Analysis

When clicking the *Gene Expression* tab, you can investigate gene expression. As for the *Cell Annotation* analysis, it starts with the *scatter plot* type in **(1)**. The main difference is that all plot types will use the same list of genes selected in *select gene(s)* in **(4)**.

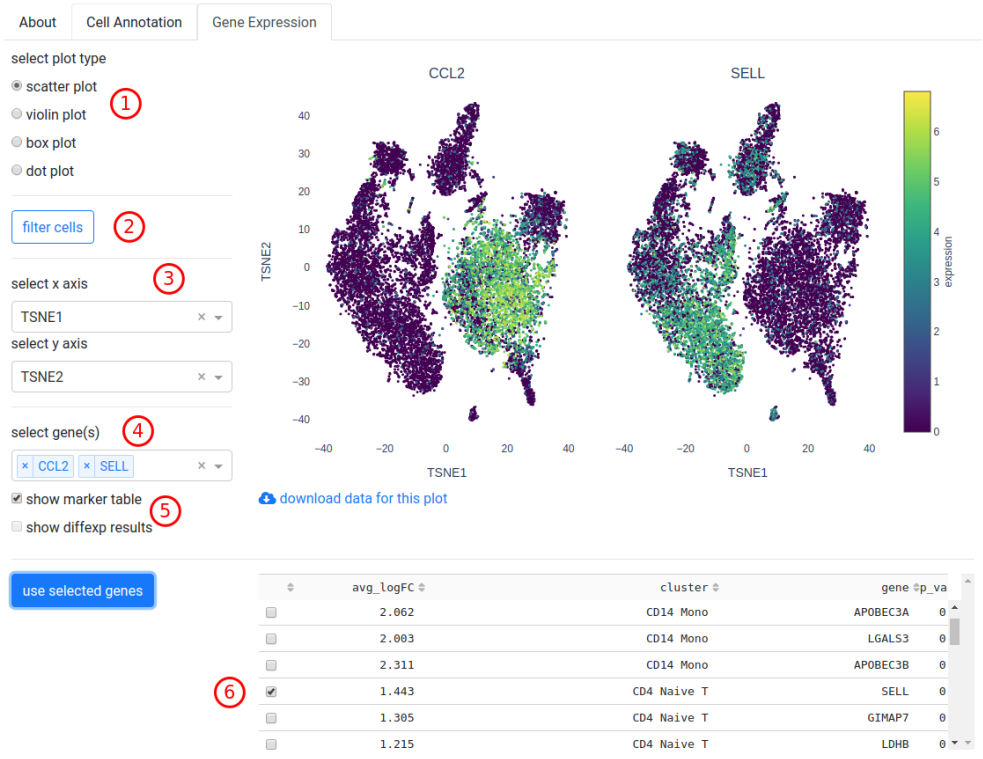

Fig. 3: The gene expression scatter plot for a monocyte (CCL2) and a T cell marker (SELL).

1. **Plot type selection.** This allows to select the plot type for the gene expression analysis. Note that changing the plot type will change the subsequent controls.
2. **Cell filter button.** Using this control, you can filter cells based on various properties, see Section *Cell Filtering*.
3. **Axis control.** This allows you to select the dimensions to display along the horizontal and vertical axes.
4. **Selecting genes.** Select one or multiple genes from this list or enter them by hand.
5. **Show tables.** Check these boxes to display tables with log2-fold change, p-values and other information for marker genes or differential expression results (if available).
6. **Table selection.** Genes can be selected from the table by checking the boxes to the left and clicking *use selected genes* to add them to the list.

Scatter Plot

Scatter plots for one or multiple genes will be shown in a grid, with expression values rescaled to the same range.

Violin or Box Plot

Violin and Box plots will show one gene per row, with one violin or box per category selected in *select grouping*, or multiple violins/boxes if *select split* is used.

Dot Plot

Dot plots summarise gene expression by size (fraction of expressing cells) and color (expression value), with genes in columns and groups (use *select grouping*) in rows. Dots can be subdivided using *select split*.

Plot Interface Controls

Fig. 4: A short demonstration of the plot controls in SCelVis.

The buttons on the top right of the plot (standard features of the Plotly library) are as follows:

Save plot as image The plot will be saved in PNG format.

Zoom After clicking this button, you can select a rectangular area to zoom into.

Pan Drag and drop the drawing area to move around in the plot.

Box Select, Lasso Select After clicking this button, you can select a rectangular area on the plot or draw a free from shape to select dots. This will be useful for differential expression analysis.

Zoom In / Zoom Out Zoom into or out of plot.

Autoscale / Reset Axes This will reset the scaling to the original area. You can obtain the same behaviour by double-clicking on a white spot in the plot.

Toggle Spike Lines Enable horizontal and vertical lines when hovering over a point.

Show Closest / Compare Data on Hover Change the spike lines behaviour.

Note that you can enable/disable individual groups by clicking their label in the legend. You can disable all but one group by double-clicking the label.

Cell Filtering

Fig. 5: A short demonstration of cell filtering in SCelVis.

The cell filtering works as follows:

1. Click the *filter cells* button to open the control panel.
2. Select a criterion by which cells should be filtered. Depending on how the data were preprocessed, the list will include cluster annotation, # of UMIs or genes per cell, etc. It is also possible to filter cells by expression of genes.
3. For categorical variables (e.g., cluster identity), checkboxes will appear and specific clusters can be (un)checked in order to include or exlude them from the analysis. For numerical variables (e.g., n_counts or gene expression), a range slider will appear: move the big circles inwards to remove cells outside the selected range.
4. Hit *update plot* to apply these filters to the current plot
5. Filters will be combined with AND logic; active filters are listed above the *update plot* button

6. Click *reset filters* to reset all filters and *update plot* to include all cells in the current plot
7. Note that current filter criteria will be applied to all subsequent plots of the current datasets, both in the *Cell Annotation* and the *Gene Expression* tabs

Differential Expression Analysis

Fig. 6: A short demonstration of differential expression analysis in SCelVis.

The differential expression analysis is available only when a scatter plot is displayed in the *Cell Annotation* tab. It works as follows:

1. Click the *differential expression* button, opening the controls panel.
2. Then, use either the box or lasso select tool of the plot for selecting cells in the scatter plot. For example, click the lasso select button in the top of the right of the plot. Move your mouse cursor to the position you want to start selecting at. Keep the left mouse button pressed and draw a shape around the cells that you are interested in. Release the mouse button. then Click *group A*.
3. Repeat step 2 but click *group B*.
4. Click *run* to perform the analysis.

The result could read something like *200 DE genes at 5% FDR* (a maximum of 200 genes will be displayed). You can click *view groups* to show the groups in the scatter plot, or click *table* to see the resulting DE genes in the *Gene Expression* tab table. You can also download the *results* or the *parameters* that were used for the DE gene analysis.

Clicking *reset* allows you to start a new DE gene analysis.

2.1.4 The End

This is the end of the data analysis tutorial. You might want to learn about the conversion of data into the HDF5 format next by reading Section *Conversion Tutorial*.

2.2 Conversion Tutorial

For visualization, data sets are provided as HDF5 files (anndata objects) that store gene expression (sparse CSR matrices) and meta data with very fast read access. You can use scanpy to create these HDF5 files directly or use the `scelvis convert` command for converting your single-cell pipeline output (see Section *Command Line Tutorial*).

2.2.1 Web File Converter

You can use the *Go To –> Convert Data* menu entry to access the file conversion screen.

Here, you can enter a title, short title and a description of your dataset, and upload a .zip or .tar.gz file containing the data with *Choose File*. Allowed formats are (see also *Source Format Details*)

- raw text (use this file as an example)
- CellRanger output (zip this directory as an example)
- loom

Hitting *Upload* will convert your data to HDF5 and take you to a screen where you can either directly view the converted dataset or download the resulting HDF5 file.

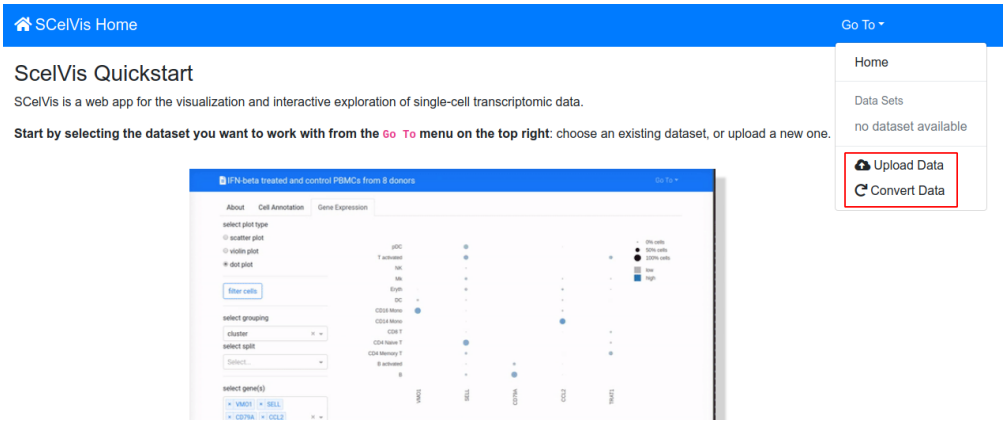

Fig. 7: Accessing the conversion screen.

2.3 Source Format Details

2.3.1 HDF5 Input

For HDF5 input (no conversion necessary), you can do your analysis with scanpy to create an anndata object `ad`. SCelVis will use embedding coordinates from `ad.obsm`, cell annotation from `ad.obs` and expression data directly from `ad.raw.X` (if present) or `ad.X` (this should contain normalized and log-transformed expression values for all genes and should be sparse, otherwise performance will suffer) If present, information about the dataset will be extracted from strings stored in `ad.uns['about_title']`, `ad.uns['about_short_title']` and `ad.uns['about_readme']` (assumed to be Markdown). Information about marker genes will be taken from entries starting with `marker_` in `ad.uns`: entries called `marker_gene` (required!), `marker_cluster`, `marker_padj`, `marker_LFC` will create a table with the columns `gene`, `cluster`, `padj`, and `LFC`. `SCelVis` will also extract marker information from `ad.uns['rank_genes_groups']`. However, certain datatypes in `ad.uns` together with version mismatches of `scanpy`, `h5py` and `anndata` can lead to `.h5ad` files that are not readable by `SCelVis` (see issue #832). To be on the safe side, it's recommended to delete unneccessary slots in `ad.uns` (e.g., `del ad.uns['rank_genes_groups']`). Also, `ad.obsm['X_pca']`, `ad.varm['PCs']` and entries in `ad.obsp` are likely dispensable.

If you prepared your data with `Seurat` (v2), you can use `Convert(from = sobj, to = "anndata", filename = "data.h5ad")` to get an HDF5 file.

Alternatively, you can use sceasy to convert your objects into `anndata` HDF5 format.

2.3.2 Text Input

For "raw" text input, you need to prepare a file with expression values, cell meta data and coordinates, and potentially information about this dataset as well as cluster markers.

- normalized expression values for each gene (rows) in each cell (columns) can be given either as tab-separated file (dense) or in matrix-market format:
  - if your file is called `expression.tsv.gz`, `SCelVis` expects a tab-separated file , e.g., like this:

```
.       cell_1   cell_2   cell_3   ...
gene_1  0.13     0.0      1.5      ...
gene_2  0.0      3.1      0.3      ...
gene_3  0.0      0.0      0.0      ...
```

  - if your file is called `expression.mtx`, `SCelVis` expects this to be a sparse matrix-market file and additional files called `barcodes.tsv` (containing a list of cell names / barcodes, one per line, no header or row names) and `genes.tsv` (containing a list of gene names, one per line, no header or row names) to be present.

- annotations for each cell can be provided in a tab-separated file called `annotation.tsv`, e.g., like this:

```
.          cluster     genotype  ...
cell_1     cluster_1   WT        ...
cell_2     cluster_2   KO        ...
```

- embedding coordinates for each cell can be provided in a tab-separated file called `coords.tsv`, e.g., like this:

```
.          tSNE_1   tSNE_2   UMAP_1  UMAP_2  ...
cell_1     20.53    -10.05   3.9     2.4     ...
cell_2     -5.34    13.94    -1.3    3.4     ...
```

- an optional tab-separated file called `markers.tsv` can contain information about marker genes. **it needs to have a column named "gene"**, e.g., like this:

```
gene    cluster     log2FC   adj_pval    ...
gene_1  cluster_1   3.4      1.5e-6      ...
gene_2  cluster_1   1.3      0.00004     ...
gene_3  cluster_2   2.1      5.3e-9      ...
```

- finally, a markdown file (e.g., `text_input.md`) can provide information about this dataset:

```
----
title: An Optional Long Data Set Title
short_title: optional short title
----

A verbose description of the data in Markdown format.
```

conversion to `.h5ad` is then performed like so:

```
$ scelvis convert --input-dir text_input --output data/text_input.h5ad --about-md␣
↪text_input.md
```

in `examples/dummy_raw.zip` and `examples/dummy_about.md` we provide raw data for a simulated dummy dataset.

if you prepared you data with `Seurat`, you can export to raw text like this

```
writeMM(sobj@assays$RNA@data, file = 'expression.mtx')
write.table(Cells(sobj), file = 'barcodes.tsv', col.names = FALSE, row.names = FALSE,␣
↪sep = ',')
write.table(row.names(sobj@assays$RNA@data), file = 'genes.tsv', col.names = FALSE,␣
↪row.names = FALSE, sep = ',')
sobj@meta.data$cluster <- paste0('cluster_', sobj@meta.data$seurat_clusters)
write.table(sobj@meta.data, file = 'annotation.tsv', sep = '\t')
write.table(sobj@reductions$umap@cell.embeddings, file = 'coords.tsv', sep = '\t')
```

2.3.3 Loom Input

for loompy or loomR input, you can convert your data like this:

```
$ scelvis convert --i input.loom -m markers.tsv -a about.md -o loom_input.h5ad
```

if you prepared your data with `Seurat` (v3), you can use `as.loom(sobj, filename = "output.loom")` to get a `.loom` file and then convert to `.h5ad` with the above command (this is quite slow, however, and exact format specifications for `.loom` and `.h5ad` are not always compatible between versions)

2.3.4 CellRanger Input

Alternatively, the output directory of `CellRanger` can be used. This is the directory called `outs` containing either a file called `filtered_gene_bc_matrices_h5.h5` (version 2) or a file called `filtered_feature_bc_matrix.h5` (version 3), and a folder `analysis` with clustering, embedding and differential expression results. This will not no any further processing except log-normalization. Additionally, a markdown file provides meta information about the dataset (see above)

```
$ mkdir -p data
$ cat <<EOF > data/cellranger.md
---
title: My Project
short_title: my_project
---

This is my project data.
EOF
$ scelvis convert --input-dir cellranger-out --output data/cellranger_input.h5ad --
↪about-md cellranger.md
```

In `examples/hgmm_1k_raw` we provide `CellRanger` output for the 1k 1:1 human mouse mix. Specifically, from the `outs` folder we selected

- `filtered_feature_bc_matrix.h5`
- tSNE and PCA projections from `analysis/tsne` and `analysis/pca`
- clustering from `analysis/clustering/graphclust` and
- markers from `analysis/diffexp/graphclust`

`examples/hgmm_1k_about.md` contains information about this dataset.

2.4 Command Line Tutorial

This tutorial explains the installation of SCelVis on your own computer.

2.4.1 Installation

A Docker container is also available via Quay.io/Biocontainers.

```
$ docker run quay.io/biocontainers/scelvis:TAG scelvis --help
$ docker run -p 8050:8050 -v data:/data quay.io/biocontainers/scelvis:TAG scelvis run␣
↪--data-source /data
```

look up the latest `TAG` to use at here, e.g.,

```
$ docker run quay.io/biocontainers/scelvis:0.7.3--py_0 scelvis --help
$ docker run -p 8050:8050 -v data:/data quay.io/biocontainers/scelvis:0.7.0--py_0␣
↪scelvis run --data-source /data
```

For installation, the only prerequisite is Python 3, everything else will be installed together with the `scelvis` package. You can install SCelVis and its dependencies using `pip` or through `conda`:

```
$ pip install scelvis
# OR
$ conda install scelvis
```

The most robust way is to use Docker, though.

For the sake of simplicity, we will give the executable as `scelvis`. When using Docker, use the corresponding prefix.

2.4.2 Running SCelVis

You can run the SCelVis Web server with `scelvis run`.

```
$ scelvis run --data-source /path/to/scelvis/examples/hgmm_1k.h5ad
$ scelvis run --data-source https://files.figshare.com/18037739/pbmc.h5ad
```

and then point your browser to http://0.0.0.0:8050/ or http://localhost:8050/.

We provide the following two example HDF5 files:

- hgmm_1k.h5ad
- pbmc.h5ad

The first command will make SCelVis directly serve the given `hgmm_1k.h5ad` file while the second command will first download the file `pbmc.h5ad` from the given URL and then complete web server startup. The files given as `--data-source` on server startup will be displayed in the top right *Go To* menu.

2.4.3 Data Sources

Data sources can be:

paths e.g., `relative/paths` or `/absolute/paths` or `file://url/paths`

HTTP(S) URLs e.g., `https://user:password@host/path/to/data`.

S3 URLs `s3://bucket/path`, optionally `s3://key:token@bucket/path`.

SFTP URLs e.g., `sftp://user:password@host/path/to/data`

FTP URLs e.g., `ftp://user:password@host/path/to/data` (sadly encryption is not supported by the underlying library PyFilesystem2.

iRODS URLS e.g., `irods://user:password@host/zoneName/path/to/data`

- Enable SSL via `irods+ssl`
- Switch to PAM authentication with `irods+pam` (you can combine this with `+ssl` in any order)
- Enable ticket access by appending `?ticket=TICKET`.

Data sources can either point to HDF5 files directly or to directories containing multiple HDF5 files.

2.4.4 Configuration

You can configure the web server by passing command line arguments (run `scelvis run --help` for all available options). Also, you can use the following environment variables:

2.4.5 Environment Variables

You can use the following environment variables to configure the server.

SCELVIS_DATA_SOURCES semicolon-separated list of data sources

SCELVIS_HOST host specification for web server to listen on

SCELVIS_PORT port for web server to listen on

SCELVIS_CACHE_DIR directory to use for the cache (default is to create a temporary directory)

SCELVIS_CACHE_REDIS_URL enable caching with REDIS and provide connection URL

SCELVIS_CACHE_DEFAULT_TIMEOUT cache lifetime coverage

SCELVIS_CACHE_PRELOAD_DATA will preload all data at startup

SCELVIS_UPLOAD_DIR the directory to store uploaded data sets in (default is to create a temporary directory)

SCELVIS_UPLOAD_DISABLED set to "0" to disable upload feature

SCELVIS_CONVERSION_DISABLED set to "0" to disable the conversion feature

SCELVIS_URL_PREFIX set if you want to run scelvis below a non-root path (e.g., behind a reverse proxy)

2.5 Developer Setup

The prerequisites are:

- Python 3, either
  - system-wide installation with `virtualenv`, or
  - installed with Conda.
- Git LFS must be installed for obtaining the example data files.

For `virtualenv`, first create a virtual environment and activate it.

```
$ virtualenv -p venv
$ source venv/bin/activate
```

For a Conda-based setup create a new environment and activate it.

```
$ conda create -y -n scelvis 'python>=3.6'
$ conda activate scelvis
```

Next, clone the repository and install the software as editable (`-e`). Also install the development requirements to get helpers such as black. (Note that you must have Git LFS installed to actually obtain the data files).

```
$ git clone git@github.com:bihealth/scelvis.git
$ cd scelvis
$ pip install -e .
$ pip install -r requirements/develop.txt
```

Afterwards, you can run the visualization web server as follows:

```
$ scelvis run --data-source path/to/data/dir
```

To explore the datasets provided in the git repository, use `git lfs fetch` to download

2.6 Releasing Packages

For the PyPi package:

```
$ python setup.py sdist
$ twine upload --repository-url https://test.pypi.org/legacy/ dist/scelvis-*.tar.gz
$ twine upload dist/scelvis-*.tar.gz
```

For the Bioconda package, see the great documentation. The Docker image will automatically be created as a BioContainer when the Bioconda package is built.

2.7 Indices and tables

- genindex
- search