**Title:** Single Cell Explorer, collaboration-driven tools to leverage large-scale single cell RNA-seq data

### Di Feng1, Charles E. Whitehurst2, Dechao Shan1, Yonggang Yue1,3

1. Computational Biology, Boehringer Ingelheim Pharmaceuticals, 900 Ridgebury Road, Ridgefield, CT, 06877, USA.

2. Immunology and Respiratory Disease Research, Boehringer Ingelheim Pharmaceuticals, 900 Ridgebury Road, Ridgefield, CT, 06877, USA.

3. Current address: Inzen Therapeutics, 430 East 29th St, New York, NY 10016, USA.

Correspondence to

Di Feng, Email: di.feng@boehringer-ingelheim.com

### Abstract

Background

Single cell transcriptome sequencing has become an increasingly applied technology to dissect complex biology at a resolution impossible with bulk sequencing. However, bridging the gap between the technical expertise required to effectively work with the resultant high dimensional data and the biological expertise required to interpret the results in their biological context remains incompletely addressed by the currently available tools.

Results

Single Cell Explorer is a Python-based web server application we developed to enable computational and experimental scientists to iteratively and collaboratively annotate cell expression phenotypes within a user-friendly and visually appealing platform. These annotations can be modified and shared by multiple users to allow easy collaboration between computational scientists and experimental biologists. Data processing and analytic workflows can be easily integrated into the system using Jupyter notebooks. The application enables powerful yet accessible features such as the easy identification of differential gene expression patterns for user-defined cell populations and convenient annotation of cell types using marker genes or differential gene expression patterns. As such, by making single cell RNAseq data sharing and querying more user-friendly, the software promotes deeper understanding and innovation by research teams applying single cell transcriptome approaches.

Conclusions

Single cell explorer is a freely-available tool developed for single cell transcriptomic analysis which enables computational and experimental biologists to collaboratively explore, annotate, and share results in a flexible software environment which intuitively promotes innovation.

### ****Keywords:**** Single Cell, RNA-Seq, Pipeline, Transcriptomics,Visualization, Django, D3, Python

### Background

The rapidly evolving single cell sequencing technologies are enabling researchers to generate data that has the potential to lead to unprecedented biological insight, at the cost of greater complexity of analysis. Open-source, point-and-click, web-based interfaces have become a popular choice to share the analytic results of single cell experiments (Hay, Ferchen, Chetal, Grimes, & Salomonis, 2018). More authors now provide Rshiny apps as a solution to share results from specific studies or collections. Other software such as iS-CellR (Patel, 2018) and ASAP (Gardeux, David, Shajkofci, Schwalie, & Deplancke, 2017) provide graphic interface for non R programmers to use specific R packages such as Seurat(Butler, Hoffman, Smibert, Papalexi, & Satija, 2018). However, because of a continuous increase in the creation of experiment types, pipelines and methods it may be considered impossible to generate a single graphic user interface (GUI) that covers a large number of methods without impairing usability. Many present tools are specialized ‘build to fit’ applications which focus on data exploration of processed data, but do not permitduplication of research findings from raw data. Furthermore, these often are constrained as data exploration tools, rather than being sufficiently full-featured to allow open-ended analysis. We developed Single Cell Explorer using hybrid approaches including the application of a Python based programming environment and web app GUI to enable result sharing and fluid data exploration. The Python based programing environment is chosen for enhancement of data reproducibility and flexible implementation of a variety of algorithms/workflows since the integration of Jupyter notebook is becoming increasingly popular in the bioinformatics research community (Torre, Lachmann, & Ma'ayan, 2018). Single Cell Explorer’s GUI was developed with a focus on easy use and intuitiveness for experimental biologists to explore with minimal training. Single Cell Explorer was developed as a generalized platform for research teams to share and use single cell transcriptome data generated from either pipelines or processed data, with full open access of complex workflow, tools, and with methodology all behind a simple interface. In contrast to the existing R-based frameworks, Single Cell Explorer will scale to large and large collections of studies by integrating with modern, performant databases and workflows such as as Scanpy(Wolf, Angerer, & Theis, 2018).

### Implementation of Single Cell Explorer

Single Cell Explorer is written using the Python 3.0 programming language, and is built with the Django framework. User interactions such as drawing and labelling are written using Javascript. The software is open source and currently available through GitHub at [https://github.com/d-feng/SingleCellExplorer](https://github.com/d-feng/scExplorer). It can be launched by servers which support the Python environment. Python WSGI HTTP Servers for UNIX such as Gunicorn are suggested to support concurrent use of this app. The steps to use the application are:

1. Raw Data Processing. Initial processing of data is performed using Python Jupyter Notebook or JupyterLab. This step includes reading alignment, gene quantitation, and quality control employing Cell Ranger v3.0 to process Chromium single-cell RNA-seq FASTQ data. Alternatively, raw data can be processed using Bash or Nextflow. We provided the Bash script sce.sh to integrate the raw data processing pipeline Cell Ranger,. The required input files are FASTQ files, as well as the appropriate genome reference files for the relevant organism.
2. Preliminary Analysis in Python Environment. This step runs quality control and dimensionality reduction using the results generated from step 1. The application is agnostic to the method used for dimensionality reduction; both t-SNE or UMAP coordinates have been generated with Seurat or Scanpy methods and used. The principal output of this step includes the filtered cell / gene expression matrix as well as the matrix describing the 2D coordinates of the cells in lower dimensional space. The output is then loaded into a MongoDB database, along with basic metadata about the project to enable project-level queries.
3. Collaborative Analysis through Web and API. After the data has been loaded, the web front end enables users to visualize and query downstream analytic results through interaction with the lower-dimensional map of the cells. This step relies on Javascript, SVG, HTML5, and CSS (cascading style sheet) to enable an interface which is highly responsive and scales well. In addition to basic data exploration, cell type annotations can be captured by users and stored. For highly customized analyses, API functions enable informaticians to work directly with the database..

### Results and Discussion

Single Cell RNA-seq Data Processing and Analysis

As an example of the utility of Single Cell Explorer, a test run was performed on a publicl- available dataset of human PBMCs (pbmc\_10k\_v3). We showed the case of using a Jupyter notebook to drive a 10X genomics based cell processing pipeline. To enable this, a JSON–format file was provided, which included parameters such as FASTQ data location and cell ranger output location, sample and study information, parameters for QC thresholds, and project information for database and web app display. The Cell Ranger pipeline can be started using a function in the notebook, and is then followed with the Scanpy analytic workflow in the same notebook for quality control and dimensionality reduction. The project metadata, cell / gene expression matrix and normalized data as well as results of the 2D cell mapping will be uploaded by the notebook to the MongoDB instance.

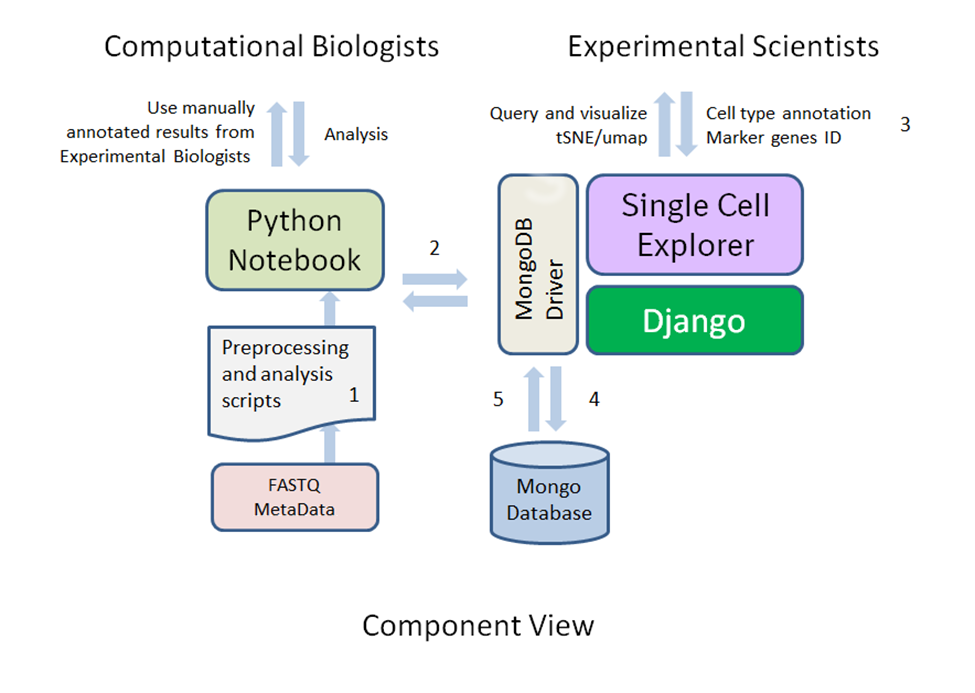


Figure 1: Single Cell Explorer workflow architecture process and component view.

Overview of the data process workflow steps for Single Cell Explorer. Step #1: Running pipeline to process FASTQ files using Python wrapper through Jupyter Notebook or NextFlow. Step #2: Quality control of data, generation of 2d representation, and upload to database. Step #3: Interactive data analyses and annotation of cell types. Step #4: Annotated results recorded in MongoDB and shared with all users. Step #5: All results from MongoDB can be accessed directly or via API.

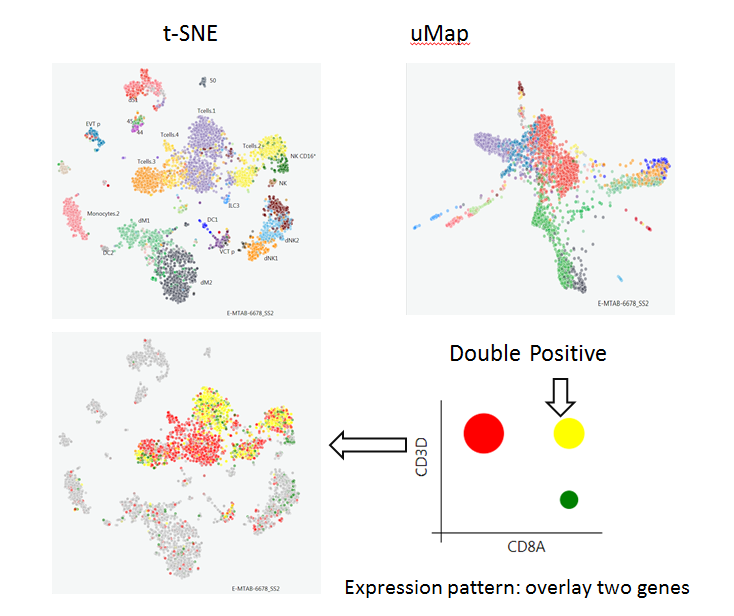


Figure 2: Interactive FeaturePlot

A t-SNE and UMAP representation from first-trimester placentas with matched maternal blood and decidual cells. Individual pre-labeled cell types are painted in different colors. The function of painting two genes (CD8A and CD3D) highlights the location of CD8 T cell clusters. A 2D plot of circles indicates the proportion of the single positive and double positive cells across cells in the data set.

Interactive tertiary results access from web page:

For high-dimensional single cell data, lower dimensional representations such as t-SNE or UMAP are necessary to interact with the data and to easily observe broad relationship between cells (Figure 2). Single Cell Explorer supports all types of low-dimensional representation (Wang & Gu, 2018). Here we showed the re-analysis of single cell RNaseq data for cells from the early human maternal-fetal interface (Vento-Tormo et al., 2018). The multiple types of metadata including cell types, cluster information,and sample information such as tissue, donor, as well as any other clinical features can be overlaid on the feature plot. The user interface provides a simple gene expression search function for each feature plot. A box plot of normalized counts and percentage of cells with positive expression (counts>1) will be shown for querying single gene expression. The interface also supports queries for two genes simultaneously with the gene expression pattern painted with different colors.

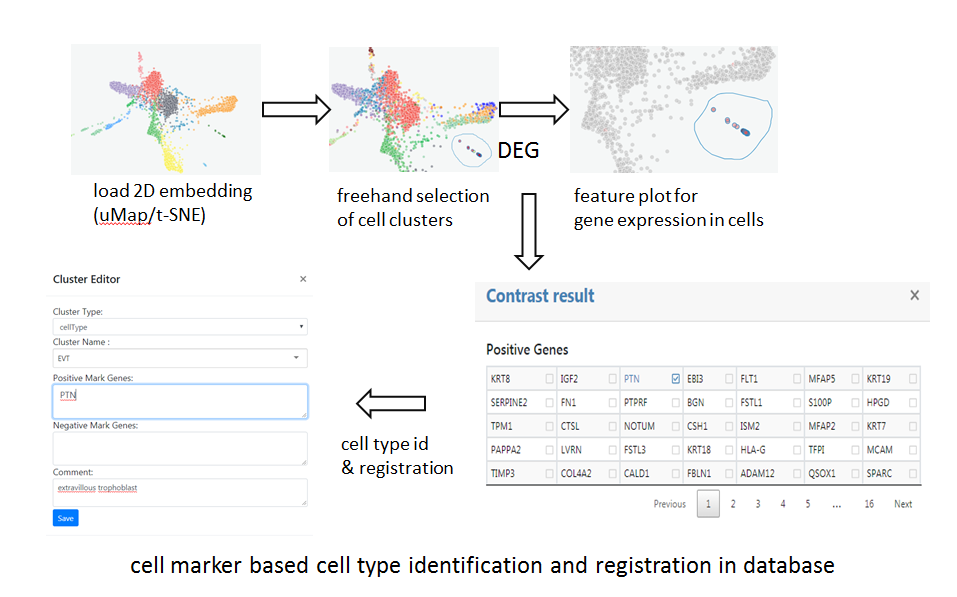


Figure 3: Cell type and feature discovery. Step #1: Use a free hand tool to select the cells of interest. Step #2: Compute the differentially expressed genes between selected cells with all unselected cells providing the background levels. Step #3: Interactively visualize gene expression levels using the result table. Step #4: Record cell types and marker genes for future reference. Step #5: Position the newly-labelled cells in the map.

Cell typing identification and annotation

The UI was designed for users to use cell markers or domain knowledge to annotate cells clusters in 2D map. First, the user can click and draw circles to select the cell cluster of interest. Next, a contrast function is executed for computing differentially-expressed genes between selected clusters and all other cells. The non-parametric Wilcoxon rank sum test is chosen as the default method due its the fast execution time and comparable performance among other algorithms (Soneson & Robinson, 2018). The computed results will be shown as a table and include p-value-ranked increased genes and decreased genes. The user can click the differentially-expressed genes which will be distinguished by their color on the t-SNE plot. The user can name the cell type by choosing cell type name from a list (to enforce controlled vocabulary) or add new names that do not exist in the database.

Database and API

The annotated data will be used for displaying in the web application. The following Python API functions were designed to retrieve data from single cell explorer database. The map id is unique for each map. clusterName is the annotated cell type. clusterType, includes cell type or other information such as donor, samples, and shared nearest neighbor’s cluster id.

|  |  |
| --- | --- |
| name | function |
| getAllClstrsByClstrsType | retrieve table of cell barcodes and annotated cell type |
| getNormalizedGeneExpr | get normalized counts matrix for genes of interest from specific cell types in specific map |
| getAllNormalizedGeneExpr | Get normalized gene counts matrix from specific cell types in specific map |
| getMarkGenesByMapidAndClusterType | get annotated marker genes |
|  |  |

Comparison to other software

To our best knowledge, no other single cell sequencing software currently provides reanalysis capabilities that include drawing, annotation, saving the results in a database, and integration with Jupyter notebook for more complex analysis. Cellxgene(Initiative) from <https://www.allencell.org/genomics.html> is a Python-based interactive data visualization tool single-cell transcriptomics datasets, but focuses on single data sets without comprehensive database support. It will show data objects but needs a single instance or independent port for each data object in the datasets. Our application is built for concurrent users to explore unlimited number of datasets. In contrast to canvas that is suitable for displaying large number of cells, we also use SVG to allow faster information accessibility and better interactive performance for data sets with fewer cells.

### Conclusion:

We developed Single Cell Explorer, a Python-based platform which promotes and encourages a collaborative data sharing experience for single cell transcriptomic data. It balances a high degree of automation integration with open source tool sets and a visually-attractive end user experience. For a genomics core lab, a complete workflow analysis with a high degree of automation allows experimental scientists to easily preview their results, quickly promoting faster cycles of hypotheses building and experimental innovation. Computational biologists can also analyze data sets using different methods to generate 2D plots of findings to load and share with research teams. Using the web app, team members can label and leave comments to share findings and promote further inquiry and innovation.

### Availability and requirements

**Project name:** Single Cell Explorer

**Project home page:**

Repository. <https://github.com/d-feng/SingleCellExplorer>

Demo software**:**<http://54.159.6.229:8000/>

Python notebook: <http://54.159.6.229:8001/>

**Operating systems:**Server: Linux, Client: Platform independent  
**Programming language:**Python, Javascripts  
**Other requirements:**Python3.6, Mongodb 3.6, Django 2.0 or above, scanpy, scipy, numpy, pymongo, scipy, pandas, numpy, subprocess, sklearn, bootstrap 4, jquery 3, d3.v4.js, optional softwares to enhance the function: cellranger 3.0, jupyter notebook, gunicorn

**License:** GNU GPL version 3

**Any restrictions to use by non-academics:** none.

## Declarations

Competing interests: DF and CW are employees of Boehringer-Ingleheim. DS is Contingent Worker providing services to Boehringer-Ingelheim. All authors declare that they have no competing interests.

Funding: Not applicable.

Author’s contributions: DF conceived, designed, implemented, tested, validated the software, and drafts the manuscript. DS implemented and tested and software and designed database. YY conceived, designed, and tested software. CW participates in the design, test, and feedback of requirement specification. All authors read and approved the manuscript.

License: Contributor Information  
Di Feng, Email: [di.feng@boehringer-ingleheim.com](mailto:di.feng@boehringer-ingleheim.com) Charles Whitehurst, Email: [charles.whitehurst@boehringer-ingelheim.com](mailto:charles.whitehurst@boehringer-ingelheim.com) Dechao Shan, Email: [dechao.shan.ext@boehringer-ingelheim.com](mailto:dechao.shan.ext@boehringer-ingelheim.com) e Yonggang Yue, Email: [ygyue@yahoo.com](mailto:ygyue@yahoo.com)

Acknowledgements: Not applicable.

**References**

Butler, A., Hoffman, P., Smibert, P., Papalexi, E., & Satija, R. (2018). Integrating single-cell transcriptomic data across different conditions, technologies, and species. *Nat Biotechnol, 36*(5), 411-420. doi:10.1038/nbt.4096

Gardeux, V., David, F. P. A., Shajkofci, A., Schwalie, P. C., & Deplancke, B. (2017). ASAP: a web-based platform for the analysis and interactive visualization of single-cell RNA-seq data. *Bioinformatics, 33*(19), 3123-3125. doi:10.1093/bioinformatics/btx337

Hay, S. B., Ferchen, K., Chetal, K., Grimes, H. L., & Salomonis, N. (2018). The Human Cell Atlas bone marrow single-cell interactive web portal. *Exp Hematol, 68*, 51-61. doi:10.1016/j.exphem.2018.09.004

Initiative, C. Z. cellxgene. Retrieved from <https://github.com/chanzuckerberg/cellxgene>

Patel, M. V. (2018). iS-CellR: a user-friendly tool for analyzing and visualizing single-cell RNA sequencing data. *Bioinformatics, 34*(24), 4305-4306. doi:10.1093/bioinformatics/bty517

Soneson, C., & Robinson, M. D. (2018). Bias, robustness and scalability in single-cell differential expression analysis. *Nat Methods, 15*(4), 255-261. doi:10.1038/nmeth.4612

Torre, D., Lachmann, A., & Ma'ayan, A. (2018). BioJupies: Automated Generation of Interactive Notebooks for RNA-Seq Data Analysis in the Cloud. *Cell Syst, 7*(5), 556-561 e553. doi:10.1016/j.cels.2018.10.007

Vento-Tormo, R., Efremova, M., Botting, R. A., Turco, M. Y., Vento-Tormo, M., Meyer, K. B., . . . Teichmann, S. A. (2018). Single-cell reconstruction of the early maternal-fetal interface in humans. *Nature, 563*(7731), 347-353. doi:10.1038/s41586-018-0698-6

Wang, D., & Gu, J. (2018). VASC: Dimension Reduction and Visualization of Single-cell RNA-seq Data by Deep Variational Autoencoder. *Genomics Proteomics Bioinformatics, 16*(5), 320-331. doi:10.1016/j.gpb.2018.08.003

Wolf, F. A., Angerer, P., & Theis, F. J. (2018). SCANPY: large-scale single-cell gene expression data analysis. *Genome Biol, 19*(1), 15. doi:10.1186/s13059-017-1382-0

### Availability of data and materials

Operation System: Linux

Languages: Python, Javascripts

Dependencies: Python3.6, Mongodb 3.6, Django 2.0 or above, scipy, numpy, pymongo, scipy, pandas, numpy, scanpy, subprocess, sklearn, bootstrap 4, jquery 3, d3.v4.js

Source Codes, Notebook scripts, Example data, and Manual will be deposited in

<https://github.com/d-feng/SingleCellExplorer>