SCOUTS Documentation

Release 2019

Juliano Faccioni

Mar 28, 2020

Contents

CHAPTER 1

Single Cell Outlier Selector

SCOUTS is a tool that quickly finds **OUTLIERS in single-cell datasets**, especially those obtained by **mass cytometry (CyToF)** and **single-cell RNA sequencing** (scRNA-Seq) platforms.

Developed by LabSinal.

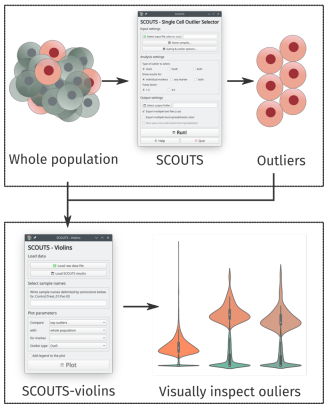
CHAPTER 2

Usage & documentation

2.1 Why SCOUTS?

Many single-cell analytical pipelines require some level of programming knowledge in order to be used. While some great tools for languages like R, Python and Julia have been developed, the entry-level barrier of programming is still intimidating for many scientists starting on the field of single-cell analysis. With this in mind, we developed SCOUTS to simplify this process. Through a desktop application, the user is able to choose the parameters for the outlier selection, and leave the hard work of programmatically subsetting the data to SCOUTS.

As a showcase of how to interpret and explore the data generated by SCOUTS, we also developed **SCOUTS-violins**, a secondary desktop application which displays the outlier populations identified by SCOUTS as violin plots.



2.2 Installation

SCOUTS is available as a:

- Python package from PyPI - install with `pip`
- Conda package - install with `conda` (coming soon!)

- GitHub repository - download/clone the repository
- binary release (experimental)

For any installation option (other than the binary release), SCOUTS requires **Python >= 3.6** to installed in your system. To check this, open a terminal/cmd and type:

```
$ python --version
```

If the output is something like `Python 3.6`, you're golden!

If the output is something like `Python 2.7`, try again with:

```
$ python3 --version
```

If your system can't locate Python3, you may need to download or upgrade Python.

For PyPI/pip or GitHub installations, we recommend using a virtual environment, instead of installing SCOUTS into your global/system-wide Python interpreter.

2.2.1 Install from PyPI using `pip`

To install SCOUTS, simply type in a terminal:

```
$ pip install scouts
```

then run SCOUTS by entering `scouts` in your terminal.

If you run into problems, check the troubleshooting section.

Optional: SCOUTS-violins

If you also want to install SCOUTS-violins, type:

```
$ pip install scouts[violins]
```

then run SCOUTS-violins by entering `scouts-violins` in your terminal.

If you run into problems, check the troubleshooting section.

2.2.2 Install using `conda`

If you use conda, you can install SCOUTS by simply typing in a terminal (preferably from a previously created conda environment):

```
$ conda install -c jfaccioni scouts
```

Note that this option installs both `scouts` and `scouts-violins`.

2.2.3 Install from the GitHub repository

Download this repository into your local machine. Alternatively, clone it with `git`:

```
$ git clone https://github.com/jfaccioni/scouts
```

Enter the repository's directory:

```
$ cd scouts
```

Make sure your Python interpreter (version >= 3.6) has the following packages installed:

- numpy
- pandas
- pyside2
- openpyxl
- xlrd

Run SCOUTS by typing:

```
$ python scouts.py
```

Optional: SCOUTS-violins

If you also want to install SCOUTS-violins, make sure your Python interpreter (version >= 3.6) has the following additional packages installed:

- matplotlib
- seaborn

Run SCOUTS-violin by typing:

```
python scouts-violins.py
```

Using `pipenv` with the GitHub repository

For `pipenv` users, we included a `Pipfile` for convenience. Simply type `pipenv install` from the repository's directory to install SCOUTS into a virtual environment, along with the necessary third-party dependencies. This covers the installation of both SCOUTS and SCOUTS-violins.

2.2.4 Download the binary executable

Download the binary release for your OS here (Mac support is coming soon).

If you choose this option, please be aware that:

- SCOUTS is a Python package. The executable has a moderately large size (~150 mb), since it has to bundle the whole Python interpreter along with it.
- this is an **experimental release of SCOUTS**, and as such it is not expected to support all OS configurations. If you run into any problems, please choose another installation method.

2.2.5 Troubleshooting

These are some common errors you may run into when trying to install SCOUTS:

Windows: `pip is not recognized as an internal or external command, operable program or batch file`

If you have Python >= 3.6 installed but see the message above, you need to add Python to your Windows PATH.

All platforms: `ERROR: Could not install packages due to an EnvironmentError`

This error happens if you have installed Python to your system (with admin privileges) but run the terminal/cmd as a user (without admin privileges). You have a couple of options to circumvent this:

1) Use a virtual environment. This creates an isolated Python interpreter to your user and avoids the `pip` packages from interfering with your system's Python.
2) Run the terminal/cmd as admin:
   - On Windows, search for `cmd` on the search bar, right-click and select "run as administrator"
   - On Mac/Linux, use the command `sudo pip install scouts`
3) Install `scouts` to your user. Add the `--user` flag to pip (e.g.''pip install –user scouts''). Note that SCOUTS may be downloaded to a folder not in PATH, so your system won't be able to automatically locate it (see below).
4) Reinstall Python inside your User folder.

All platforms: `SyntaxError: invalid syntax`

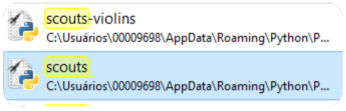
You are probably trying to run `pip` from within the Python interpreter. Exit the Python interpreter with `exit()` and use `pip` from your system shell/command line.

Windows: `scouts is not recognized as an internal or external command, operable program or batch file`

Make sure that `pip install scouts` has successfully installed `scouts`.

If you still see this message, the folder containing `scouts` is probably not in your PATH (likely due to conflicts between where you installed Python and where `pip` installed `scouts`). You can either:

1) Manually run SCOUTS by searching for `scouts` on the Windows Explorer search bar, and running the application (as per the image below):



2) Add the folder containing the files found in step 1 to Windows PATH

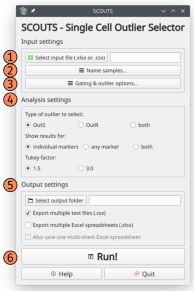
2.3 Basic usage

The `scouts` package include two user interfaces:

- **SCOUTS**: used to select outliers in a population of single-cells.
- **SCOUTS-violins** (optional): used to visually inspect outliers selected by SCOUTS.

2.3.1 Using SCOUTS

This is a basic rundown of the interface when you start SCOUTS:



1) Choose your input data
2) Choose your sample names - explanation here
3) Choose whether to gate samples and analyse other outlier populations - explanation here
4) Choose how to perform outlier selection - explanation here
5) Choose how and where to save output
6) Run the program

2.3.2 Using SCOUTS-violins

This is a basic rundown of the interface when you start SCOUTS-violins:

1) Choose your input data (same input file used for SCOUTS + SCOUTS output folder)
2) Choose your sample names - explanation here
3) Choose plot parameters
4) Plot your data

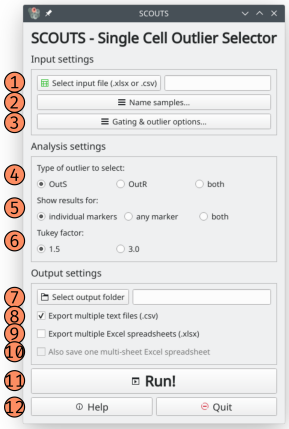
2.4 How SCOUTS works

2.4.1 Interface elements

This section explains what every button and option of the SCOUTS interface does.

Main window

These are the elements of the main window:



1) Input file: select your input file by clicking on this button. Valid input file formats are Excel spreadsheets (.xlsx) and comma-separated values (.csv). Please be sure that the input file is properly formatted.

2) Name Samples: this opens the sample names window.

3) Gating & outlier options: this opens the gating window.

4) Type of outlier to select: here you can select which cutoff value to consider when selecting outliers.

- **Outliers by sample (OutS)**: SCOUTS will calculate the cutoff value for each sample independently. In other words, the point from which a given cell is considered outlier will be different in different samples, since each sample has its own distribution of values.
- **Outliers by reference (OutR)**: SCOUTS will calculate the cutoff value for an user-defined reference sample, and use this value when selecting outliers for the remaining samples. This is useful to compare outliers from different samples using the same baseline for comparisons (e.g. "how many outliers do the treatment samples have, compared to the control sample and using the same cutoff point?").
- **both**: SCOUTS performs both analyses for OutS and OutR separately.

5) Show results for: here you can select how SCOUTS considers when a cell is an outlier or not.

- **Individual markers**: SCOUTS will find outliers for each marker present in the input file. Each marker generates its own output file, where only cells that are outliers *for that specific marker* are present
- **Any marker**: SCOUTS will find outliers for *at least one* marker present in the input file. The output file contains all cells that are considered outliers for at least one of the markers.
- **both**: SCOUTS performs both analyses for individual markers and any markers separately.

6) Tukey factor: choose whether to consider 1.5 or 3.0 as the Tukey factor for calculating outliers. Conceptually, a Tukey factor of 1.5 selects possible outliers, while a Tukey factor of 3.0 selects probable outliers.

7) Select output folder: the base folder in which to save all results from SCOUTS. We recommend creating a new folder, in order to keep files organized. Be aware that saving **multiple SCOUTS analyses to the same output folder will likely result in your data being overwritten**, so use different folders for different runs! Here is a detailed description of the files that SCOUTS outputs.

8) Export csv: this option generates comma-separated values (.csv) files for the outliers for each marker/sample combination.

9) Export Excel: this option generates Excel spreadsheet (.xlsx) files for the outliers for each marker/sample combination.

10) Generate multi-sheet Excel: this option generates one large Excel workbook, where all Excel spreadsheets generated are joined in the same file. This option is included for users that want to have all analyses on a single file. **Be aware that this option consumes a lot of RAM, and may slow your computer down**, so be sure that your machine can handle it!

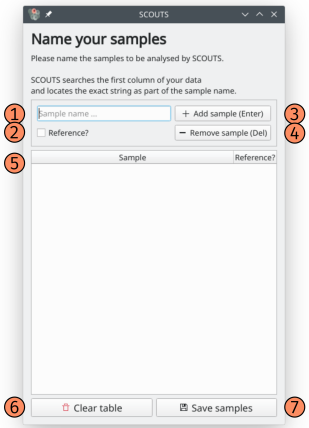
11) Run: click here to start SCOUTS. The button cannot be clicked again while SCOUTS is running (although you can still cancel SCOUTS by closing the program). Once SCOUTS has finished, a message box will appear informing you when that the analysis is done.

12) Help & Quit: used to open the online documentation and to exit SCOUTS.

Sample names window

In this window, the user must choose what samples to analyse. It is crucial that sample names are present in the first column of your input data.

These are the elements of the sample names window:



1) Sample name: type your sample name here.

2) Reference checkbox: check this box to mark the sample in the box above as the reference sample, for OutR selection.

3) Add sample: click here to add the sample to the list (shortcut: Enter).

4) Remove sample: click here to remove a highlighted sample from the list (shortcut: Delete).

5) Sample table: this table keeps track of all samples, and which sample is the reference (if any).

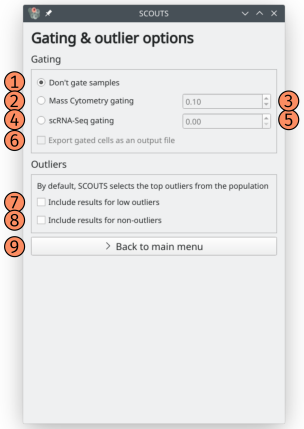
6) Clear table: click this button to clear all samples from the table.

7) Save samples: click this button to accept the names in the sample table and go back to the main window.

Gating window

In this window, users can choose whether to apply gating to cells *before* running SCOUTS, and to generate output for different cell populations.

These are the elements of the gating window:



1) No gating: do not gate your input data. This is the default option.

2) Mass cytometry gating: this gating option aims to exclude poorly stained cells from a mass cytometry experiment. Cells that have an **average row expression lower than the gating value** will be **excluded**.

3) Mass Cytometry gating value: the value to use for mass cytometry gating. Only applies if mass cytometry gating is active.

4) Gate scRNA-Seq: this gating option aims to exclude cells with low read counts. Cells that have an **read count for a given marker smaller or equal to the gating value** will **not contribute for the cutoff value** of that marker. Additionally, a cell that has **all markers below the gating value** will be **excluded**.

5) scRNA-Seq gating value: the number of reads to use for scRNA-Seq gating. Only applies if scRNA-Seq gating is active.

6) Export gated cells: select this option to generate an extra output file (.xlsx) that's identical to your input file, except for the absence of the gated cells.

7) Include results for low outliers: select this option to also generate output for *low outliers*, i.e. cells that are outliers because of their low expression profile.

8) Include results for non-outliers: select this option to also generate output for *non-outliers*, i.e. cells that are not outliers because their expression profile is not too high nor too low.

9) Back to main menu: click this button to go back to the main window, saving current gate settings.

2.4.2 How SCOUTS selects outliers

SCOUTS treats each marker/sample combination as being a subpopulation, and calculates the first quartile (Q1) and third quartile (Q3) for this subpopulation.

The cutoff value for each marker/sample combination is, in turn, calculated using Tukey's fences:

```
upper cutoff = Q3 + (IQR * t)
```

```
lower cutoff = Q1 - (IQR * t)
```

where `IQR` is the interquartile range (`Q3 - Q1`) and `t` is the Tukey factor.

The quantiles are calculated by linear interpolation. See the Pandas documentation on quantiles to learn more about this.

Having the cutoff value from each marker/sample combination, SCOUTS proceeds to select cells from the input table with expression values higher than the upper cutoff (top outliers), lower than the lower cutoff (for bottom outliers) or in between (for non-outliers).

Depending on user choice, outliers for *each marker* may be selected, or outliers for *at least one* marker. Additionally, the cutoff value used may come from a reference sample (OutR) or from each sample itself (OutS).

2.4.3 About input files

The input file for SCOUTS should have:

- a header containing all markers. This is the first line of a .csv file, or the first row of an Excel spreadsheet;
- Cell ID in the first (leftmost) column. This is the field that SCOUTS uses to search for sample names, so each cell ID must have a name that conveys from which sample it belongs to;
- expression values of cell ID x marker, for all other positions in the input data.

Deviations from these rules will likely result in a failed/error-prone analysis.

2.4.4 About sample names

When starting the analysis, SCOUTS divides the input data into samples. SCOUTS searches for each sample (i.e. each name in the sample table) in the first column of the input data. Sample names are **case-sensitive**, so be sure to type them correctly.

SCOUTS will throw explicit errors if you:

- try to run the program with an empty sample list;
- try to perform OutR analysis with no reference sample;
- none of the sample names are found in the first column of the input data.

SCOUTS will **not** stop the analysis nor warn you if:

- *some, but not all* sample names are found in the first column of the input data (these sample names will be ignored).
- a given sample name appears in more than one subset of samples (these cells will be analysed twice).
- some cells do not belong to any sample (these cells will be ignored)

Make sure to use sample names that are unique to each sample, and double-check if you have included all samples!

2.4.5 About output files

SCOUTS creates a subfolder called `data` in the output folder. In this folder, every file corresponds to a different subpopulation of outliers selected by SCOUTS.

Output files are organized in numerical order. The `summary.xlsx` spreadsheet contains the correspondence between file number and outliers selected.

SCOUTS also generates the following output files in the output folder:

- `stats.xlsx`: contains information of number of cells, mean, median and standard deviation in the different populations selected.
- `cutoff_values.xlsx`: contains the upper and lower cutoff values for each sample x marker combination.
- `gated_population.xlsx` (optional): contains the whole gated population, prior to SCOUTS

- `merged_data.xlsx` (optional): contains all individual Excel file in `data` as spreadsheets in a single Excel workbook.

2.5 How SCOUTS-violins works

2.5.1 About SCOUTS-violins

SCOUTS-violins was designed as an exploratory interface through which users can visualize the results from a SCOUTS analysis.

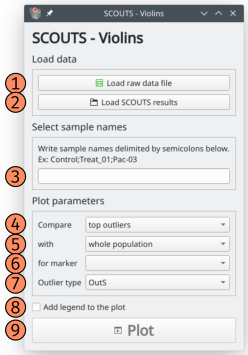
Note that some options in SCOUTS-violins may result in errors, depending on how you chose to use SCOUTS - e.g. you can't inspect non-outliers or OutR outliers if you did not select those rules when running SCOUTS.

2.5.2 Page elements

This section explains what every button and option of the SCOUTS-violins interface does.

Main window

These are the elements of the main window:



1) Load raw data: Select the same input file given to SCOUTS.

2) Load SCOUTS results: Select the same output folder used to save SCOUTS results.

3) Select sample names: Write sample names to plot, separated by semicolons (e.g. "control;patient;drug01").

4) and 5) Populations to compare: Select populations to compare on the violin plot. The population in the top selection box appears below the population in the bottom box.

6) Marker: Select marker for which to compare populations. Markers are automatically loaded from the input file header.

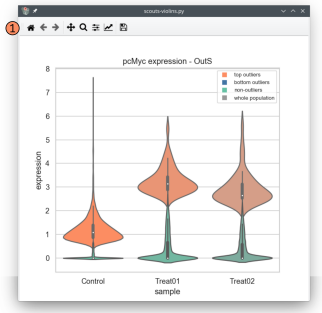
7) Outlier type: select whether to plot OutS or OutR outliers.

8) Legend: Whether do display a legend in the figure.

9) Plot: click here to plot the figure with the selected parameters. The figure opens in a separate window, and can be reloaded with new parameters by clicking again on this button.

Plot window

This is the plot window:



The plot window works like a regular matplotlib figure window - it can be stretched, edited and modified. You can also use the tool bar at the top (**1**) to zoom, move and export the plot in different formats.

2.6 FAQ

2.6.1 Errors when starting SCOUTS

I can't start SCOUTS on my computer! `scouts` **does nothing!**

Please check the installation troubleshooting section.

I can't start SCOUTS-violins on my computer! `scouts-violins` **does nothing!**

Please confirm that you have installed the necessary dependencies. Then check the installation troubleshooting section.

2.6.2 About the SCOUTS workflow

What type of pre-processing does SCOUTS perform on my input?

By default, SCOUTS does not perform any pre-processing prior to analysing the input data. This means that transformations like data normalization and scaling should be performed by the user prior to using SCOUTS.

SCOUTS does, however, include a simple module for gating mass cytometry and scRNA-seq samples, should the user want to do so. Refer to How SCOUTS works for details.

My data is stored in multiple files. How do I use SCOUTS with it?

Ideally, you are able to merge your files into a single input file and use that to work with SCOUTS. When this is not possible, you can still use SCOUTS to run individual samples by passing one file at a time. This will not affect the output, *as long as all the cells from one sample are together in one file*. One side-effect of this method is that, in order to use the "*control*" option in the "*Consider outliers using cutoff from*" field, you need to make sure that the cells from the control sample are present in every file.

Does the output for non-outliers exclude both top and bottom outliers?

Yes. Even if you did not generate output files for bottom outliers, non-outliers are still considered as being "the population of cells that are not top outliers, nor bottom outliers".

2.6.3 Misc.

I have another question/suggestion/issue when using SCOUTS. How do I contact you?

You can contact us through our Github page. Check the authors section for more information. Also remember that you can use the Github issues to post about problems or suggestions for SCOUTS.

SCOUTS is working nicely, but can you implement X?

Maybe! Tell us more about it on the Github issues page :-)

If you use SCOUTS, please cite us.

CHAPTER 3

Authors

Juliano Faccioni - Programming and GUI development

- GitHub
- LinkedIn

Giovana Onzi - Concept and testing

- LinkedIn

CHAPTER 4

License

This project is licensed under the MIT License.

CHAPTER 5

Acknowledgements

Scientific Counseling

- Prof. Dr. Guido Lenz

Funding

- CAPES/CNPq
- NIH