Cytofpipe v1.1

This pipeline was developed to by Lucia Conde at the BLIC - UCL Cancer Institute, in collaboration with Jake Henry from the Immune Regulation and Tumour Immunotherapy Research Group, for the automatic analysis of flow and mass cytometry data in the UCL cluster **Legion**. Any UCL researcher can access the pipeline via legion. Researchers from other institutions can download the stand-alone version of Cytofpipe and run it directly in their personal computers. An external script developed by Heng Li (https://github.com/lh3/asub) is also provided so that users can easily submit Cytofpipe to their cluster queue system (LSF, SGE/UGE, SLURM) if desired.

Cytofpipe v1.1 can be used 1) to run standard cytometry data analysis for subset identification, 2) for comparison of groups of samples, and 3) to construct scaffold maps for visualizing complex relationships between samples. The methods underneath Cytofpipe v1.1 are based on publicly available R packages for flow/cytof data analysis

- Cytofpipe **–clustering** is based mainly on cytofkit (https://bioconductor.org/packages/release/bioc/html/cytofkit.html), which is used for preprocessing/clustering, and openCyto (https://www.bioconductor.org/packages/release/bioc/html/openCyto.html), for basic automated gating.
- Cytofpipe **–scaffold** is based on scaffold (https://github.com/nolanlab/scaffold)
- Cytofpipe **–citrus** is based on citrus (https://github.com/nolanlab/citrus)

How to run Cytofpipe v1.1

1. Using the UCL cluster

1.- Connect to legion and bring inputfiles:

You will need to connect to legion (apply for an account here: https://wiki.rc.ucl.ac.uk/wiki/Account_Services), and transfer there the inputfiles.

For example if you are doing clustering, that would be a folder with the input FCS files, a file with the list of clustering markers and optionally a config file.

To connect to legion, you can use Putty if you have Windows (check this UCL link: https://wiki.rc.ucl.ac.uk/wiki/Accessing_RC_Systems) or use SSH from your Mac terminal:

```
$ ssh UCL_ID@legion.rc.ucl.ac.uk
```

To transfer the files to legion, you can either use SCP from your laptop:

```
$ scp -r FILES
UCL_ID@legion.rc.ucl.ac.uk:/home/user/Scratch/cytof_data/.
```

or if you have a FTP transfer program (for example cyberduck: http://download.cnet.com/Cyberduck/3000-2160_4-10246246.html or WinSCP: https://winscp.net/eng/download.php) you can also transfer the files from/to legion simply by dragging them from one window to another.

2. Load modules:

Once you are in legion, you will need to load the modules necessary for the pipeline to work.

```
$ module load blic-modules
```

```
$ module load cytofpipe/v1.1
```

3.- Submit the job:

Let's say you have a folder called 'cytof_data' in your home in legion that contains a directory with the FCS files, a file that contains the markers that you want to use for clustering, a file listing which samples belong toeach condition, and perhaps a config file, for example:

```
/home/user/Scratch/cytof_data/
/home/user/Scratch/cytof_data/inputfiles/
/home/user/Scratch/cytof_data/inputfiles/file1.fcs
/home/user/Scratch/cytof_data/inputfiles/file2.fcs
/home/user/Scratch/cytof_data/inputfiles/file3.fcs
/home/user/Scratch/cytof_data/inputfiles/gated/file1_cellType1.fcs
/home/user/Scratch/cytof_data/inputfiles/gated/file1_cellType2.fcs
/home/user/Scratch/cytof_data/inputfiles/gated/file1_cellType3.fcs
/home/user/Scratch/cytof_data/inputfiles/gated/file1_cellType4.fcs
/home/user/Scratch/cytof_data/markers.txt
/home/user/Scratch/cytof_data/conditions.txt
/home/user/Scratch/cytof_data/config.txt
```

To run the pipeline in "clustering" mode with default parameters, just go to the 'cytof_data' folder and run:

```
$ cytofpipe --clustering -i inputfiles -o results -m markers.txt
```

That will crate a new folder called “results” that will contain all the results of the analysis.

Similarly, to run the pipeline in “scaffold” mode with default parameters:

```
$ cytofpipe --scaffold -i inputfiles --ref file1.fcs -o results -m
markers.txt
```

Finally, to run the pipeline in “citrus” mode with default parameters:

```
$ cytofpipe --citrus -i inputfiles --cond conditions.txt -o results -m
markers.txt
```

4.- Errors before running

When you submit the job, before it actually runs, there is a JSV script that checks that everything is in order. For example, that the inputfiles folder exists, that there is not a results folder already there (so that nothing is overwritten), that if a config.txt file is inputed it has the appropriate format, etc… Only if everything looks fine, the job will be submitted. Otherwise, an error message will appear that will tell you that there is a problem. For example:

```
Program: Cytofpipe --clustering
Version: 1.1
Contact: Lucia Conde <l.conde\@ucl.ac.uk>

Usage:   cytofpipe --clustering -i DIR -o DIR -m FILE [options]

Required: -i DIR       Input directory with the FCS files
          -o DIR       Output directory where results will be generated
          -m FILE      File with markers that will be selected for clustering
Options: --config FILE            Configuration file to customize the analysis
         --flow|--cyto            Use autoLgcl (flow) or cytofAsinh (cytof)
                                   transformation [--cytof]
         --all|--downsample NUM   Use all events or downsample each FCS to NUM
                                  [--downsample 10000
         --displayAll             Display all markers in output files [NULL]
         --randomSampleSeed      Use a random sampling seed instead of default
                                 seed used for reproducible expression matrix
                                 merging
         --randomTsneSeed        Use a random tSNE seed instead of default
                                 seed used for reproducible tSNE results
         --randomFlowSeed        Use a random flowSOM seed instead of default
                                 seed used for reproducible flowSOM results

Unable to run job: Please check that you are providing a inputdir (-i), output
dir (-o) and markersfile (-m)
Exiting.
```

5.- Check job is running

If no errors were found, the job will be submitted to the queue through a qsub system. To check that the job is queued or running, use qstat:

```
$ qstat
```

```
job-ID  prior   name       user         state submit/start at     queue
---------------------------------------------------------------------------------
2739095 3.50000 cytof-raw_ regmond      r     04/03/2017 10:52:31 Yorick@node-z00a-011
2739177 0.00000 cytof-gate regmond      qw    04/03/2017 10:59:51  1
```

In the above example I have one job (with ID 2739095) that is already running (state = r), and a second job (with ID 2739177) that is in queue (state = q).

If you submit a job, and later on it does not show when you do qstat, that means that it finished. You should then be able to see a new folder that has the results of the analysis.

2. Using a machine (mac, linux) without queue system

1.- Installation

Download cytofpipe from **URL** and uncompress it using tar:

```
$ tar -xvf cytofpipe_v1.1.tar.gz
```

That will create a 'cytofpipe_v1.1' masters folder that contains the code almost ready to use. You will just need to tell cytofpipe where you have downloaded the code. For that, open the cytofpipe_v1.1/Cytofpipe.pl perl script and change the variable CYTOFPIPE_HOME so that it points to the master folder:

```
$ENV{CYTOFPIPE_HOME}="/path/where/your/have/cytofpie_v1.1"
```

2.- Dependencies

You will need to have R installed and in your path (https://www.r-project.org/), as well as Pandoc (https://pandoc.org/). Pandoc is simply used to generate a summary PDF after each run. If you don't have pandoc installed is not a big issue: the summary PDF will not be generated and you will see an error in the terminal regarding this, but cytofpipe will run anyway and you should be able to see all the other output files.

Cytofpipe depends on several R packages, mainly:
cytofkit (https://bioconductor.org/packages/release/bioc/html/cytofkit.html)
scaffold (https://github.com/nolanlab/scaffold)
citrus (https://github.com/nolanlab/citrus)

as well as other not related to flow/cytof data analysis (ini, hash, …). Perhaps you have already installed some of these R packages locally in your system. However, it is possible that the Cytofpipe version that you have downloaded might only work with a specific version of these packages (for example, Cytofpipe_v1.1 works fine

with cytofkit_1.10, but not with cytofpipe_1.6, or potentially with newer cytofkit versions). This is because when a new version of'cytofkit' is released, if it has gone through substantial changes to the code, the code of Cytofpipe needs to be changed accordingly. Therefore, to avoid issues with version clashes we provide a folder Rlib that contains the necessary R packages to run Cytofpipe and we have checked that the versions are compatible. You don't need to do anything about this, every time that Cytofpipe is launched it will assign the R_LIBS environment variable to that Rlib folder.

3.- Submit the job:

Let's say you have a folder called 'cytof_data' that contains a directory with the FCS files, a file that contains the markers that you want to use for clustering, a file listing which samples belong to each condition, and perhaps a config file, for example:

```
/home/user/Scratch/cytof_data/
/home/user/Scratch/cytof_data/inputfiles/
/home/user/Scratch/cytof_data/inputfiles/file1.fcs
/home/user/Scratch/cytof_data/inputfiles/file2.fcs
/home/user/Scratch/cytof_data/inputfiles/file3.fcs
/home/user/Scratch/cytof_data/inputfiles/gated/file1_cellType1.fcs
/home/user/Scratch/cytof_data/inputfiles/gated/file1_cellType2.fcs
/home/user/Scratch/cytof_data/inputfiles/gated/file1_cellType3.fcs
/home/user/Scratch/cytof_data/inputfiles/gated/file1_cellType4.fcs
/home/user/Scratch/cytof_data/markers.txt
/home/user/Scratch/cytof_data/conditions.txt
/home/user/Scratch/cytof_data/config.txt
```

To run the pipeline in "clustering" mode with default parameters, just run the "cytofpipe.pl" perl script located in the masters cytofpipe_v1.1 folder:

```
$ /path/to/cytofpipe_v1.1/cytofpipe.pl --clustering -i inputfiles -o
results -m markers.txt
```

That will crate a new folder called "results" that will contain all the results of the analysis.

Similarly, to run the pipeline in "scaffold" mode with default parameters:

```
$ /path/to/cytofpipe_v1.1/cytofpipe.pl --scaffold -i inputfiles --ref
file1.fcs -o results -m markers.txt
```

Finally, to run the pipeline in "citrus" mode with default parameters:

```
$ /path/to/cytofpipe_v1.1/cytofpipe.pl --citrus -i inputfiles --cond
conditions.txt -o results -m markers.txt
```

Cytofpipe will first check that everything is in order. For example, that the inputfiles folder exists, that there is not a results folder already there (so that nothing is overwritten), that if a config.txt file is inputed it has the appropriate format, etc…

Only if everything looks fine, the job will be submitted. Otherwise, an error message will appear that will tell you that there is a problem. For example:

```
Program: Cytofpipe --clustering
Version: 1.1
Contact: Lucia Conde <l.conde\@ucl.ac.uk>

Usage:   cytofpipe --clustering -i DIR -o DIR -m FILE [options]

Required: -i DIR       Input directory with the FCS files
          -o DIR       Output directory where results will be generated
          -m FILE      File with markers that will be selected for
                       clustering
Options: --config FILE            Configuration file to customize the
                                  analysis
         --flow|--cyto            Use autoLgcl (flow) or cytofAsinh (cytof)
                                  transformation [--cytof]
         --all|--downsample NUM   Use all events or downsample each FCS to NUM
                                  [--downsample 10000]
         --displayAll             Display all markers in output files [NULL]
         --randomSampleSeed   Use a random sampling seed instead of default
                               seed used for reproducible expression matrix
                               merging
         --randomTsneSeed    Use a random tSNE seed instead of default seed
                              used for reproducible tSNE results
         --randomFlowSeed   Use a random flowSOM seed instead ofdefault
                             seed used for reproducible flowSOM results

Unable to run job: Please check that you are providing a inputdir (-i), output
dir (-o) and markersfile (-m)
Exiting.
```

3. Using a cluster with queue system

1.- asub:

While we have a Cytofpipe version that runs in the UCL cluster using its queue system, we don't provide a version that will take advantage of the queue system in other clusters outside UCL. However, this is easily accomplished by using the 'asub' script developed by Heng Li at the Braod Institute (https://github.com/lh3/asub), and that we provide in the Cytofpipe_v1.1 masters folder for convenience.

The only difference in running the script directly or via asub, is that if you use asub, the command line needs to be written into a file, and then that file is called with asub. For example, to submit a job directly you would do:

```
$ /path/cytofpipe.pl --clustering -i inputfiles -o results -m
markers.txt
```

The same job submited via asub would require you to write the above line into a file, and then call asub on the file:

```
$ echo "/path/cytofpipe.pl --clustering -i inputfiles -o results -m
markers.txt" > cmd.txt

$ /path/asub cmd.txt
```

Every time you run a jub via asub, asub will generate two folders and a file with names that start with 'asub_XXXXX', where XXXXX is a random number. These are just STDOUT and STDERR folders that can give clues if something went wrong with the job, but that can be ignored otherwise.

The advantage of using asub is that it makes possible to run Cytofpipe in the user's cluster regardless of their platform and queuing systems (LSF, GSE, SLURM). Additionally, it facilitates the submission of array jobs. For example, if you want to run several Cytofpipe jobs in parallel you could write a cmd.txt file that contains all your jobs, one per line, and all these jobs will be submitted as an array job and will be run in parallel. For example if you want to run a –clustering job using different configurations, you cna write a cmd.txt file like this:

```
/path/cytofpipe.pl --clustering -i inputfiles -o results_A -m markers.t
xt --config config_A.txt
/path/cytofpipe.pl --clustering -i inputfiles -o results_B -m markers.t
xt --config config_B.txt
/path/cytofpipe.pl --clustering -i inputfiles -o results_C -m markers.t
xt --config config_C.txt
```

and then run it like this:

```
$ /path/asub cmd.txt
```

Additionally, the user can specify different directrices for the queue scheduler via asub. For example, to run the above job but requesting 6 hours of running time and 10Gb of RMA per job, you would do:

```
$ /path/asub -M 10 -W 2:0:0 cmd.txt
```

All the asub options can be found in Heng Li's github page (https://github.com/lh3/asub) or by running asub without arguments:

```
$ /path/asub
```

2.- Check submission

Sometimes, particularly when the user requests a lot of time, memory, nodes, or simply when the cluster is busy, the job might be queued for a long time before ir runs. And one frustrating thing is to find out that, after being in queue for perhaps hours, the job finally starts running but immediately stops because one of the

arguments given by the user was invalid, or because one if the input files was mispelled.

To avoid this, cytofpipe also contains a script in the masters folder called "check_submission.pl", that do some checkings of the arguments passed to cytofpipe.pl. In reality, this script is just an almost identical copy of cytofpipe.pl, which checks that everything is in order before doing any analysis. To use it, simply run it with the same arguments that you want to pass to cytofpipe.pl.

For example, if your cmd.txt file is

```
/path/cytofpipe.pl --clustering -i inputfiles -o results -m markers.txt
```

you can check that the job will run fine when submitted via asub if you first type:

```
$ /path/checkSubmission.pl --clustering -i inputfiles -o results -m
markers.txt
```

checkSubmission.pl will check that you indeed have a folder called 'inputfiles' and a filed called 'markers.txt', that there is not a folder called 'results' already there (so that nothing is overwritten), or if you also use a config file, it will check that it has the appropiate format. If everything seems correct, you will see a "No issues detected" message, otherwise an error message will appear that will tell you that there is a problem. For example:

```
Program: Cytofpipe --clustering
Version: 1.1
Contact: Lucia Conde <l.conde\@ucl.ac.uk>

Usage:   cytofpipe --clustering -i DIR -o DIR -m FILE [options]

Required: -i DIR       Input directory with the FCS files
          -o DIR       Output directory where results will be generated
          -m FILE      File with markers that will be selected for
                       clustering
Options: --config FILE            Configuration file to customize the
                                  analysis
         --flow|--cyto            Use autoLgcl (flow) or cytofAsinh (cytof)
                                  transformation [--cytof]
         --all|--downsample NUM   Use all events or downsample each FCS to NUM
                                  [--downsample 10000]
         --displayAll             Display all markers in output files [NULL]
         --randomSampleSeed   Use a random sampling seed instead of default
                               seed used for reproducible expression matrix
                               merging
         --randomTsneSeed    Use a random tSNE seed instead of default seed
                              used for reproducible tSNE results
         --randomFlowSeed   Use a random flowSOM seed instead ofdefault
                             seed used for reproducible flowSOM results

ERROR: Can't find markers file <markers.tx>
```

However, please note that checkSubmission.pl will only do some inital basic checking. It will check that all the arguments given to Cytofpipe are valid, but **will not examine throughly every single aspect that could go wrong with your job**. For example, it will not check whether the markers listed in the markers file indeed exist in the FCS files, or if the FCS files are corrupted.

In any case, to avoid finding out about a mispelled markers filename after being waiting in queue for hours, we recommend that you use checkSubmission.pl in your command line before submitting it with asub (if you don't use asub, this is not necessary, because cytofpipe.pl will do the checkings anyway as soon as you submit it)

Cytofpipe v1.0 commands

–clustering

```
Usage: cytofpipe --clustering -i DIR -o DIR -m FILE [options]
```

Cytofpipe **–clustering** can be used to analyze data from multiple FCS files.

First, FCS files will be merged, expression data for selected markers will be transformed, and data will be downsampled according to the user's specifications. Then, clustering will be performed to detect cell types. Finally, the high dimensional flow/mass cytometry data will be visualized into a two-dimensional map with colors representing cell type, and heatmaps to visualize the median expression for each marker in each cell type will be generated.

- *Note 1*: The markers uploaded by the user should be the ones provided in the "Description" filed of the FCS. This will usually be a longer format (141Pr_CD38) in cytof data and a shorter format (CD38) in Flow data. However, the shorter version can be used when uploading cytod data.
- *Note 2*: Markers uploaded by the user are used for clustering and dimensional reduction, and by default they are the only ones displayed in the results (heatmaps, etc..). Using the **–displayAll** option will override this and all the markers (with exception of Time, Event, viability and FSC/SSC channels) will be included in the output plots and files. All markers are transformed to the user's specifications with exception of FSC/SSC that are linearly transformed.
- *Note 3*: Cytofpipe v1.1 runs cytofkit_1_10_0

Cytofpipe assumes that the data has been properly preprocessed beforehand, i.e., that normalisation, debarcoding and compensation (if flow) were done properly, and that all the debris, doublets, and live_neg events were removed before analysis. However, if manual gating has not been done, automatic gating of live events can be done by setting the "Gating" option in the config file to "YES". But importantly, the default Cytofpipe gating strategy (shown below) is based on the manual gating example provided by Jake. **The gating will not work properly on datasets coming from other sources where the staining panel, instrument, or other factors are different than the ones used by Jake at the Cancer Institute.**

Root —> BeadNeg —> Cells —> Live

Automated gating can be performed for more complex assays, however, a proper gating template needs to be created before using the pipeline for gating these dtasets. Any type of dataset can be used with the current pipeline if gating is not needed.

Command arguments

Mandatory arguments

-i DIR: A folder with FCS files

-o DIR: The name for the folder where you want to output the results. It can not be an existing folder.

-m FILE: A text file with the names of the markers, one per line. For example:

```
CD3
CD4
CD8
FOXP3
TCRgd
CD11b
CD56
HLA-DR
```

Optional arguments

–config FILE: The config file is not mandatory. If is not provided, the pipeline will use a default config.txt file, which has GATING = no, TRANSFORM = cytofAsinh, MERGE = ceil (n = 10,000), PHENOGRAPH = yes (other clustering methods = no), DISAPLAY_ALL = no, TSNE parameters: perplexity = 30, theta = 0.5, max_iter = 1000. If provided, it has to have the following format:

```
[ clustering ]              #-- MANDATORY FIELD, IT SHOULD BE THE FIRST LINE
                            OF THE CONFIG FILE
```

```
GATING = yes|no             #-- MANDATORY FIELD:

TRANSFORM = autoLgcl, cytofAsinh, logicle, arcsinh or none  #-- MANDATORY
MERGE = ceil, all, min, or fixed                            #-- MANDATORY
DOWNSAMPLE = number between 500 and 100000              #-- MANDATORY if
                                                        MERGE = fixed/ceil

#- DimRed method (tSNE) parameters:
PERPLEXITY = 30
THETA = 0.5
MAX_ITER = 1000

#- Clustering methods:
PHENOGRAPH = yes|no
CLUSTERX = yes|no
DENSVM = yes|no
FLOWSOM = yes|no
FLOWSOM_K = number between 2 and 50           #-- MANDATORY if FLOWSOM = YES:

#- Additional visualization methods:
PCA = yes|no
ISOMAP = yes|no

#- Other:
DISPLAY_ALL = yes|no
RANDOM_SAMPLE_SEED = yes|no
RANDOM_TSNE_SEED = yes|no
RANDOM_FLOW_SEED = yes|no
```

–flow, **–cytof**: Shorcut to let cytofpipe know if the user is analyzing flow or cytof data, without having to provide a config file. If **–flow** is selected, the autoLgcl transformation will be used. If **–cytof** is selected, the cytofAsinh transformation will be used. **–flow** and **–cytof** cannot be used at the same time, and they will override the TRANSFORMATION option of the config file if a cofig file is supplied too.

–all, **–downsample NUM**: Shorcut to let cytofpipe know if we want to use all the events/downsample the data, without having to provide a config file. **–all** and **–downsample NUM** cannot be used at the same time, and they will override the DOWNSAMPLE option of the config file if a cofig file is supplied too.

–displayAll: Shorcut to let cytofpipe know if we want to display all the markers in the output files and plots, without having to provide a config file. **–displayAll** will override the DISPLAY_ALL option of the config file if a cofig file is supplied too. Please note that Time, Event, viability and FSC/SSC channels will not be displayed even if the **–displayAll** option is selected. Please contact me if you want to change this.

–randomSampleSeed: Force cytofpipe to use a random seed for expression matrix merging. I.e., if the user is downsampling the data, a different set of random cells will

be picked up in each run, as opposed to the default cytofpipe configuration which uses a seed to ensure reproducibility of the expression matrix. **–randomSampleSeed** will override the RANDOM_SAMPLE_SEED option of the config file if a config file is supplied too.

–randomTsneSeed: Force cytofpipe to use a random seed for tSNE analysis to avois tSNE reproducibility in each run. **–randomTsneSeed** will override the RANDOM_TSNE_SEED option of the config file if a config file is supplied too.

–randomFlowSeed: Force cytofpipe to use a random seed for FlowSOM analysis to avoid reproducibility in each run. **–randomFlowSeed** will override the RANDOM_FLOW_SEED option of the config file if a config file is supplied too.

Outputfiles

Rphenograph: Contains the data/plots from the clustering. i.e., the markers average values per cluster, cell percentages in each cluster per FCS file, heatmaps… If more than one clustering method was selected, then there will be several folders, one per clustering method (i.e., one Rphenograph folder, one clusterX folder, etc..)

Gating: If GATING was selected, there will be folder with plots that show the gating applied to each FCS file, a plot that shows the gating scheme, and a folder called "gating_fs_live" that will have the FCS files for the live population.

cytofpipe_analyzedFCS: The original FCS files (i.e., the original files if gating = YES, or the "gating_fs_live" files if gating = NO), with the clutering and tSNE added information.

cytofpipe_tsne_level_plot.pdf: Marker level plot (shows the expression level of markers on the tSNE data in one plot)

Marker_level_plots_by_sample: Marker level plots separated by FCS file

cytofpipe_tsne_dimension_reduced_data.csv: tSNE data (tSNE1 and tSNE2 values per event)

cytofpipe_markerFiltered_transformed_merged_exprssion_data.csv: Expression data (expression of each marker per event)

summary_clustering.pdf: This is a PDF with a summary of the analysis. It describes what files, markers and config options were used, and shows the main plots from the analysis (gates, cluster plots, marker level plots, heatmaps…).

cytofpipe.RData: The object resulting from the cytofkit analysis. i.e., a file that was saved and that can be used for loading to the cytofkit shiny APP to further exploration of the results.

log_R.txt: This is just the log file from the R script, just so that the user can see what R commands were run when doing the analysis. It will help me figure out what the problem is if the job finishes with an error.

--scaffold

```
Usage: cytofpipe --scaffold -i DIR -o DIR -m FILE --ref FILE [options]
```

Cytofpipe **–scaffold** can be used to generate scaffold maps to compare cell population networks across different conditions.

Cytofpipe clusters each FCS sample file independently (currently set up to 200 clusters) using the clara function in R, as implemented in scaffold. A graph is then constructed connecting the nodes fom the manually gated populations and the clusters from the reference FCS file, with edge weights defined as the cosine similarity between the vectors of median marker values of each cluster. Edges of low weight are filtered out and the graph is then laid out (shaped) using a ForceAtlas2 algorithm implemented in scaffold. Graphs are generated for every FCS file, where the position of the landmark nodes stay constant, providing a visual reference that allows the comparison of the different datasets.

- *Note 1*: Because the clustering is very computationally intensive, by default cytofpipe downsamples the original FCS files to 10,000 events (with NO replacement if the total number of cell in the file is less than 10,000), and then **all the clustering and construction of maps are done on these downsampled files** to be able to run the jobs in a timely fashion. This can be changed with the **–all** and **–downsample NUM** arguments.
- *Note 2*: Cytofpipe v1.1 is running scaffold_0.1

Cytofpipe assumes that the FCS data has been properly preprocessed beforehand, i.e., that normalisation, debarcoding and compensation (if flow) were done properly, and that all the debris, doublets, and live_negevents were removed before analysis. With regards to compensation, please note that the software will try to apply a compensation matrix if one is found in the FCS file. So if the data in the file is already compensated, it will be erroneously compensated again. If your data needs compensation make sure that the FCS file has a spillover matrix embedded and the data is uncompensated.

Command arguments

Mandatory arguments

-i DIR: Folder with FCS files. It should also contain a “gated” subfolder with the manually gated populations (see below).

–ref FILE: The FCS file that will be used for the construction of the reference map together with the landmark (manually gated) populations. It should be one of the FCS files inside INPUTDIR

-o DIR: Name for the folder where you want to output the results. It can not be an existing folder.

-m FILE: A text file with the names of the markers that will be used in the clustering step, one per line. For example:

```
CD3
CD4
CD8
FOXP3
TCRgd
CD11b
CD56
HLA-DR
```

The landmark populations have to be provided as single FCS files (one for each population) that need to be located in a subdirectory called "gated" of the INPUTDIR directory. The program will split the name of the FCS file using "_" as separator and the last field will be used as the population name. For instance if you want an FCS file to define your "B cells" population you have to use the following naming scheme: XXXX_B cells.fcs

Optional arguments

–flow, **–cytof**: Shorcut to let cytofpipe know if the user is analyzing flow or cytof data, without having to provide a config file. If **–flow** is selected, arcsinh transformation will be used with asinh_cofactor = 150. If **–cytof** is selected, arcsinh transformation will be used with asinh_cofactor = 5. **–flow** and **–cytof** cannot be used at the same time

–all, **–downsample NUM**: Shorcut to let cytofpipe know if we want to use all the events/downsample the data, without having to provide a config file. **–all** and **–downsample NUM** cannot be used at the same time.

Outputfiles

Clustering: Contains the files created after the clustering step. For each FCS files two files will be created:

*.clustered.txt: this file contains the marker medians for each cluster

*.clustered.all_events.RData: this file is an RData object which contains all the events in the original FCS file but with an added column that specifies the cluster membership. The data in this file is arcsinh transformed

downsampled_X: If downsampling, this folder will contain the FCS files created after downsampling the original FCS files to the selected number of events. **If downsampling, these would the FCS files that are actually analised**.

scaffold_map_XXX.pdf: These are the PDFs with the scaffold maps, one for each input dataset. By default, landmark nodes are coloured in red and population clusters in blue.

XXX.scaffold: A .scaffold file with the same name of the dataset that you have used as reference. This is a single self-contained bundle that contains everything you need to browse the data. It can be loaded into the original scaffold software for further exploration of the results.

summary_scaffold.pdf: This is a PDF with a summary of the analysis. It describes what files, markers and config options were used, and shows the scaffold maps in reduced size.

log_R.txt: Log file from the R script, just so that the user can see what R commands were run when doing the analysis. It will help me figure out what the problem is if the job finishes with an error.

--citrus

```
Usage: cytofpipe --citrus -i DIR -o DIR -m FILE --cond FILE [options]
```

Cytofpipe **–citrus** can be used to identify cell populations associated with an experimental or clinical endpoint in flow and cytof data.

After indicating which samples belong to each condition, expression data are transformed (using the arcsin hyperbolic transform) and scaled, and cell populations are indentified in every sample using hierarchical clustering based on selected markers. Sample features are calculated (either abundance of cell populations in each sample [default], or median expression of specific markers in a cluster), and these are used to identify the features (cell populations or markers) that are likely to be predictive (pamr, glmnet) or correlated (sam) with the experimental/clinical endpoint. Briefly, predictive models identify the fewest number of markers needed to predict the experimental endpoint. Correlative models detect all the markers that are correlated with the experimental endpoint but that are not necessarily accurate predictors of an experimental outcome. Among the predictive models, 'pamr' can be used with 2 or more groups, whereas 'glmnet' can only deal with 2 groups or continuous endpoint measures.

- *Note 1*: Citrus requires 8 or more samples in each experimental group. Running Citrus with fewer than 8 samples per group will likely produce spurious results.
- *Note 2*: To ensure that each sample is equally represented in the clustering, by default Citrus selects an equal number of events from each sample (10,000) that are combined and clustered together. This can be changed with the **–all** and **–downsample NUM** arguments.

- *Note 3*: By default, cytofpipe –citrus runs in "abundances" mode. You can change to "medians" mode using the **–medians FILE** parameter by supplying a list of amrkers used for median calculation. *Markers that were selected for clustering should not be selected again for statistics (for example, you might want to use surface markers for clustering and functional markers for median level expression calculation).*
- *Note 4*: By default, clusters (cell populations) of size lower than 5% of the total number of clustered events will be ignored. If you wish to change the minimum cluster size threshold please contact me.
- *Note 5*: Parameters must be measured on the same channels in each file, the same parameters must be measured in all FCS files (no extras or missing parameters in any FCS file) and measured parameters and channels must appear in the same order in each FCS file.
- *Note 6*: Due to the stochastic nature of the citrus algorithm, is recommended to run (repeat) the analysis at least 3 times to make sure the results are not false positives.
- *Note 7*: Cytofpipe v1.1 runs citrus_0.08

Cytofpipe assumes that the data have been properly preprocessed beforehand, i.e., that normalisation, debarcoding and compensation (if flow) were done properly, and that all the debris, doublets, and live_neg events were removed before analysis. For flow cytometry data, raw FCS file data must be compensated. Compensation matrices stored in FCS files will not be applied when running Cytofpipe in –citrus mode.

Command arguments

Mandatory arguments

-i DIR: Folder with FCS files.

–cond FILE: A text file that tells Citrus which samples belong to each condition/group, one sample per line. For example:

```
Patient1_FL2.fcs    Case
Patient2_FL2.fcs    Case
Patient3_FL2.fcs    Case
Patient4_FL2.fcs    Case
Patient5_Ref.fcs    Control
Patient6_Ref.fcs    Control
Patient7_Ref.fcs    Control
Patient8_Ref.fcs    Control
```

-o DIR: Name for the folder where you want to output the results. It can not be an existing folder.

-m FILE: A text file with the names of the markers that will be used in the clustering step, one per line. For example:

```
CD3
CD4
CD8
FOXP3
TCRgd
CD11b
CD56
HLA-DR
```

Optional arguments

-medians FILE: A text file with the names of the markers that will be used as features, one per line. For example:

```
p-NFkB
p-S6
p-PI3K
p-STAT5
```

–flow, **–cytof**: Shorcut to let cytofpipe know if the user is analyzing flow or cytof data, without having to provide a config file. If **–flow** is selected, arcsin hyperbolic transformation will be used with asinh_cofactor = 150. If **–cytof** is selected, arcsin hyperbolic transformation will be used with asinh_cofactor = 5. **–flow** and **–cytof** cannot be used at the same time

–all, **–downsample NUM**: Shorcut to let cytofpipe know if we want to use all the events/downsample the data, without having to provide a config file. **–all** and **–downsample NUM** cannot be used at the same time.

Outputfiles

MarkerPlots.pdf: Plots of the clustering hierarchy, one marker per pdf. Helpful for determining the phenotype of identified clusters.

MarkerPlotsAll.pdf: Same content as MarkerPlots.pdf in a single PDF instead of many.

sam_/pamr_/glmnet_results: A directory containing results from model analysis. There will be one result for each model used to analyze the data. All models operate on the same clustering result. Within a mode result directory you will find:

clusters-{threshold}.pdf: Shows phenotype of stratifying clusters at specified significance threshold.

features-{threshold}.pdf: Plot showing values stratifying features at specified significance threshold.

features-{threshold}.csv: Raw values of stratifying features at specified significance threshold.

featurePlots-{thresholds}.pdf: Shows the the location and relatedness of identified stratifying clusters in the clustering hierarchy. There will be a separate plot for each tested feature.

ModelErrorRate.pdf: Used to determine the accuracy of the constructed model. For predictive models only (PAMR and GLMNET).

exportedClusters: Folder that contains data (FCS files) exported from each cluster, useful for further analysis on the clusters of interest.

summary_citrus.pdf: This is a PDF with a summary of the analysis. It describes what files, markers and config options were used, as well as the MarkerPlotsAll plot.

cytofpipe_citrusClustering.rData: Saved version of the clustered data that was computed during the citrus analysis, which can be loaded in R for further exploration of the results.

log_R.txt: This is just the log file from the R script, just so that the user can see what R commands were run when doing the analysis. It will help me figure out what the problem is if the job finishes with an error.

Version changes

v1.1

- Added –randomSampleSeed, –randomTsneSeed, –randomFlowSeed options
- Versions: cytofkit 1.10.0, scaffold 1.0, citrus 0.08

v1.0

- Changed command line usage
- Added –flow, –cytof, –all, –downsample, –displayAll options
- By default, outputfiles display only clustering Markers (use –displayAll to override)
- Marker level plots per sample are shown on the same scale on the x- and y-axis
- Fixed bug with mergeMethod=fixed from previous cytofkit version (https://github.com/JinmiaoChenLab/cytofkit/issues/12)
- More –citrus functionalities exposed ('medians' mode, clusters exported as new FCS files)
- Versions: cytofkit 1.10.0, scaffold 1.0, citrus 0.08

v0.3

- Added –citrus mode
- Versions: cytofkit 1.8.4, scaffold 1.0, citrus 0.08

v0.2.1

- Output marker level plots per FCS fiyle

v0.2

- Added –scaffold mode
- Major changes to adapt cytofpipe to updated cytofkit 1.8.3

Questions?

Email me at l.conde@ucl.ac.uk

Last modified Mar 2018.