## Hardware and OS requirement

Our pipeline has been tested on Linux computers. Because it requires creation of a virtual machine with 16GB memory and 200GB disk size, please make sure you have enough memory and free disk space before you proceed. The following instructions require the Bash shell, Perl, as well as GNU utils such as find and sed.

## Installing software

1. Install docker-machine (instructions available at: https://github.com/docker/machine/releases).

2. Start Bash shell and run the following command to create and activate a docker-machine VM with 16GB memory and 200GB disk:

```
# docker-machine create -d virtualbox --virtualbox-cpu-count "1" --virtualbox-memory "16384" --virtualbox-disk-size "200000" default
# docker-machine start default
# eval "$(docker-machine env default)"
```

   If you see error messages such as:

   *"Error checking TLS connection: Error checking and/or regenerating the certs: There was an error validating certificates for host "192.168.99.105:2376": x509: certificate is valid for 192.168.99.100, not 192.168.99.105"*

   Or

   *"An error occurred trying to connect: Post https://192.168.99.110:2376/v1.19/containers/create: dial tcp 192.168.99.110:2376: no route to host"*

   You may solve the problem by:

```
# docker-machine regenerate-certs default
# eval "$(docker-machine env default)"
```

3. Download and import our pipeline from the docker registry:

```
# docker pull lyongu/expressionheterogeneityparameters:version0.1
```

4. Stop docker-machine:

```
# docker-machine stop default
```

5. Download additional scripts and our dataset ( https://heterogeneity.niaid.nih.gov/immune_heterogeneity/data/DATA.tgz), decompress into a data folder (we use $HOME/DATA; if you use a different path, replace it with the actuall path in the following command). Also update the PATH to allow running our scripts:

```
# wget https://heterogeneity.niaid.nih.gov/immune_heterogeneity/data/DATA.tgz
# tar -xzf DATA.tgz -C $HOME
# DATA_FOLDER="$HOME/DATA"
# export PATH=$DATA_FOLDER/bin:$DATA_FOLDER:$PATH
```

6. Config the docker-machine to recognize the shared folder, and enable creation of symbolic links within the folder:

```
# VBoxManage sharedfolder add default --name DATA --hostpath "$DATA_FOLDER" --automount
# VBoxManage setextradata default VBoxInternal2/SharedFoldersEnableSymlinksCreate/DATA 1
```

7. Restart docker-machine, ssh into the virtual machine, and mount the shared folder:

```
# docker-machine start default
# eval "$(docker-machine env default)"
# docker-machine ssh default
# mkdir –p DATA
# sudo mount -t vboxsf -o defaults,gid=`id -g docker`,uid=`id -u docker` DATA DATA
# exit
```

8. From now on, all pipeline commands should be able to access the data folder. To verify, we should be able to list the contents of the data folder by:

```
# pipe_do ls -l $DATA_FOLDER
```

Again, if you see error messages such as:

*"Error checking TLS connection: Error checking and/or regenerating the certs:*
*There was an error validating certificates for host "192.168.99.105:2376": x509: certificate is valid for 192.168.99.100, not 192.168.99.105"*

Or

*"An error occurred trying to connect: Post https://192.168.99.110:2376/v1.19/containers/create: dial tcp 192.168.99.110:2376: no route to host"*

You may solve the problem by:

```
# docker-machine regenerate-certs default
# eval "$(docker-machine env default)"
```

## Preparing data

Follow these steps to prepare data files for the pipeline:

1. Export your FlowJo workspace containing all flow samples as an xml file. Currently FlowJo version 9.x is supported. If you are running the pipeline on the provided example, the exported file is already saved to $DATA_FOLDER/cohort/T1.xml.

2. Fix the FlowJo xml file by removing Mac-encoded characters, then extract sample set information from the workspace:

```
# DATA_FOLDER="$HOME/DATA"
# export PATH="$DATA_FOLDER:$DATA_FOLDER/bin:$PATH"
# pipe_do xml.FlowJo.fix_names.pl $DATA_FOLDER/cohort/T1.xml
# pipe_do xml.FlowJo.to_flowFrame.R $DATA_FOLDER/cohort/T1.fixed.xml
```

   The list of available sample sets can be found in $DATA_FOLDER/cohort/ T1.fixed.samples.txt

3. Organize FCS files from the same sample set into one of the subfolders -- for example, use subfolder $DATA_FOLDER/cohort/FCS/2 for sample set 2.

4. Create annotations of the FCS files. Our example can be found in $DATA_FOLDER/FCS.annotations.txt.

## Running

1. Import data (the following commands import sample sets 2, 3, and 4 from our FlowJo workspace, which was generated in the previous step):

```
# 00-import.sh $DATA_FOLDER/cohort/T1.fixed.xml 2
# 00-import.sh $DATA_FOLDER/cohort/T1.fixed.xml 3
# 00-import.sh $DATA_FOLDER/cohort/T1.fixed.xml 4
```

   In our example, the imported files would be all saved in $DATA_FOLDER/cohort/T1.fixed.xml.d. For convenience, we will store the output folder name in a new variable for use in subsequent steps:

```
# export OUTPUT_FOLDER="$DATA_FOLDER/cohort/T1.fixed.xml.d"
```

2. Generate R data frames from the imported FCS files:

```
# 01-A-to_dframe.sh $OUTPUT_FOLDER  $DATA_FOLDER/cohort/FCS.annotation.txt
```

3. Generate a table of all imported files ($OUTPUT_FOLDER/all.rda.lst.txt) as well as extract FlowJo gate names by running:

```
# 01-B-export_popnames.sh $OUTPUT_FOLDER  $DATA_FOLDER/cohort/FCS.annotation.txt
# pipe_do txt.to_rda.R $OUTPUT_FOLDER/all.rda.lst.txt all.rda.lst
```

The second column in the generated tab-delimited text files ($OUTPUT_FOLDER/*/*/popnames.txt) contains the existing gate names. If the same gates have been applied to all FCS files, then these files would contain the same information.

4. [Manual Step] Use a text editor to create a tab-delimited text file (see $OUTPUT_FOLDER/TextID.txt for an example) containing a unique mapping from the extracted gate names to user-created cell subset IDs. The table should have one row for each unique gate name, and 6 columns:
   * **TextID**: ID of the cell subset. The format should be "TUBE|xxx" or "TUBE|IDxxx", where TUBE is the tube name, xxx is a unique number.
   * **Tube**: Tube name (only non-space characters are allowed), e.g. "T1".
   * **isAllAlive**: 1 if the cell subset is the topmost "viable" subset, 0 otherwise. There should be exactly one entry being "1" among all rows.
   * **isTopProperGate**: 1 if the cell subset is the topmost reference subset, 0 otherwise. There should be exactly one entry being "1" among all rows.
   * **isControl**: for internal use; set to 0.
   * **POP**: cell subset names extracted from Step 4 above.

   After creating TextID.txt, convert it to R format:
   ```
   # pipe_do txt.to_rda.R $OUTPUT_FOLDER/TextID.txt TextID
   ```

5. Update data frames and replace gate names with corresponding cell subset IDs:
   ```
   # 01-C-update_popnames_with_TextID.sh $OUTPUT_FOLDER
   ```

6. Extract basic statistics (cell count, etc.):
   ```
   # 01-D1-popstats_each.sh $OUTPUT_FOLDER
   # 01-D1a-update_popstats.sh $OUTPUT_FOLDER
   # 01-D2-popstats_collect.sh $OUTPUT_FOLDER
   # pipe_do 01-D3-popstats.to_rda.R $OUTPUT_FOLDER
   ```

7. Down-sample:
   ```
   # 01-E-downsize.sh $OUTPUT_FOLDER
   ```

8. [Manual Step] Use a text editor to create a table ($OUTPUT_FOLDER/Panels.txt) describing the flow cytometry panel. It should have the following columns:
   * **Tube**: Tube name
   * **ChannelID**: One of "FSC-A", "SSC-A", "Time", "P1", "P2", ..., "Pxxx", where "xxx" is the number of channels of the flow panel.
   * **Par.All**: For rows corresponding to P1, ..., Pxxx, put the corresponding names appear in $OUTPUT_FOLDER/*/*/popnames.txt. Otherwise the same as ChannelID.
   * **Par**: For rows corresponding to P1, ..., Pxxx, the same as Par.All. For other rows it should be empty.
   * **Par.Marker**: The same as Par, except for the "viability" channel, where it should be empty.
   * **Name**: For P1, ..., Pxxx, name of the channel as shown within the parentheses of the Par field.
   * **Marker**: Same as Name, except for the "viability" channel, where it should be empty.

After creating the file, convert it to R format:

```
# cp -i $OUTPUT_FOLDER/Panels.txt $OUTPUT_FOLDER/../
# pipe_do txt.to_rda.R $OUTPUT_FOLDER/../Panels.txt Panels
# sed -n '2,2p' $OUTPUT_FOLDER/Panels.txt | cut -f1 > $OUTPUT_FOLDER/Tube
```

9. Generate $DATA_FOLDER/cohort/data.list.rda:

```
# data.list.all.rda.txt.sh $OUTPUT_FOLDER
# pipe_do txt.to_rda.R $OUTPUT_FOLDER/../data.list.all.rda.txt all.rda
# mv -i $OUTPUT_FOLDER/../data.list.all.rda.rda $OUTPUT_FOLDER/../data.list.rda
```

10. Pool samples and compute global scale factor:

```
# pipe_do 03-A-1-pooled_sample_alive_pop.R $OUTPUT_FOLDER
# pipe_do 03-A-2-global_scale.R $OUTPUT_FOLDER
```

11. Compute statistics:

```
# 05-A-rda.flowFrame.stats01.pl $OUTPUT_FOLDER # may take some time
# 05-D-rda.flowFrame.collect.pl $OUTPUT_FOLDER
# pipe_do 05-E-rda.stats.to_rda.R $OUTPUT_FOLDER
```

12. Generate heterogeneity parameters:

```
# pipe_do HM.R $OUTPUT_FOLDER
# pipe_do 10-A-make_metrics_table.R $OUTPUT_FOLDER
# pipe_do 10-B1-SD_and_globally_scaled_SumSD.R $OUTPUT_FOLDER
```

The computed heterogeneity parameters are saved to $OUTPUT_FOLDER/10-B1-eset_combined-dt.rda.