ReadMe for the R scripts required to prepare the data for the randomizer script

This tutorial uses publicly available data from two studies ([1] and [2]) available via The Single Cell Portal (https://singlecell.broadinstitute.org/single_cell)

Some steps during this pipeline require your attention, others will run automatically (and might take some time – depending on your computer and the size of the data set you chose).

To get the most out of this tutorial/semi-automated pipeline, we recommend creating a new directory on your hard drive and recreating the directory structure used in this tutorial.

After some steps it is required to create a new R project, this is done for practical reasons as you might need to perform some operations more often than others (details below).

In our pipeline, we imported two different data sets in R.

1. Data from the study “A single-cell atlas of human and mouse white adipose tissue”, which is available as RDS-file at https://singlecell.broadinstitute.org/single_cell/study/SCP1376/a-single-cell-atlas-of-human-and-mouse-white-adipose-tissue

2. Data from the study “Deep learning enables genetic analysis of the human thoracic aorta”, which is available as H5AD-file at https://singlecell.broadinstitute.org/single_cell/study/SCP1265/deep-learning-enables-genetic-analysis-of-the-human-thoracic-aorta

---

1. Emont, M.P., et al., *A single-cell atlas of human and mouse white adipose tissue.* Nature, 2022. **603**(7903): p. 926-933.

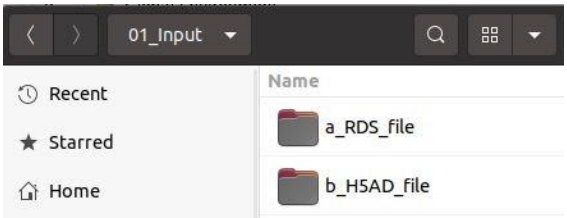
2. Pirruccello, J.P., et al., *Deep learning enables genetic analysis of the human thoracic aorta.* Nature Genetics, 2022. **54**(1): p. 40-51.

ATTENTION:

If you would like to analyze the RDS-file [1], be warned:

- The file is big and will become even bigger after importing in R
    - You will need sufficient disk space
    - And more importantly sufficient RAM (and virtual RAM / swap)
    - If you're using Linux and your R session aborts during the workflow it is most likely due to insufficient RAM.
      To fix this:
        - Create a bigger swap file (a virtual machine with Linux and 8 GB RAM 100 GB swap (80% swappiness) worked fine but you can also try smaller sizes and just increase the swap if it still aborts)
    - If you would like to try the tutorial but don't have much time/disk space, we recommend using two subsets of the H5AD dataset [2]

- **Create a directory 01_Input**
    - If you're interested in using RDS-files,
      create the sub directory **a_RDS_file** (optional)
        - This part of the tutorial will end after importing the RDS-file
        - The variable created will have the same name as the variable created when importing the H5AD-file (hence, 2 R projects), thus, you can continue the workflow with your RDS-file BUT you will have to adjust some names
    - If you're interested in using H5AD-files,
      create the sub directory **b_H5AD_file** (required)
        - This part of the tutorial will import the H5AD-file available at the Single Cell Portal, create subsets and use two of the subsets for the workflow



Importing an RDS-file:

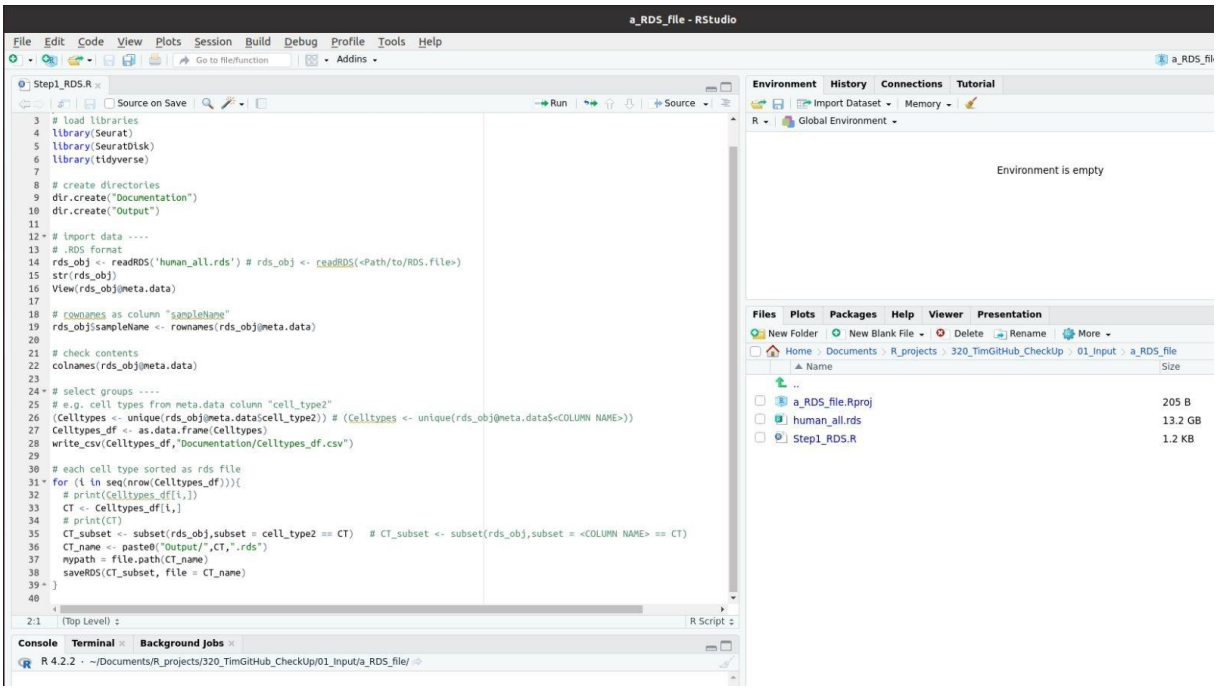
- Create the subdirectory directory **a_RDS_file**
- Copy the script Step1_RDS.R in the subdirectory
- Download the RDS file **human_all.rds** from https://singlecell.broadinstitute.org/single_cell/study/SCP1376/a-single-cell-atlas-of-human-and-mouse-white-adipose-tissue and save it in the subdirectory
- Create a new R project in the subdirectory
- Open the script in R studio and run it

Required Packages:

- Seurat https://satijalab.org/seurat/articles/install.html
- SeuratDisk https://github.com/mojaveazure/seurat-disk
- tidyverse https://tidyverse.tidyverse.org/

What this script will do:

- Import the RDS-file
- Create subsets (by cell type)
- Save these subsets as RDS-files in the **Output** directory



Importing an H5AD-file (and the rest of the workflow)

- Create the subdirectory directory **b_H5AD_file**
- Copy the script Step1_RDS.R in the subdirectory
- Download the H5AD file **ascending_descending_human_aorta_v1.h5a**d from https://singlecell.broadinstitute.org/single_cell/study/SCP1265/deep-learning-enables-genetic-analysis-of-the-human-thoracic-aorta and save it in the subdirectory
- Create a new R project in the subdirectory
- Open the script in R studio and run it

Required Packages:

- Seurat https://satijalab.org/seurat/articles/install.html
- SeuratDisk https://github.com/mojaveazure/seurat-disk
- tidyverse https://tidyverse.tidyverse.org/

What this script will do:

- Prepare the H5AD file for import in R
- Import the Seurat object ascending_descending_human_aorta_v1.h5Seurat
- Create subsets according to cell type (the column “cell_type_leiden”) and save them in the Output directory

If you would like to save space and time, you don’t need to create all of the subsets, “01. Fibroblast I” and “02. Macrophage“ are sufficient.

Part IIa – Preparing the Data

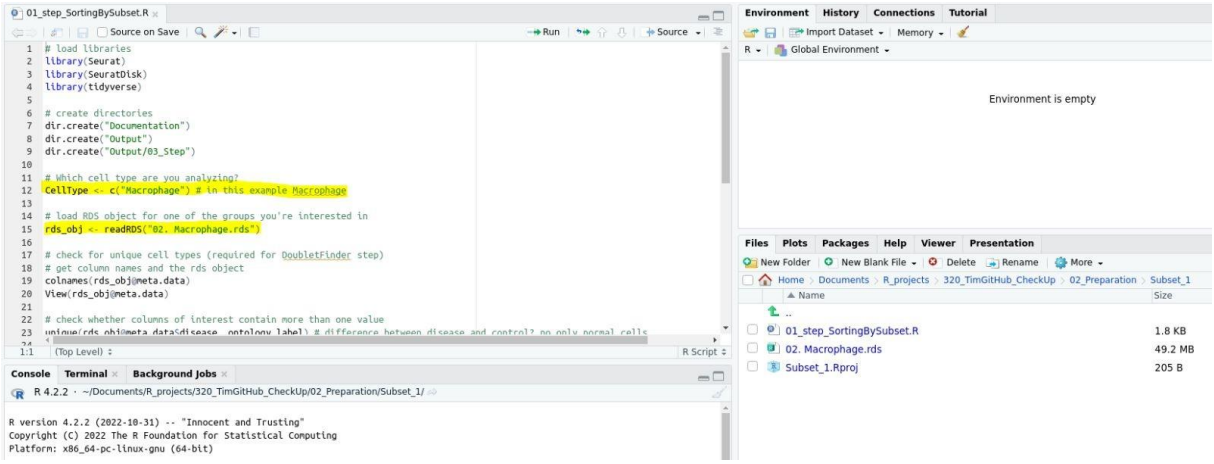
- Create the directory **02_Preparation** and subdirectories for each subset/group you would like to analyze
  (e.g., Fibroblast I and Macrophage)
- Copy the script **01_step_SortingBySubset.R** into each of the subdirectories
- Copy the respective RDS-files into the subdirectories
- Create new R projects for each group
- E.g., if you plan to analyze Fibroblast I and Macrophage:
  - Create the subdirectory Subset_1 and an R project in the directory
  - Create the subdirectory Subset_2 and an R project

Required Packages:

- Seurat https://satijalab.org/seurat/articles/install.html
- SeuratDisk https://github.com/mojaveazure/seurat-disk
- tidyverse https://tidyverse.tidyverse.org/

Manual changes in the script:

- Line 12: You need to **name the cell type** you are using and have to **change the variable CellType** accordingly
- Line 15: You need to use the name of the subset you are planning to analyze
- Run the script



Part IIa – Preparing the Data

What this script will do:

- Create subsets of the groups that are going to be analyzed (in the Output directory)
- This is necessary for DoubletFinder analysis in case the groups contain subsets
    - E.g., the group “02. Macrophage“ contains 2 kinds of macrophages (“ascending aorta” and “descending aort”), see unique(rds_obj@meta.data$organ__ontology_label)
    - Thus the group “02. Macrophage“ will be split into 2 subsets, resulting in 3 directories:
        - 03_Step
        - Macrophage_ascending aorta
        - Macrophage_descending aorta

This step needs to be repeated for every group that is going to be analyzed (e.g., in Subset_1 and in Subset_2)

The next steps are performed in the “named subdirectories” which will be created in the Output directory,
e.g., for Macrophage:

- Macrophage_ascending aorta
- Macrophage_descending aorta

Part IIb – Preparing the Data

- Create a new R project in the respective subset
- Copy the **script 02_step_RemovingDoublets.R** into each of the named subdirectories, e.g., for Macrophage:
    - Macrophage_ascending aorta
    - Macrophage_descending aorta

Required Packages:

- Seurat https://satijalab.org/seurat/articles/install.html
- SeuratDisk https://github.com/mojaveazure/seurat-disk
- tidyverse https://tidyverse.tidyverse.org/
- DoubletFinder https://github.com/chris-mcginnis-ucsf/DoubletFinder

Manual changes in the script:

- Line 13: You need to **name the cell type** you are using and have to **change the variable Input_name** accordingly
- Line 14: You need to set the variable for the group label
    - **as_Number** can be **0** or **1**
    - **Subsets of the same group require the same label number** (e.g., Macrophage as 0 and Fibroblast as 1)
- Line 19: You need to use the name of the subset you are planning to analyze
- Run the script

```
11 # # import data ----
12 # read in
13 Input_name <- c("Macrophage_asc") # the name will be used for the output file
14 as_Number <- 0 # one group has to be labeled as "0", the other as "1",
15 # if a group contains subgroups, the sub groups need to have the same label number
16 # e.g. Macrophage contains cells from the ascending aorta and the descending aorta
17 # group Macrophage labeled as "0"
18 # => both sub groups "Macrophage_ascending aorta" and "Macrophage_descending aorta" need to be labeled as "0"
19 Input_RDS <- readRDS("Macrophage_ascending aorta.rds") # name / path to subgroup created in the step before
```

What this script will do:

- Prepare the data for DoubletFinder analysis
- Perform DoubletFinder analysis
- Remove doublets
- **Create a table in the directory "03_Step"** containing
    - the relevant information for the subsequent analyses of our workflow
    - a label (0 or 1)
- Repeat this for every group-subset
  e.g., for Macrophage (labeled as "0" after running the script):
        - Macrophage_ascending aorta
        - Macrophage_descending aorta

e.g., also for the subdirectories of the Fibroblast I group (labeled as "1" after running the script)

Remember to check and adjust the variables!

```
11  # # import data ----
12  # read in
13  Input_name <- c("Fibroblast_asc") # the name will be used for the output file
14  as_Number <- 1 # one group has to be labeled as "0", the other as "1",
15  # if a group contains subgroups, the sub groups need to have the same label number
16  # e.g. Macrophage contains cells from the ascending aorta and the descending aorta
17  # group Macrophage labeled as "0"
18  # => both sub groups "Macrophage_ascending aorta" and "Macrophage_descending aorta" need to be labeled as "0"
19  Input_RDS <- readRDS("Fibroblast_ascending aorta.rds") # name / path to subgroup created in the step before
```

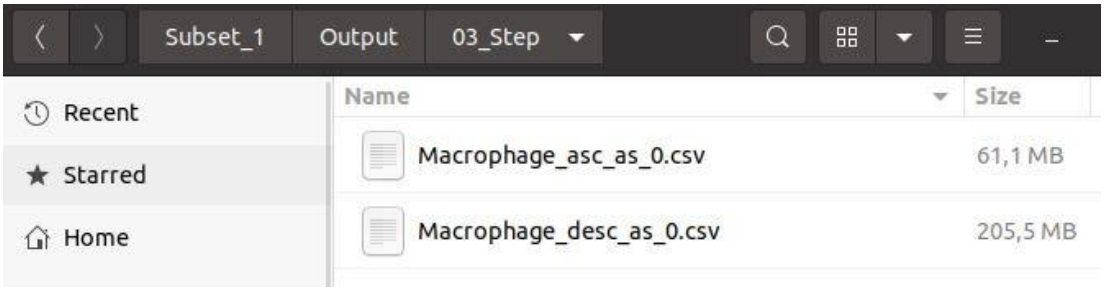
You might also need to adjust the filter (around line 32). In case "Input_RDS_filtered is not found, it might be due to different column names in the InputRDS@meta.data

```
# filter out low quality cells ----
Input_RDS_filtered <- subset(Input_RDS, subset = nCount_RNA > 800 &
                               nFeature_RNA > 500 &
                               percent_mito < 5)

# Input_RDS_filtered <- subset(Input_RDS, subset = <nCount_RNA or the respective name from the meta.data> > 800 &
#                                <nFeature_RNA or the respective name from the meta.data> > 500 &
#                                <percent_mito or the respective name from the meta.data> > < 5)
```

percent_mito is sometimes also named mt.percent (e.g. in the human WAT atlas)

Afterward, you will find a table for each subset of the group in the **03_Step directory**, e.g., for Macropage:



Part IIc – Preparing the Data

- Copy the file 03_Step_joiningSubGroups.R into the 03_Step directory,
- create an R project,
- and adjust the name variables

Manual changes in the script:

- Line 5: To import the respective subset, you need **to read in the correct CSV-file** of the first subset
- Line 6: To import the respective subset, you need to **read in the correct CSV-file** of the second subset
- Line 8: **Change the variable name** accordingly (so that you will later recognize the subset imported in line 5)
- Line 9: **Change the variable name** accordingly (so that you will later recognize the subset imported in line 6)
- Run the script
-

```
# joining the sub groups, assuming a total of 2 sub groups
SubGroup_1 <- read.csv("Macrophage_asc_as_0.csv", header = FALSE) # the first sub group: SubGroup_1 <- read.csv(<name of first sub group.CSV>, header = FALSE)
SubGroup_2 <- read.csv("Macrophage_desc_as_0.csv", header = FALSE) # the second sub group: SubGroup_2 <- read.csv(<name of second sub group.CSV>, header = FALSE)

SubGroup_1_name <-c("Macrophage_asc_as_0") # SubGroup_1_name <- c("<NAME of first sub group>", header = FALSE)
SubGroup_2_name <- c("Macrophage_desc_as_0") # SubGroup_2_name <- c("<NAME of second sub group>", header = FALSE)
```

And repeat the same steps for the subsets of the second group

```
# joining the sub groups, assuming a total of 2 sub groups
SubGroup_1 <- read.csv("Fibroblast_asc_as_1.csv", header = FALSE) # the first sub group: SubGroup_1 <- read.csv(<name of first sub group.CSV>, header = FALSE)
SubGroup_2 <- read.csv("Fibroblast_desc_as_1.csv", header = FALSE) # the second sub group: SubGroup_2 <- read.csv(<name of second sub group.CSV>, header = FALSE)

SubGroup_1_name <-c("Fibroblast_asc_as_1") # SubGroup_1_name <- c("<NAME of first sub group>", header = FALSE)
SubGroup_2_name <- c("Fibroblast_desc_as_1") # SubGroup_2_name <- c("<NAME of second sub group>", header = FALSE)
```

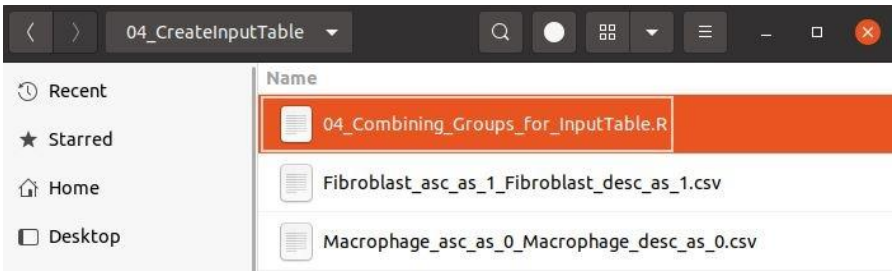
What this script will do:

- Create a labeled (0 and 1) table for the two groups that will be compared in the subsequent analyses
- Put the respective tables into the directory **04_CreateInputTable**

Part III – Creating the Input Table

- After performing the previous steps, you should find the labeled and joined tables for both groups in **04_CreateInputTable**

1. Table for the first group (labeled and both subgroups joined)
2. Table for the second group (labeled and both subgroups joined)



- Create a new R project in this directory and adjust the script **04_Combining_Groups_for_InputTable.R** (if needed)

Manual changes in the script:

- Line 9: To import the respective subset, you need **to read in the correct CSV-file** of the first group
- Line 10: To import the respective subset, you need to **read in the correct CSV-file** of the second group
- Line 13: **Change the variable name** accordingly (so that you will later recognize the group imported in line 9)
- Line 14: **Change the variable name** accordingly (so that you will later recognize the group imported in line 10)
- Run the script and continue the workflow using the **resulting table** (in **Output**) as **input for the Randomizer script**