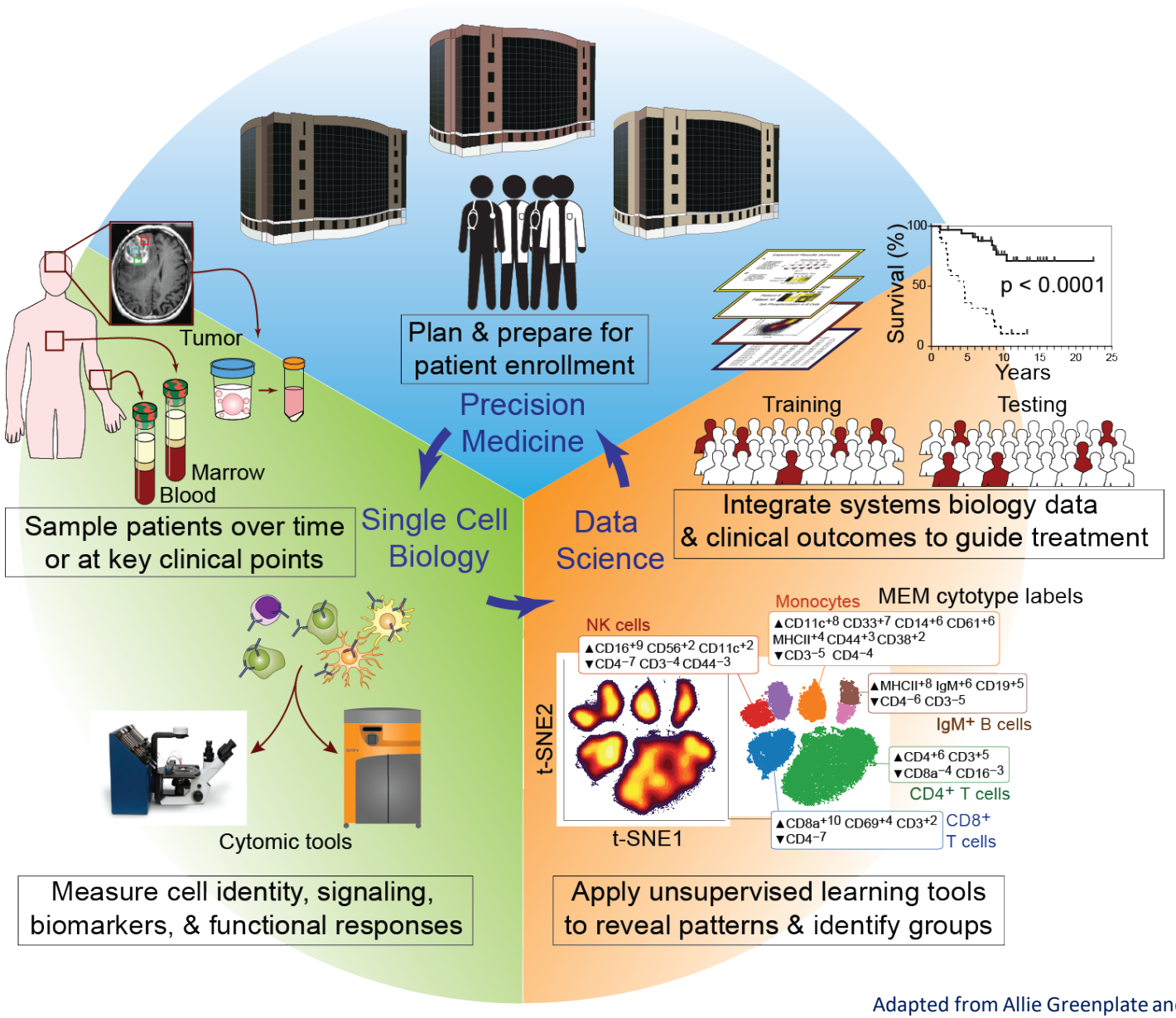
Data Science Tools for Cancer Biologists

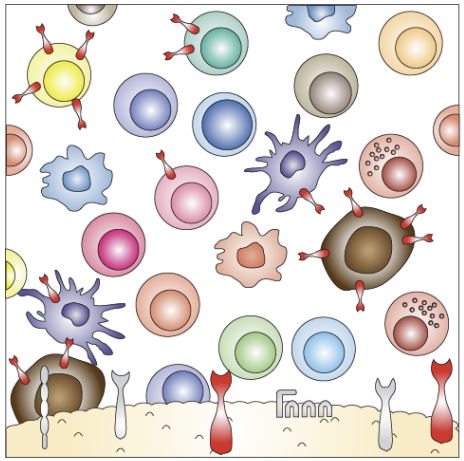
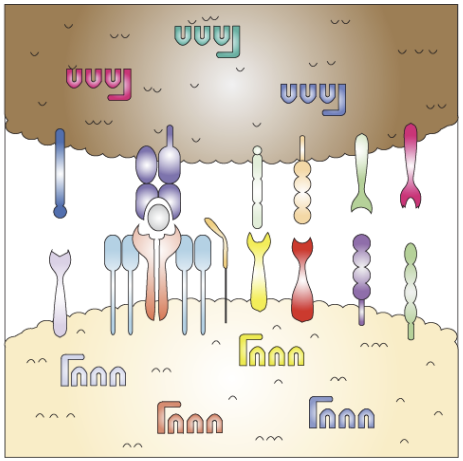
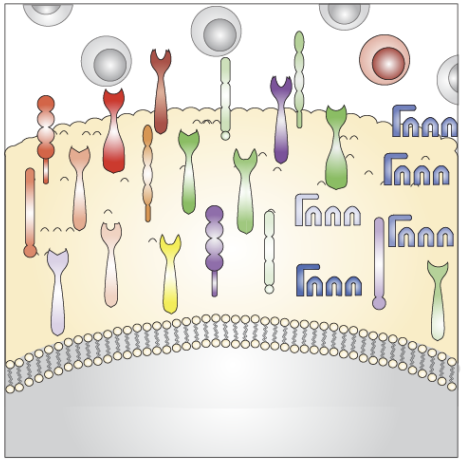
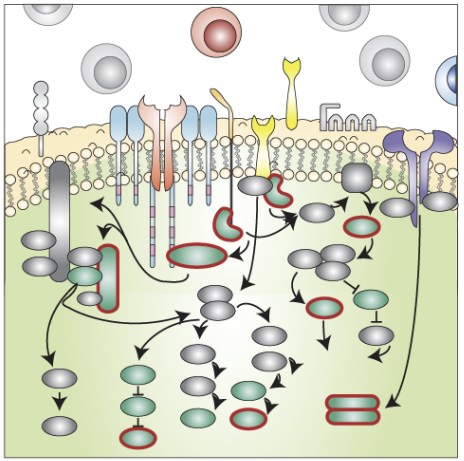
Sierra Barone
Justine Sinnaeve
8/22/2019

Goal: Systematically Dissect Cellular Mechanisms Across Time, Treatments, Tissues, & Tumor Types



Adapted from Allie Greenplate and Kirsten Diggins

Applications of Mass Cytometry in Cancer Biology

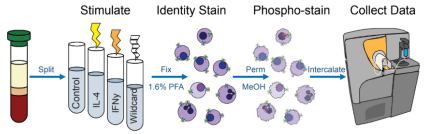
Microenvironment	Cell:cell interactions	Immunophenotype	Signaling & function
			
Track key biomarkers on all cell types; find cytokine producers	Monitor immune checkpoint proteins on T cells, APCs, & cancer cells	Measure differentiation; deep phenotype using fewer cells	Dissect signaling changes; characterize mechanisms of treatment response

Greenplate et al., *Euro J Cancer* 2016

Single Cell Biology Workflow

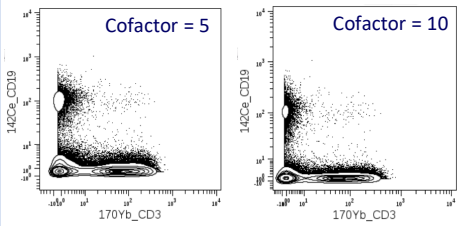
Data collection

1) Panel design
2) Data collection



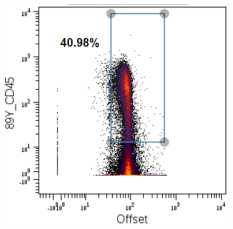
Data processing

3) Cell event parsing
4) Scale transformation



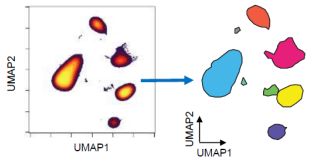
Distinguishing initial populations

5) Live single cell gating
6) Focal population gating



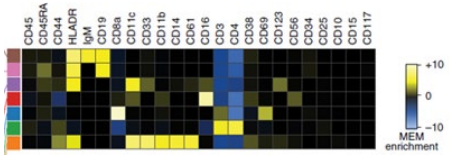
Revealing cell subsets

7) Feature selection
8) Dimensionality reduction
9) Identify cell clusters
10) Cluster refinement



Characterizing cell subsets

11) Feature comparison
12) Model populations
13) Learn cell identity
14) Statistical testing



How much can be automated?

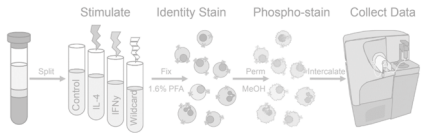
How do we select tools and use them well?

Workflow Steps Adapted from Diggins et al., *Methods* 2015

Single Cell Biology Workflow

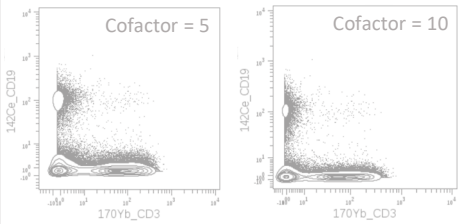
Data collection

1) Panel design
2) Data collection



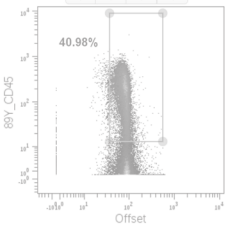
Data processing

3) Cell event parsing
4) Scale transformation



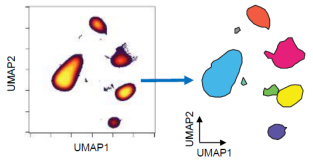
Distinguishing initial populations

5) Live single cell gating
6) Focal population gating



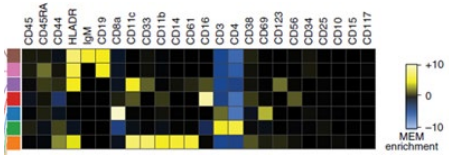
Revealing cell subsets

7) Feature selection
8) Dimensionality reduction
9) Identify cell clusters
10) Cluster refinement



Characterizing cell subsets

11) Feature comparison
12) Model populations
13) Learn cell identity
14) Statistical testing



How much can be automated?

How do we select tools and use them well?

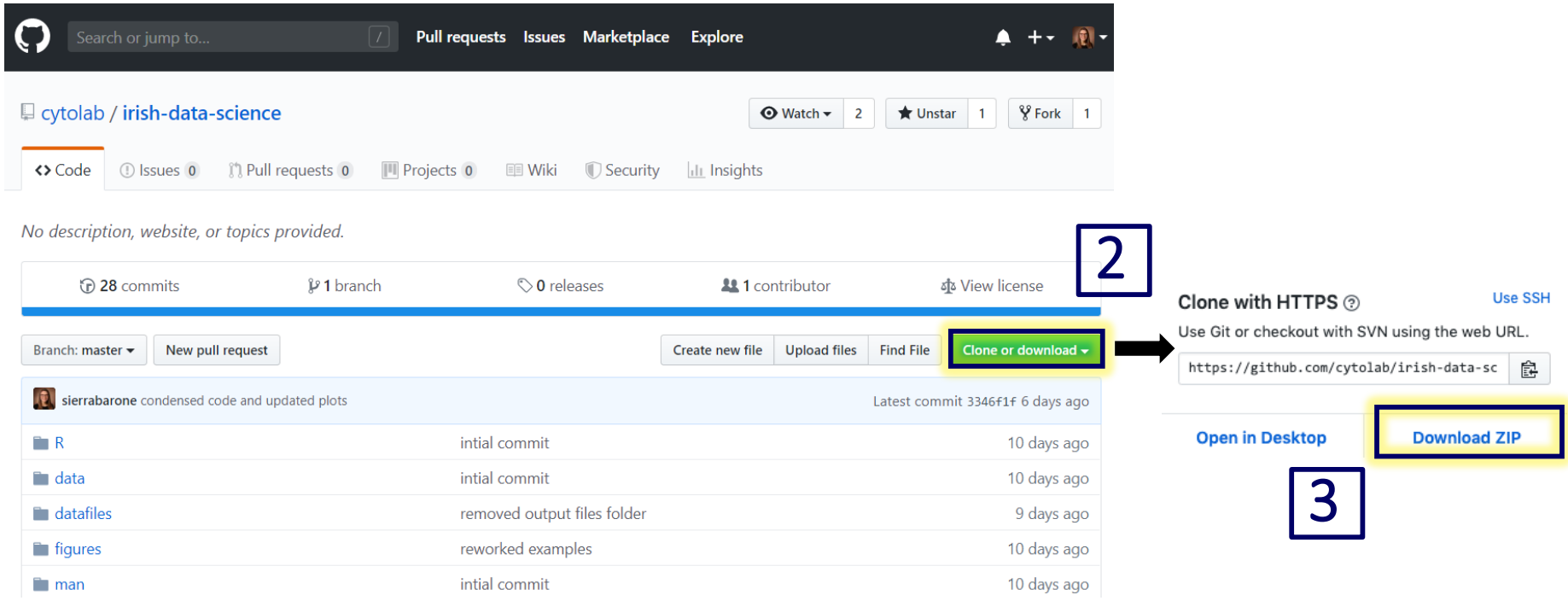
Workflow Steps Adapted from Diggins et al., *Methods* 2015

Installation and Intro to Working in R

Download scripts, data, and R packages from GitHub

1 Go to link below and download repository:

https://github.com/cytolab/irish-data-science



Irish-data-science repository contents

1 installation script (R markdown, .rmd)

2 example analysis scripts (.rmd)

Data files (.fcs)

MEM package (.r, .rproj, etc.)

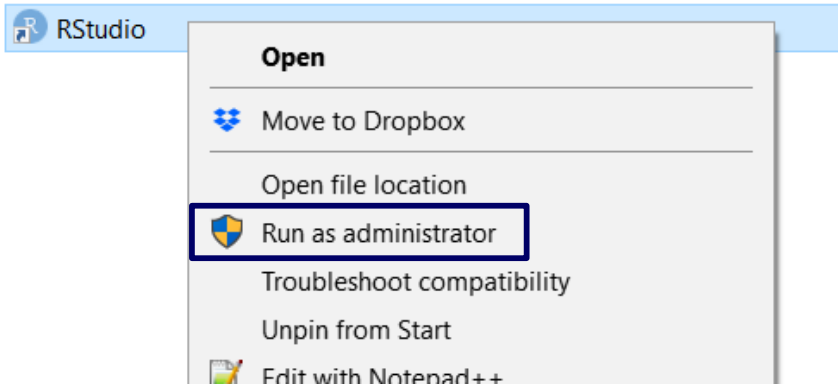
Documentation files (.rd, .md)

Other misc. files (.txt, .pdf, .rdata, etc.)

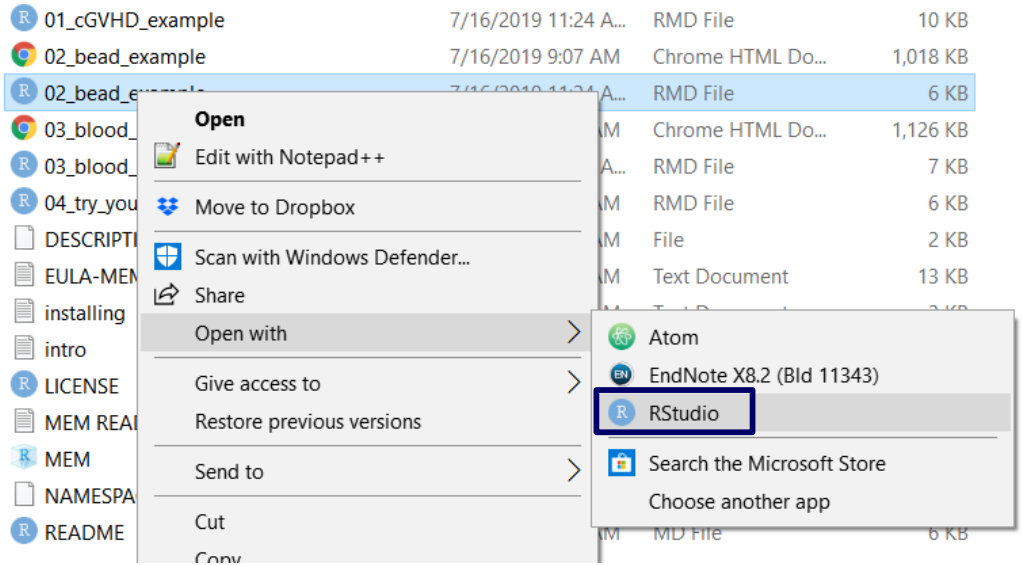
*make sure you unzip downloaded folder

RStudio

For PC users, run RStudio as administrator



For all, open .Rmd files with RStudio



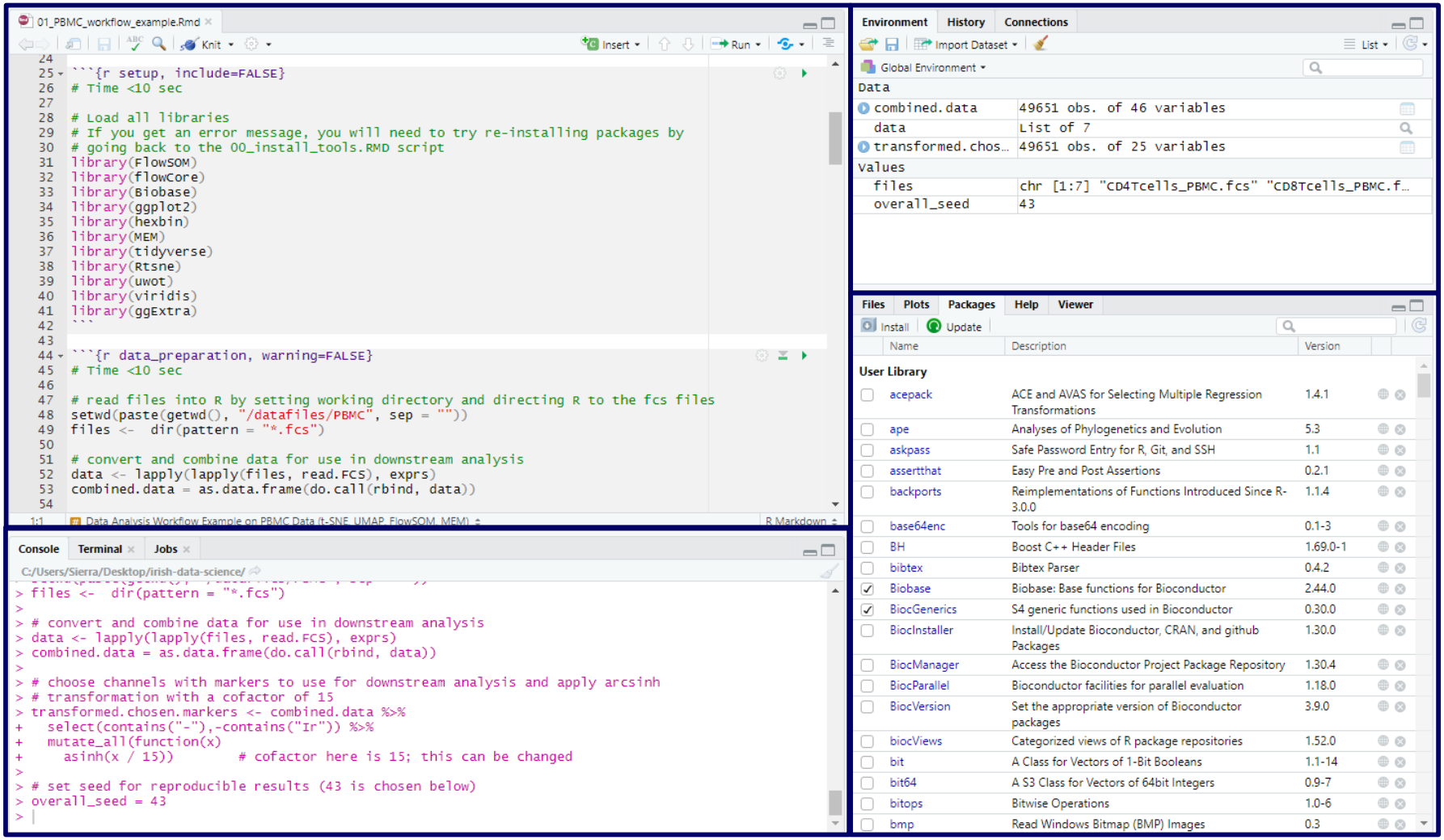
*make sure you unzip downloaded folder

Open 00_install_tools.rmd and 01_PBMC_workflow_example.rmd

*make sure console is open

Working Script and Code

Environment

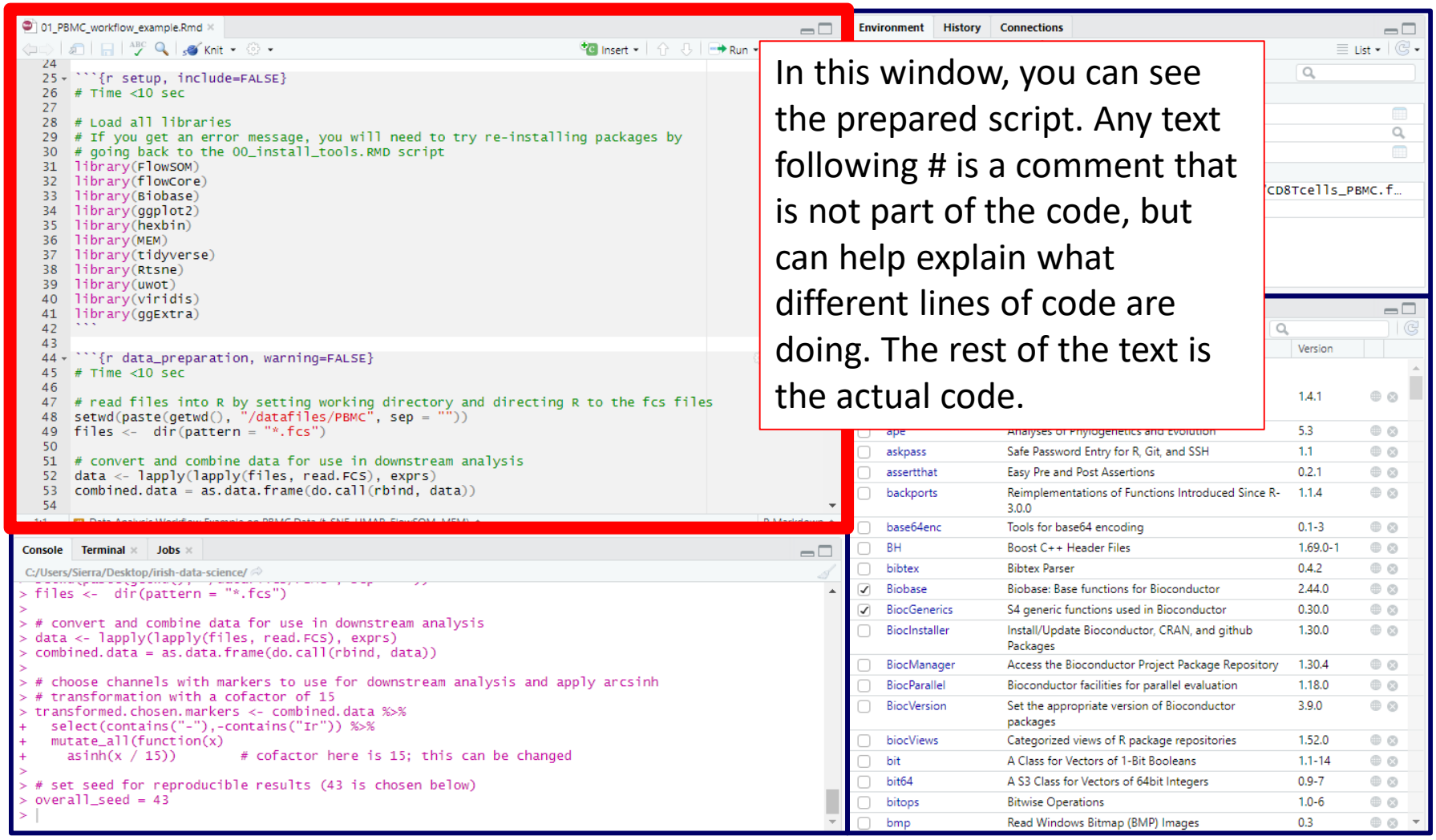


Console

Plots, Files, Help, etc.

Working Script and Code

Environment



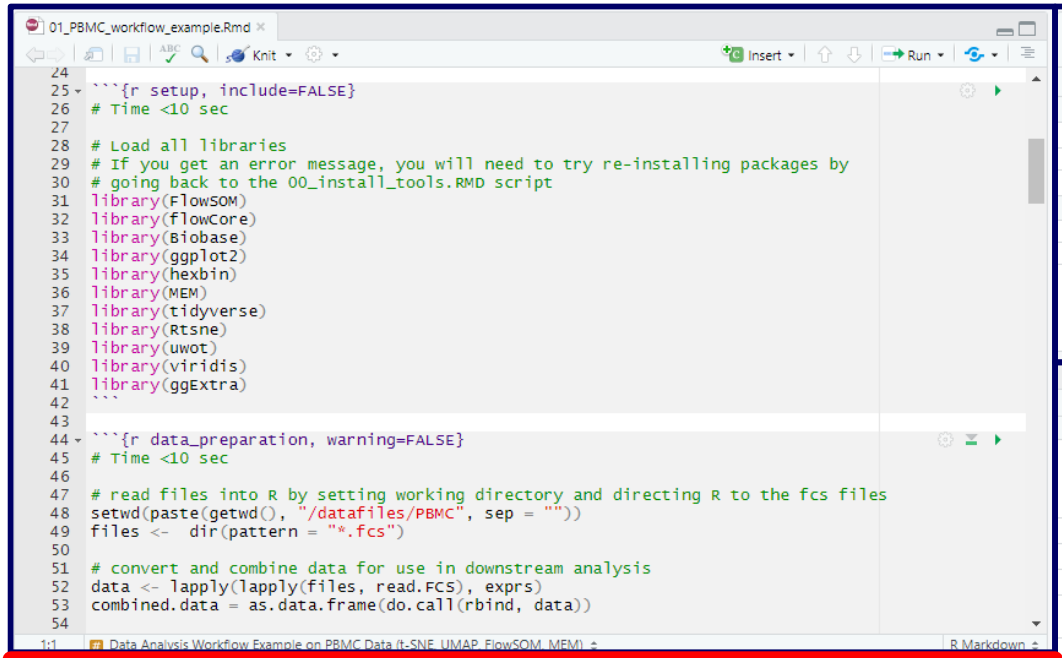
In this window, you can see the prepared script. Any text following # is a comment that is not part of the code, but can help explain what different lines of code are doing. The rest of the text is the actual code.

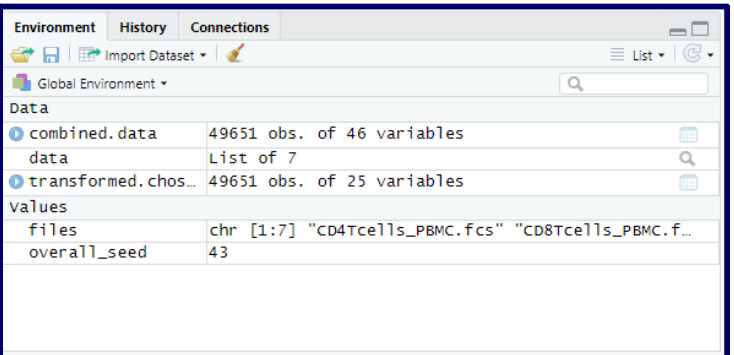
Console

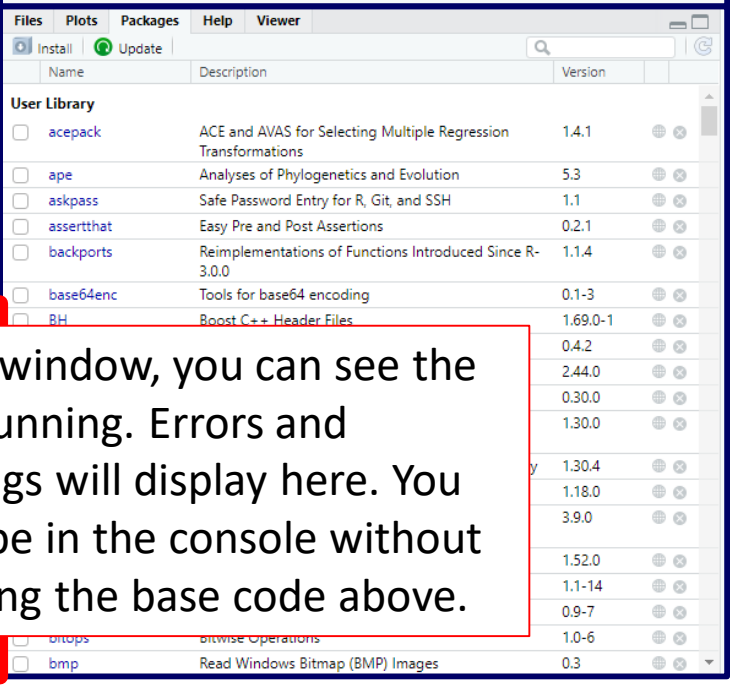
Plots, Files, Help, etc.

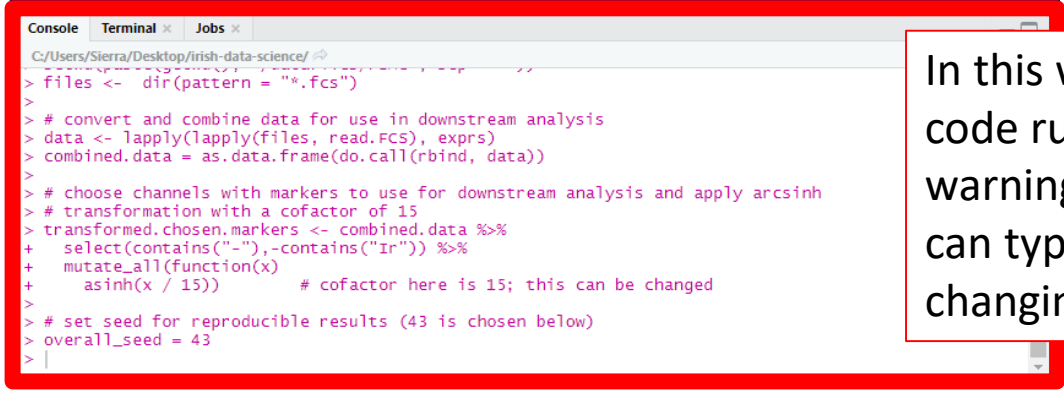
Working Script and Code

Environment









In this window, you can see the code running. Errors and warnings will display here. You can type in the console without changing the base code above.

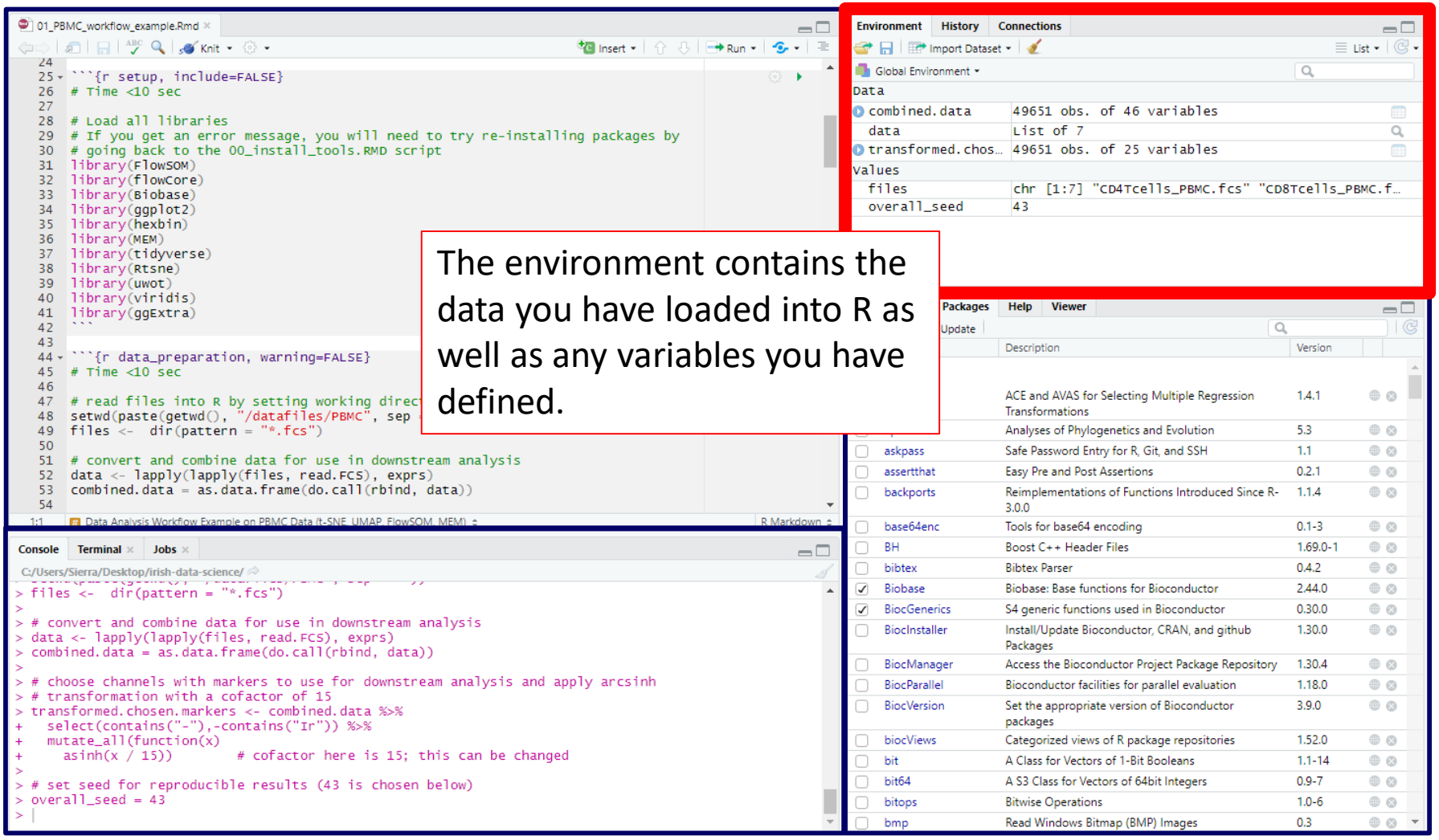
Console

Plots, Files, Help, etc.

*make sure console is open

Working Script and Code

Environment



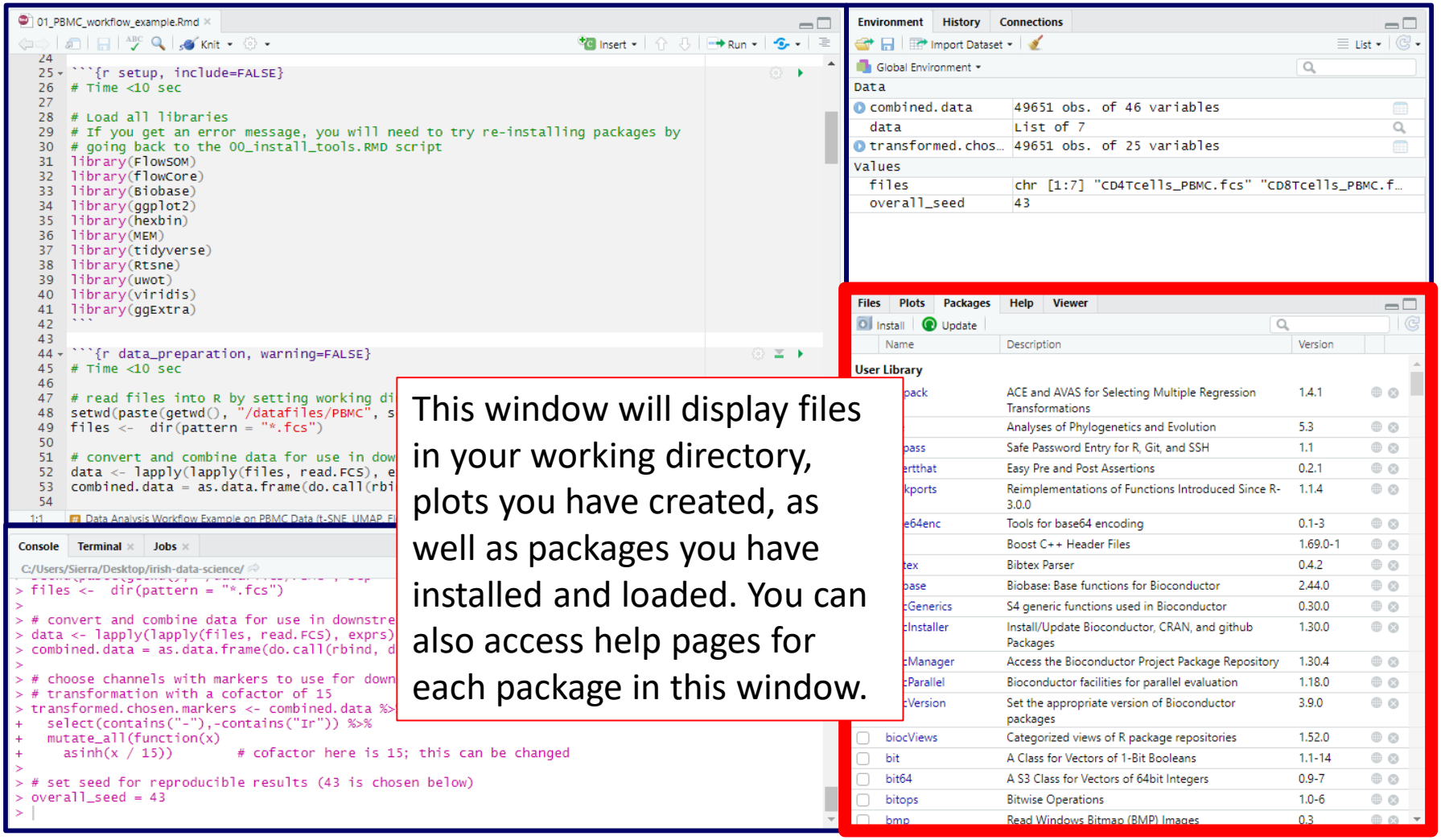
The environment contains the data you have loaded into R as well as any variables you have defined.

Console

Plots, Files, Help, etc.

Working Script and Code

Environment

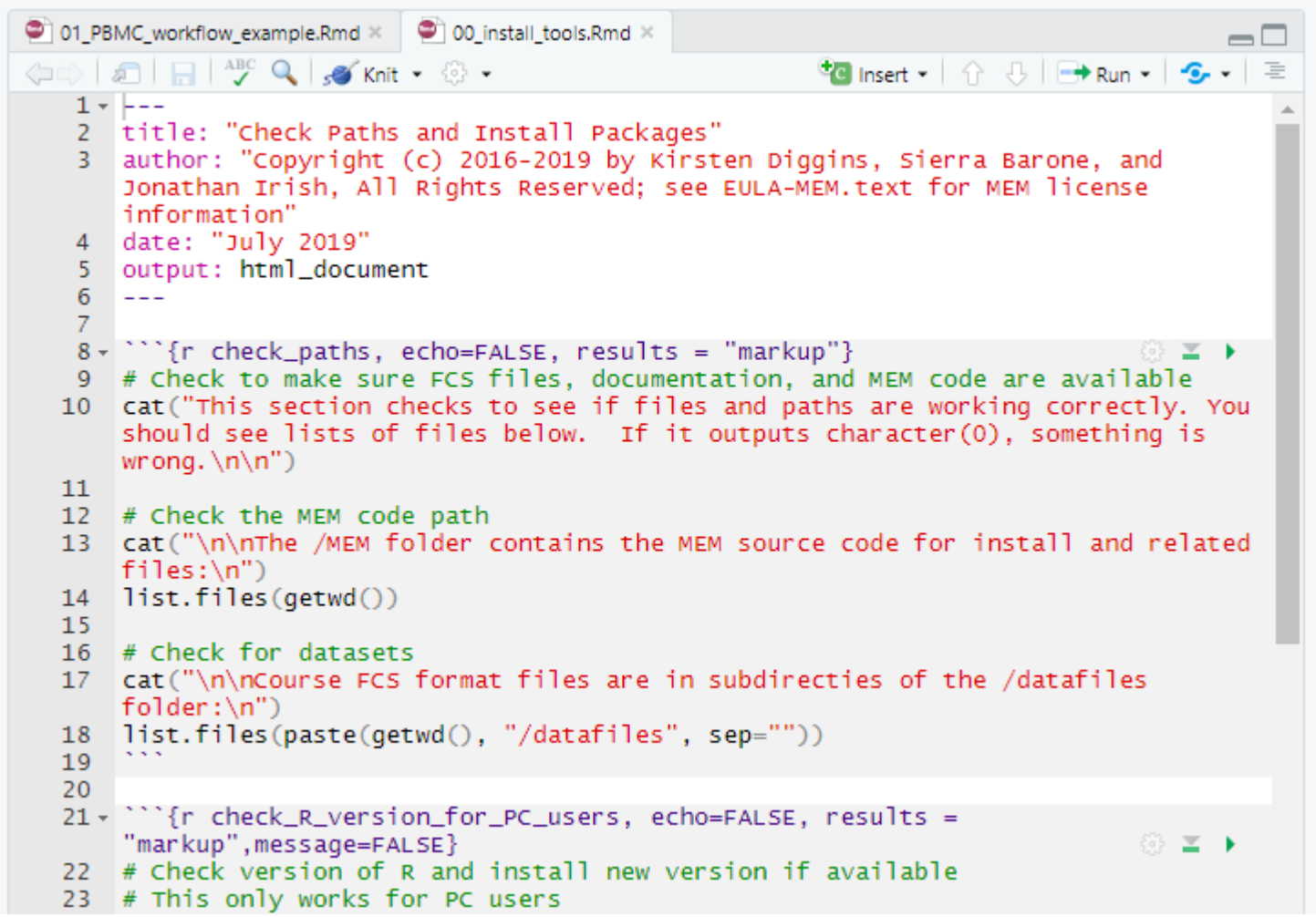


Console

Plots, Files, Help, etc.

Open **00_install_tools.rmd**

1



Header

2

Code

3

Open **00_install_tools.rmd** and begin installing required packages

CNTL-ENTER or COMMAND-RETURN to run a single line of code

OR

Press play to run entire section of code

Code Section Title

```
```{r check_paths, echo=FALSE, results = "markup"}
# Check to make sure FCS files, documentation, and MEM code are available
cat("This section checks to see if files and paths are working correctly. You
should see lists of files below.  If it outputs character(0), something is
wrong.\n\n")

# Check the MEM code path
cat("\n\nThe /MEM folder contains the MEM source code for install and related
files:\n")
list.files(getwd())

# Check for datasets
cat("\n\nCourse FCS format files are in subdirecties of the /datafiles
folder:\n")
list.files(paste(getwd(), "/datafiles", sep=""))
```
```

This section checks that the files we will need are accessible in our working directory

```
```{r installation_notes, echo=FALSE, results = "markdown"}
# Print the contents a help file that explains installing packages
writeLines(readLines(paste(getwd(), "installing.txt", sep="/")))
```
```

This section prints installation text

Open **00_install_tools.rmd** and begin installing required packages

```{r install_bioconductor_packages, echo=FALSE, results = "hide"}
# install bioconductor and flow cytometry tools for R
cat("If this works, you should see 4 sets of messages about downloading files
that end in a message saying something like package 'BiocManager' successfully
unpacked and MD5 sums checked.  You should see this for BioCManager, Biobase,
flowCore, and FlowSOM.\n\n")
install.packages("BiocManager", repos = "http://cran.us.r-project.org")

if (!requireNamespace("BiocManager", quietly = TRUE))
    install.packages("BiocManager")
BiocManager::install("flowCore")
BiocManager::install("FlowSOM")
```

This section downloads Bioconductor and flow cytometry tools we will need

```{r test_flow_installs, echo=FALSE, results = "markdown"}
# Load and test whether bioconductor and flow packages are installed
cat("If this works, you may see Attaching Package messages or no message at
all; that's good.  If you get a warning, go back to the last CHUNK.\n\n")
library(FlowSOM)
library(flowCore)
library(Biobase)
```

This section tests to make sure Bioconductor and flow cytometry tools are installed

```{r install_ggplots, echo=FALSE, results = "markup"}
# install plotting packages
cat("If this works, you will see text about packages being downloaded.\n\n")
install.packages("gplots", repos = "http://cran.us.r-project.org")
install.packages("ggplot2", repos = "http://cran.us.r-project.org")
install.packages("hexbin", repos = "http://cran.us.r-project.org")
install.packages("viridis", repos = "http://cran.us.r-project.org")
install.packages("ggExtra", repos = "http://cran.us.r-project.org")
```

```{r load_gplots, echo=FALSE, results = "markup"}
# Load and test whether gplots and ggplot2 packages are installed
cat("If this works, you may see Attaching Package messages or no message at
all; that's good.  If you get a warning, go back to the last CHUNK.\n\n")
library(gplots)
library(ggplot2)
library(hexbin)
library(viridis)
library(ggExtra)
```

The next sections install and load the tools to make plots

Open **00_install_tools.rmd** and begin installing required packages

```{r install_MEM, echo=FALSE, results = "markup"}
# install MEM, load it, and test if it is all set
cat("If this works, you should see several lines about installing files, then
DONE (MEM) near the end.  The MEM help page will also open in the Help menu in
RStudio.\n\n")

# If you have previously installed MEM, you may get an error message. If this
is the case, try restarting your RStudio session
install.packages(getwd(), type="source", repos=NULL)
library(MEM)
?MEM

# OR
# install.packages("devtools", repos = "http://cran.us.r-project.org")
# devtools::install_github("cytolab/mem")
```

This section installs and loads the marker enrichment modeling tool

```{r install_last_packages, echo=FALSE, results = "markup"}
# install the last packages for UMAP, t-SNE and other tools
print("You may see a bunch of messages, this is OK as long as they are not
errors.\n\n")
install.packages("tidyverse", repos = "http://cran.us.r-project.org")
install.packages("Rtsne", repos = "http://cran.us.r-project.org")
install.packages("uwot", repos = "http://cran.us.r-project.org")
install.packages("RColorBrewer", repos = "http://cran.us.r-project.org")
```

```{r load_last_packages, echo=FALSE, results = "markup"}
# Load and test the last libraries
library(tidyverse)
library(Rtsne)
library(uwot)
library(RColorBrewer)
```

These sections install and load the other tools we will use for analysis

Open **01_PBMC_workflow_example.rmd**
and work through the example

01_PBMC_workflow_example.rmd

01_PBMC_workflow_example.Rmd

Knit | Insert | Run

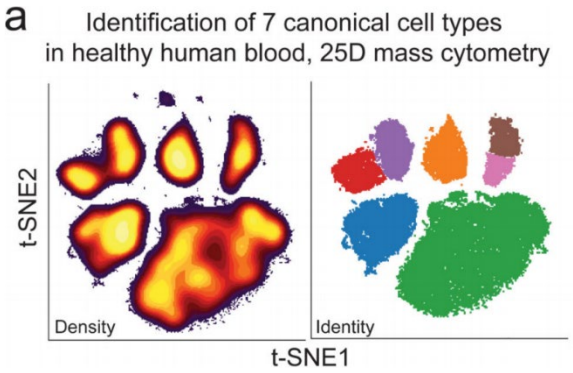
```
---
title: "Data Analysis Workflow Example on PBMC Data (t-SNE, UMAP, FlowSOM, MEM)"
author: "Copyright (c) 2016-2019 by Kirsten Diggins, Sierra Barone, and Jonathan Irish, All Rights Reserved; see EULA-MEM.text for MEM license information"
date: "July 2019"
output:
  pdf_document:
    latex_engine: xelatex
  html_document:
    df_print: paged
editor_options:
  chunk_output_type: inline
---

This data set contains 7 FCS (flow cytometry standard) files. Each FCS file contains single cell data for one cell subset that is a well-established, phenotypically distinct population. This is mass cytometry data for healthy human PBMC (peripheral blood mononuclear cells). The populations were expert gated following a t-SNE analysis. The first section of the code will run two dimensionality reduction tools, UMAP and t-SNE, on the data set. Next, you will run FlowSOM on the UMAP axes to cluster the islands of cell populations. Finally, you will run MEM to see enrichment scores for each of the FlowSOM clusters. The goal of this exercise is to run several computational tools on a single cell data set to get a feel for the workflow used in the Irish lab.

```{r setup, include=FALSE}
# Time <10 sec

# Load all libraries
# If you get an error message, you will need to try re-installing packages by
# going back to the 00_install_tools.RMD script
library(FlowSOM)
library(flowCore)
library(Biobase)
library(ggplot2)
library(hexbin)
library(MEM)
library(tidyverse)
library(Rtsne)
library(uwot)
library(viridis)
library(ggExtra)
```
```

A description of the code and its purpose



This section loads the necessary libraries

Diggins et al., *Nature Methods* 2017

01_PBMC_workflow_example.rmd

Data Preparation

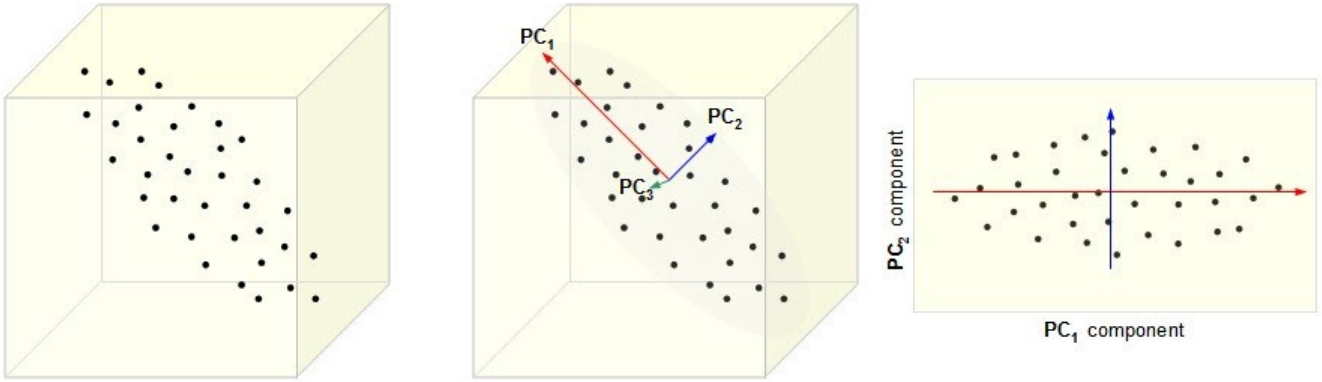
```
43
44 ```{r data_preparation, warning=FALSE}
45 # Time <10 sec
46
47 # read files into R by setting working directory and directing R to
   files
48 setwd(paste(getwd(), "/datafiles/PBMC", sep = ""))
49 files <-  dir(pattern = "*.fcs")
50
51 # convert and combine data for use in downstream analysis
52 data <- lapply(lapply(files, read.FCS), exprs)
53 combined.data = as.data.frame(do.call(rbind, data))
54
55 # choose channels with markers to use for downstream analysis and apply arcsinh
56 # transformation with a cofactor of 15
57 transformed.chosen.markers <- combined.data %>%
58   select(contains("-"),-contains("Ir")) %>%
59   mutate_all(function(x)
60     asinh(x / 15))        # cofactor here is 15; this can be change
61
62 # set seed for reproducible results (43 is chosen below)
63 overall_seed = 43
64 ```
65
```

Read the data files into R and format for analysis

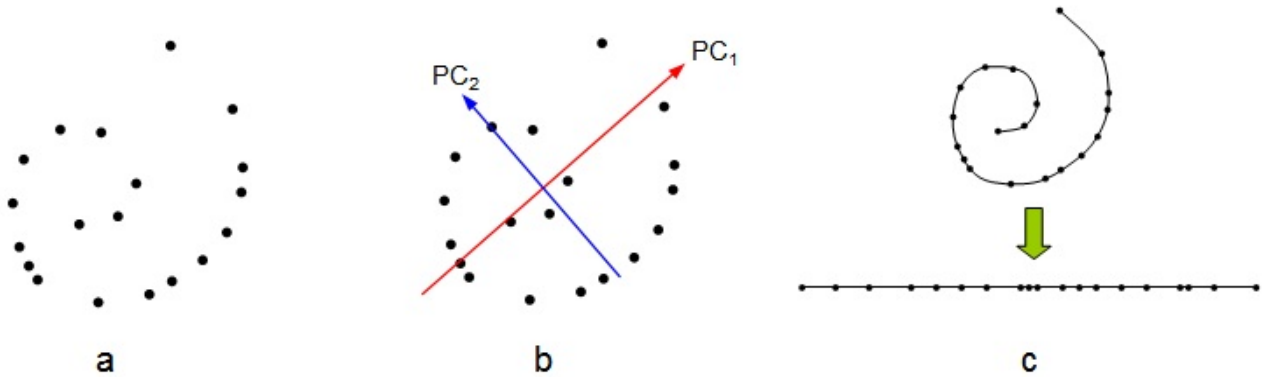
Select channels and scale the data

Choose parameters

PCA is a Linear Dimensionality Reduction Tool



An illustration of PCA. **a)** A data set given as 3-dimensional points. **b)** The three orthogonal Principal Components (PCs) for the data, ordered by variance. **c)** The projection of the data set into the first two PCs, discarding the third one.

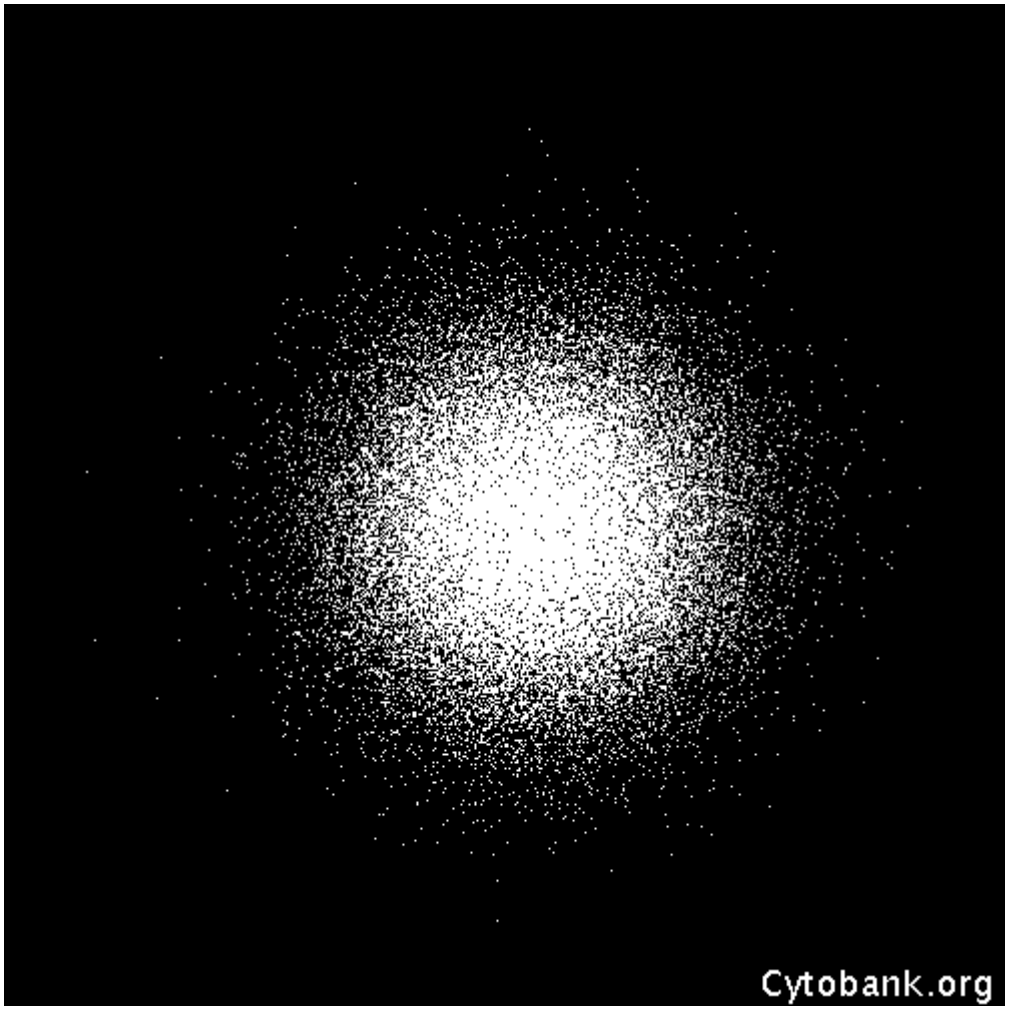


Effects of dimensionality reduction on an inherently non-linear data set. **a)** The original data given as a two-dimensional set. **b)**PCA identifies two PCs as contributing significantly to explain the data variance. **c)** However, the inherent topology (connectivity) of the data helps identify the set as being one-dimensional, but non-linear.

t-Distributed Stochastic Neighbor Embedding is a Dimensionality Reduction Tool

minimizes the divergence between two distributions (one that measures pairwise similarities of input objects and one that measures pairwise similarities of corresponding low-dimensional points)

Parameters:
-perplexity
-iterations
-seed

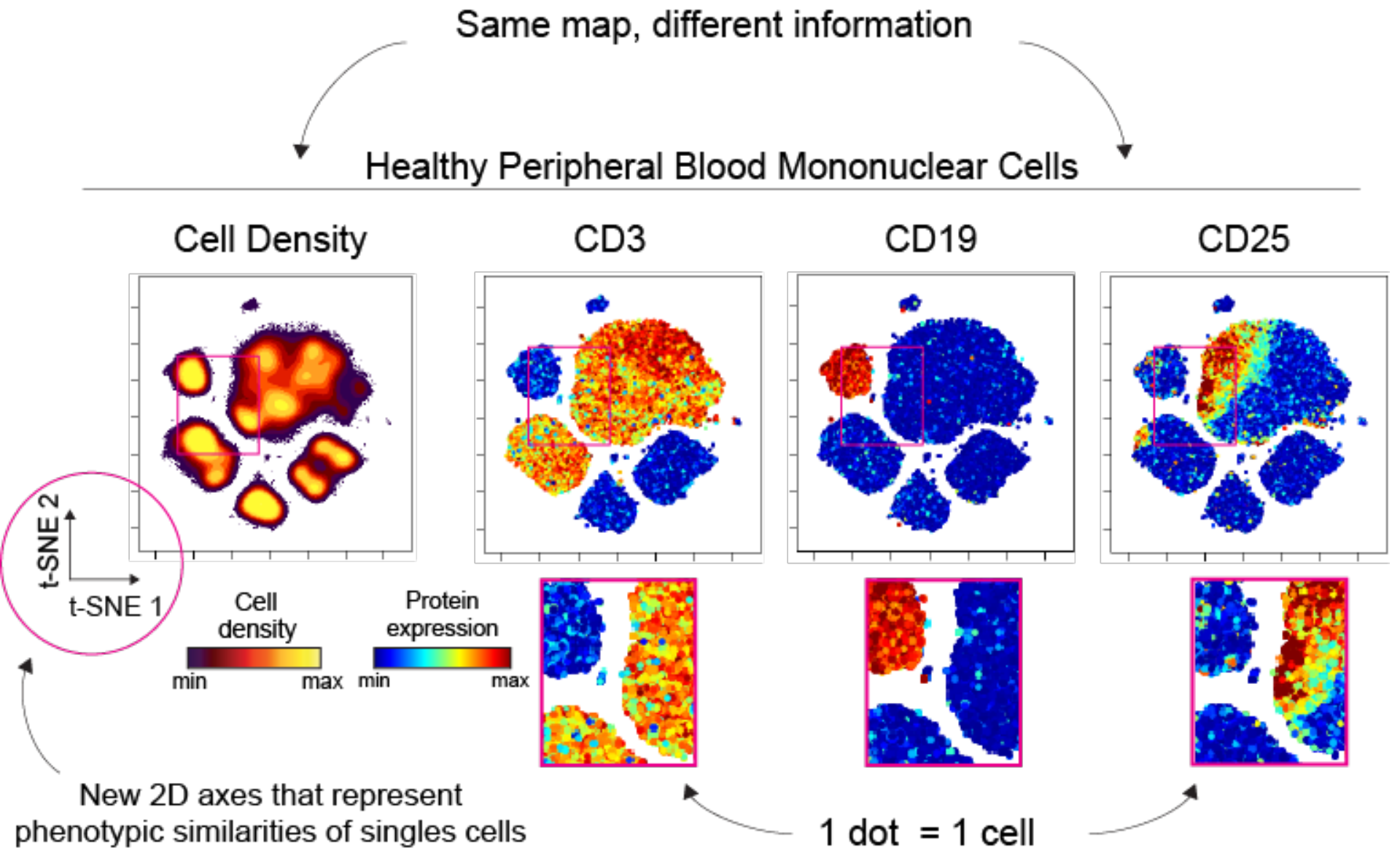


developed by Laurens van der Maaten and Geoffrey Hinton

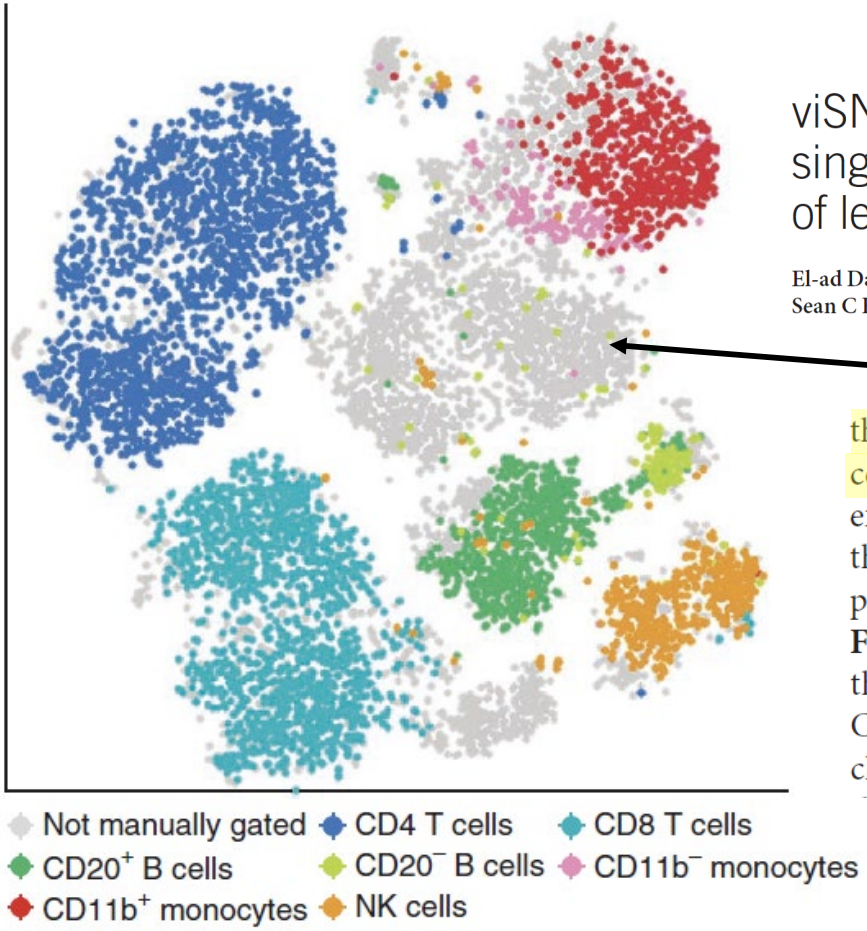
PRESS PLAY ON t-SNE chunk in RMD

Animation created by Cytobank team from iterations of viSNE / t-SNE using Healthy PBMC (26 features)

t-SNE Analysis Allows 2D Visualization of High Dimensional Single Cell Data



t-SNE can Help to Identify Cells Otherwise Lost by Expert Identification



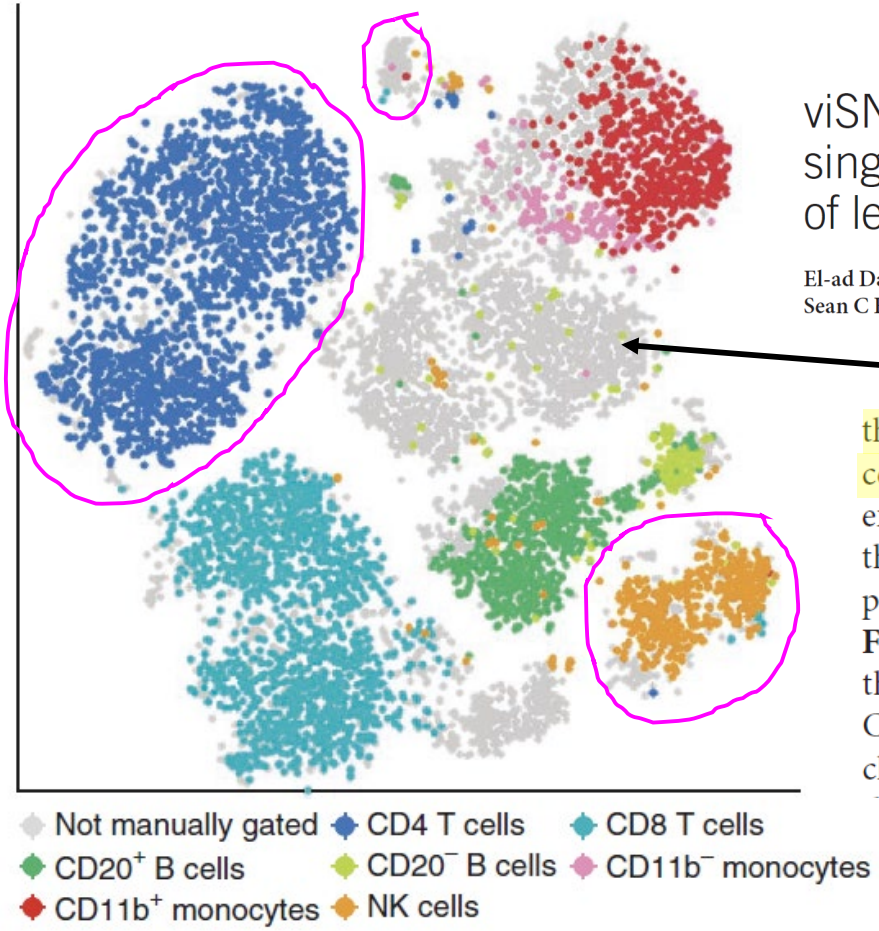
viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia

El-ad David Amir[1], Kara L Davis[2,3], Michelle D Tadmor[1,3], Erin F Simonds[2,3], Jacob H Levine[1,3], Sean C Bendall[2,3], Daniel K Shenfeld[1,3], Smita Krishnaswamy[1], Garry P Nolan[2,4] & Dana Pe'er[1,4]

In all cases, the viSNE gate included cells that were not classified by the expert manually gated biaxial plots; these cells are labeled in gray in the viSNE map. Examination of the marker expression of these cells reveals that they are typically just beyond the threshold of one marker, but the viSNE classification is strongly supported based on the expression of all other markers. For example, in **Figure 1d**, wherein cells are colored for CD11b marker expression, the cells in the gated region express the canonical monocyte marker CD33 (**Supplementary Fig. 1b**). However, only 47% of these cells were classified as monocytes by the manual gating (**Fig. 1b**).

Amir et al., *Nature Biotechnology* 2013

Experts can use t-SNE Axes to Select Cells of Interest



viSNE enables visualization of high dimensional single-cell data and reveals phenotypic heterogeneity of leukemia

El-ad David Amir[1], Kara L Davis[2,3], Michelle D Tadmor[1,3], Erin F Simonds[2,3], Jacob H Levine[1,3], Sean C Bendall[2,3], Daniel K Shenfeld[1,3], Smita Krishnaswamy[1], Garry P Nolan[2,4] & Dana Pe'er[1,4]

In all cases, the viSNE gate included cells that were not classified by the expert manually gated biaxial plots; these cells are labeled in gray in the viSNE map. Examination of the marker expression of these cells reveals that they are typically just beyond the threshold of one marker, but the viSNE classification is strongly supported based on the expression of all other markers. For example, in **Figure 1d**, wherein cells are colored for CD11b marker expression, the cells in the gated region express the canonical monocyte marker CD33 (**Supplementary Fig. 1b**). However, only 47% of these cells were classified as monocytes by the manual gating (**Fig. 1b**).

Amir et al., *Nature Biotechnology* 2013

01_PBMC_workflow_example.rmd

Run t-SNE

```
66 ```{r run_t-SNE}
67 # Time ~5 min
68
69 set.seed(overall_seed)
70
71 # the line below will run t-SNE on the scaled surface markers (to see help p
   for t-SNE, type "?Rtsne -- enter" in console)
72 # you can view t-SNE progress by opening up the console below
73 mytSNE = Rtsne(
74   transformed.chosen.markers,                          # input scaled data
75   dims = 2,                                            # number of final
   dimensions
76   initial_dims = length(transformed.chosen.markers), # number of initial
   dimensions
77   perplexity = 30,          # perplexity (similar to # of nearest neighbors,
78                             # will scale with data sets, cannot be greater t
79                             # the number of events minus 1 divided by 3)
80   check_duplicates = FALSE,
81   max_iter = 1000,          # number of iterations
82   verbose = TRUE
83 )
84 tsne.data = as.data.frame(mytSNE$Y)
85 ```
86
87 ```{r plot_t-SNE}
88 # Time <10 sec
89
90 # setting aspect ratio for plots
91 range <- apply(apply(tsne.data, 2, range), 2, diff)
92 graphical.ratio <- (range[1] / range[2])
93
94 # t-SNE flat dot plot and density dot plot (1 dot = 1 cell)
95 tsne.plot <- data.frame(x = tsne.data[, 1], y = tsne.data[, 2])
```

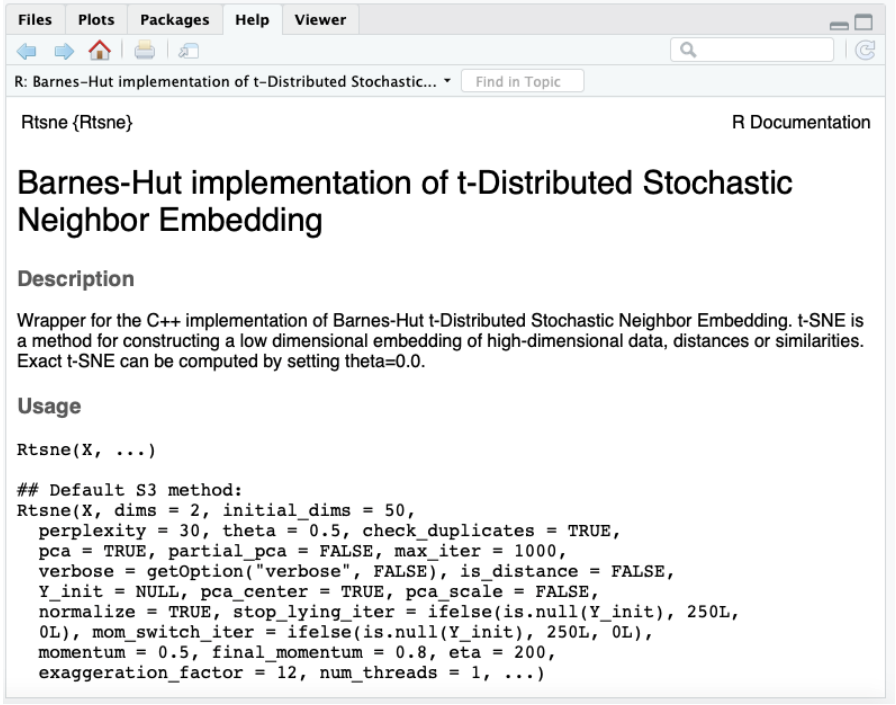
This section will run a t-SNE analysis on the PBMC data with set parameters

You can choose the resulting numbers of dimensions, the perplexity, and the iterations

This section will plot the t-SNE results. Two plots will appear, a “flat” dot plot and a density plot

For help pages for tools type…

?Rtsne -> enter
?umap -> enter
?FlowSOM -> enter
?MEM -> enter



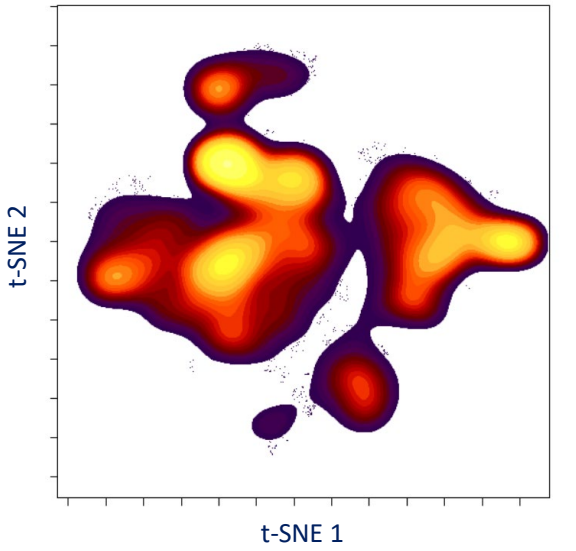
…in console to the right of >

UMAP (Uniform Manifold Approximation and Projection) is Another Dimensionality Reduction Tool

Superior run times

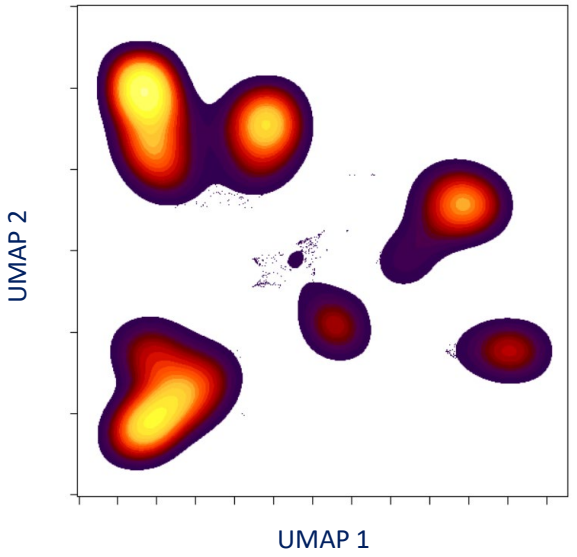
Emphasis on both global and local structure in the data

t-SNE



VS.

UMAP



UMAP: McInnes et al., *arXiv* 2018; Becht et al., *Nature Biotech* 2019
Data used for t-SNE and UMAP: human PBMC

UMAP Preserves Local and Global Structure

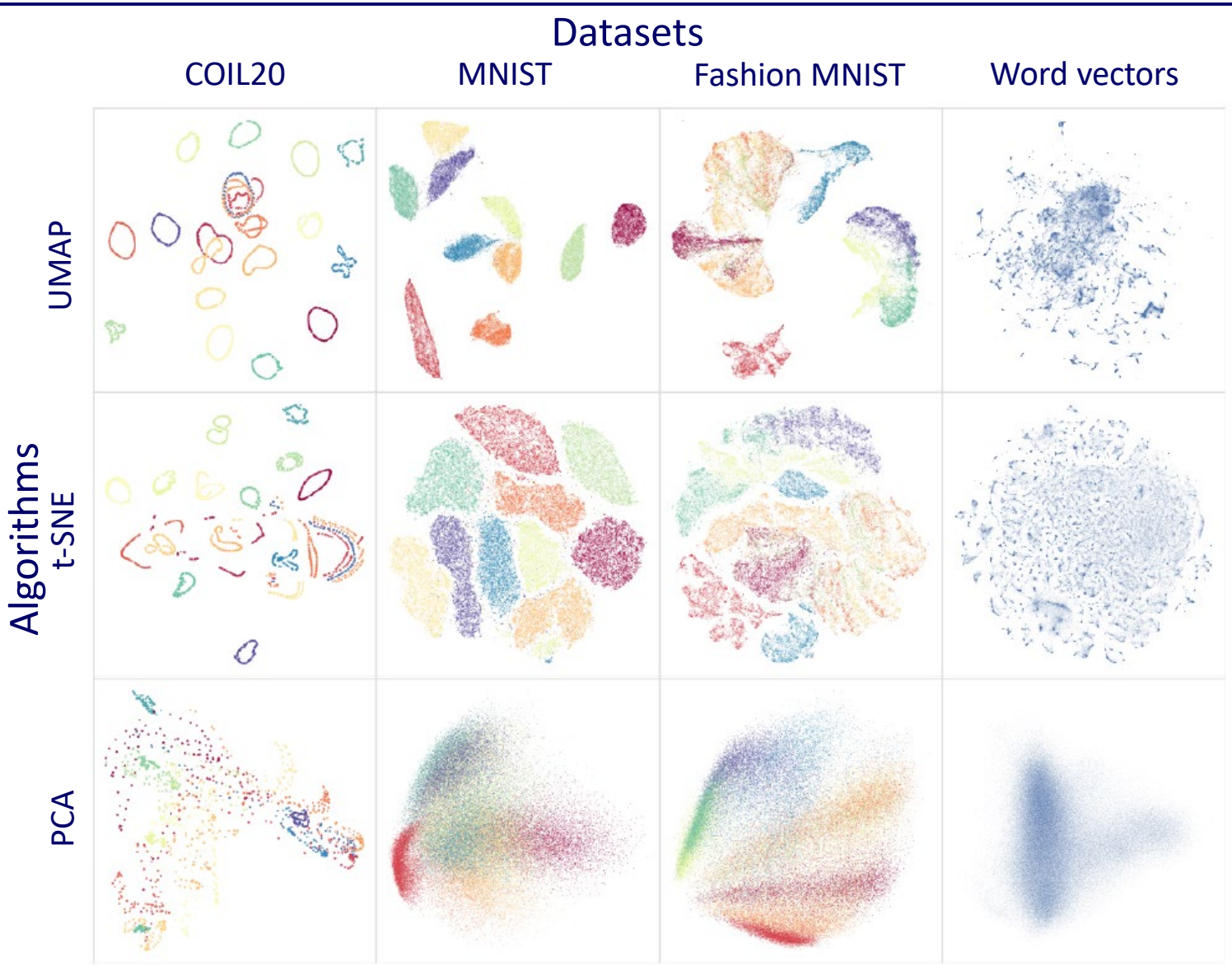


Figure 2: A comparison of several dimension reduction algorithms. UMAP reflects much of the large scale global structure, while also preserving the local fine structure similar to t-SNE.

McInnes et al., *arXiv* 2018

01_PBMC_workflow_example.rmd

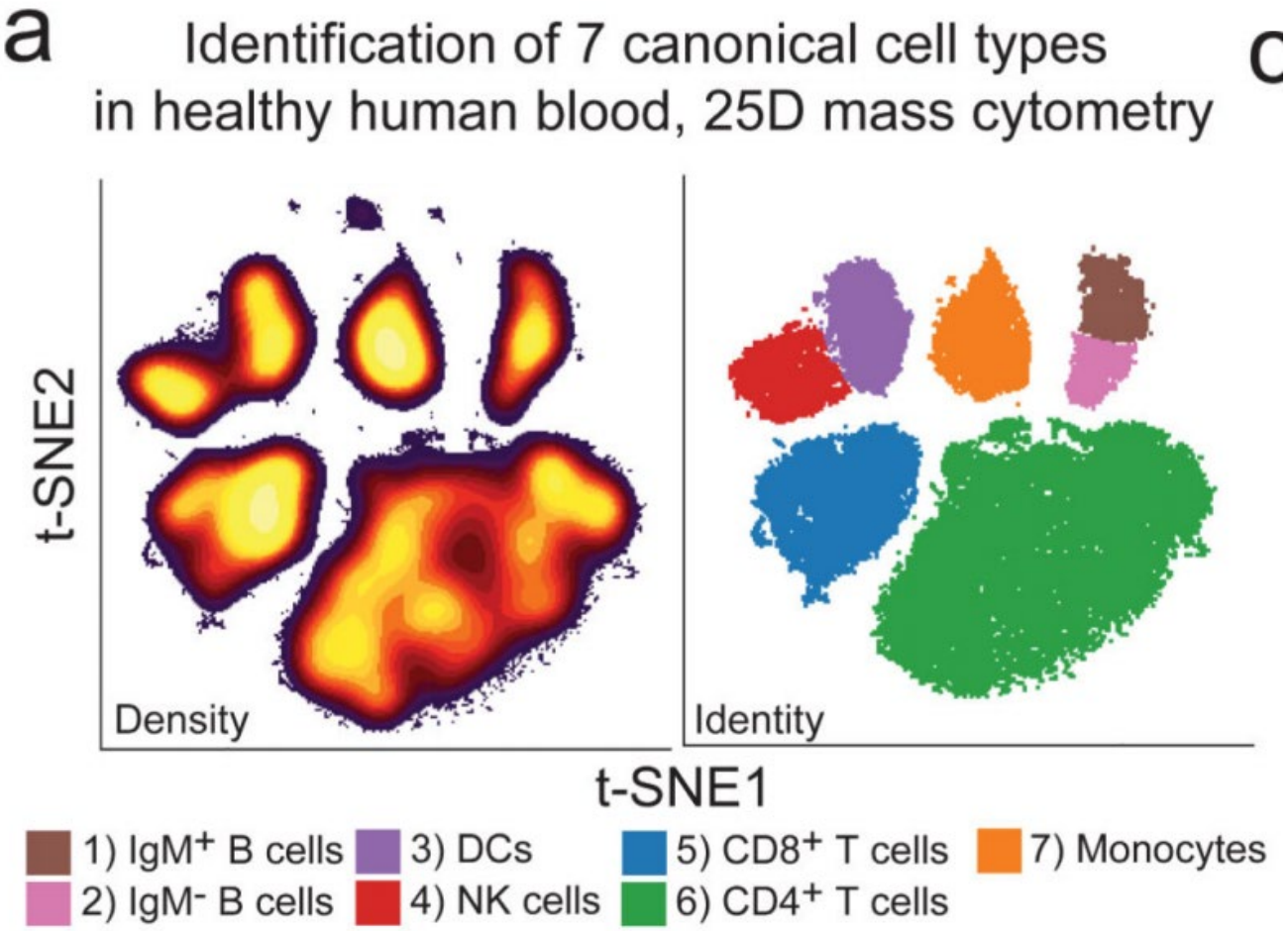
Run UMAP

```
119 ```{r run_UMAP}
120 # Time ~1 min
121
122 # Run UMAP on all scaled surface markers
123 set.seed(overall_seed)
124
125 # the line below will run UMAP on the data set (to see help page for UMA
    "UMAP -- enter" in console)
126 # you can view UMAP progress by opening up the console below
127 myumap <-
128   umap(transformed.chosen.markers,  # input scaled data
129        n_neighbors = 15,            # number of nearest neighbors to look at,
    scales with data set
130        n_threads = 1,               # this argument makes UMAP reproducible
131        verbose = TRUE)
132 umap.data = as.data.frame(myumap)
133 ```
134
135 ```{r plot_UMAP}
136 # Time <10 sec
137
138 # setting aspect ratio for plots
139 range <- apply(apply(umap.data, 2, range), 2, diff)
140 graphical.ratio <- (range[1] / range[2])
141
142 # UMAP flat dot plot and density dot plot (1 dot = 1 cell)
143 UMAP.plot <- data.frame(x = umap.data[, 1], y = umap.data[, 2])
144
145 # dot plot
146 ggplot(UMAP.plot) + coord_fixed(ratio = graphical.ratio) +
147   geom_point(aes(x = x, y = y), cex = 0.5) + labs(x = "UMAP 1", y = "UMAP 2",
148                                                  title = "PBMC Data on UMAP
    Axes") +
149   theme_bw() +
```

This section will run a UMAP analysis on the PBMC data using set parameters

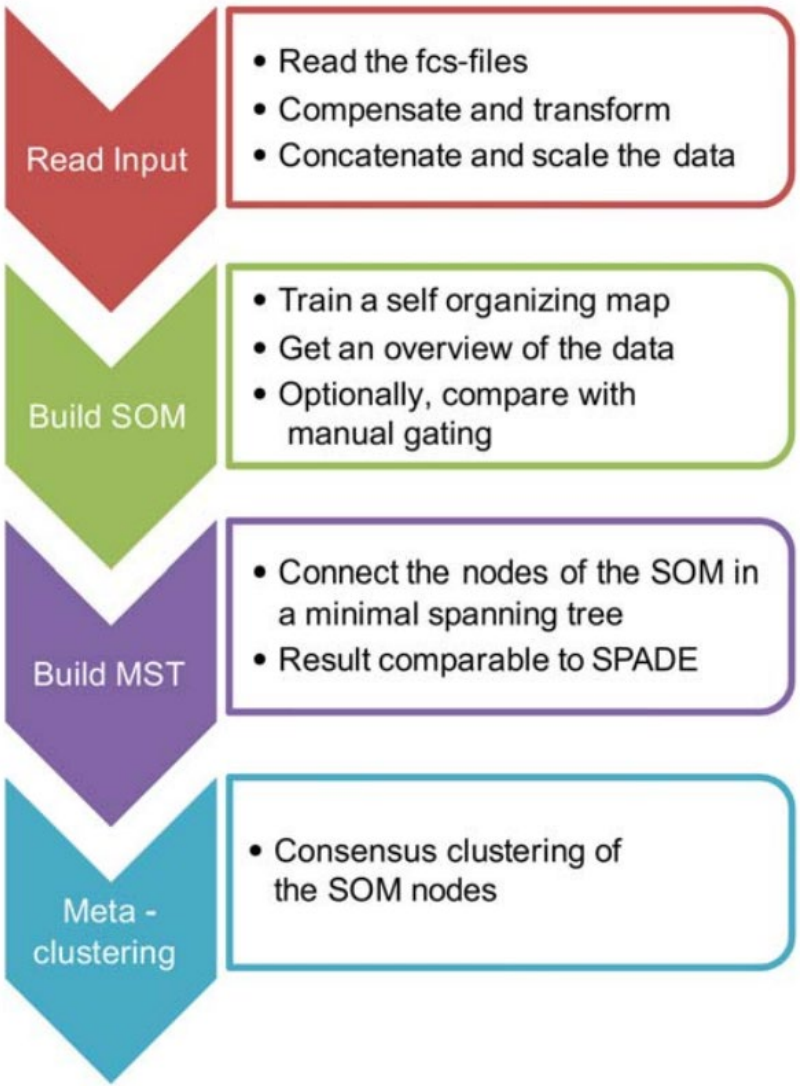
This section will plot the UMAP results. Two plots will appear, a “flat” dot plot and a density plot

Clusters can be Identified Based on Dimensionality Reduction Results



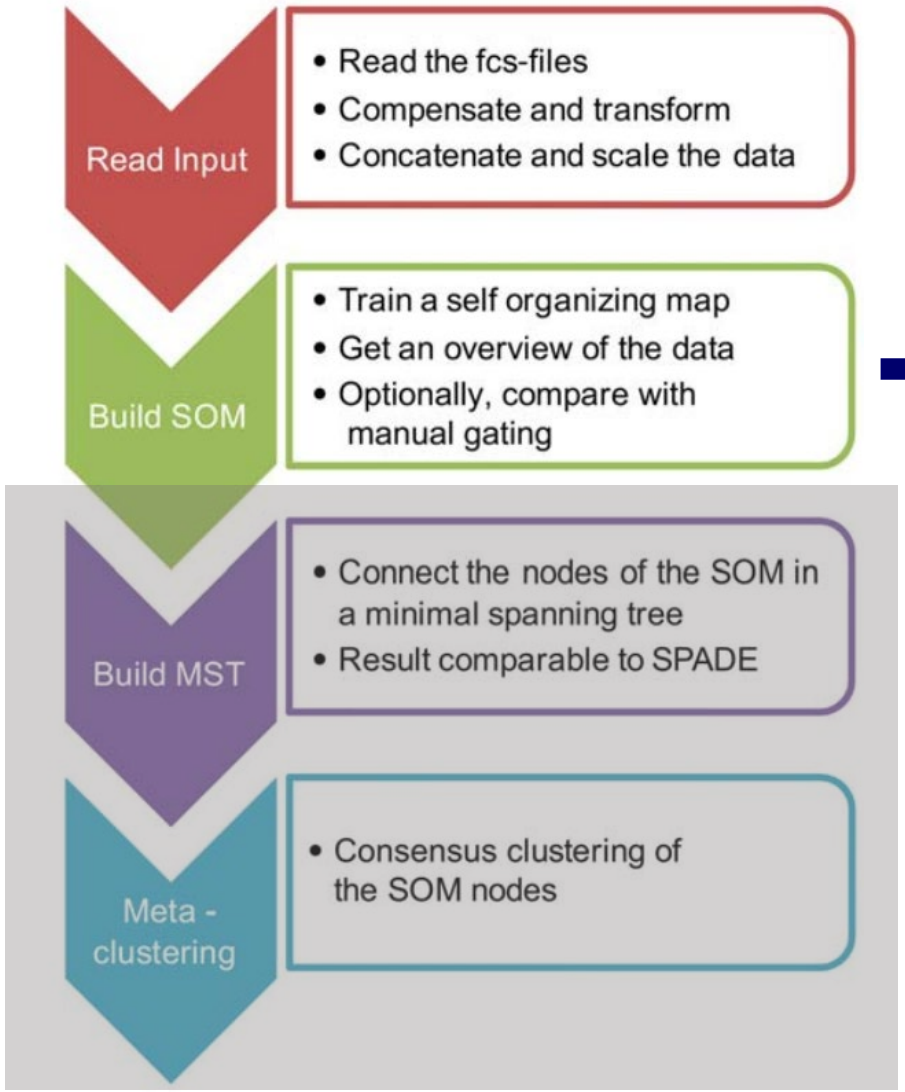
Diggins et al., *Nature Methods* 2017

Clustering with FlowSOM: Self-organizing Maps



Van Gassen et al., *Cytometry A* 2015
See also, FlowSOM does well in a comparison of clustering tools: Weber & Robinson, *Cytometry A* 2017

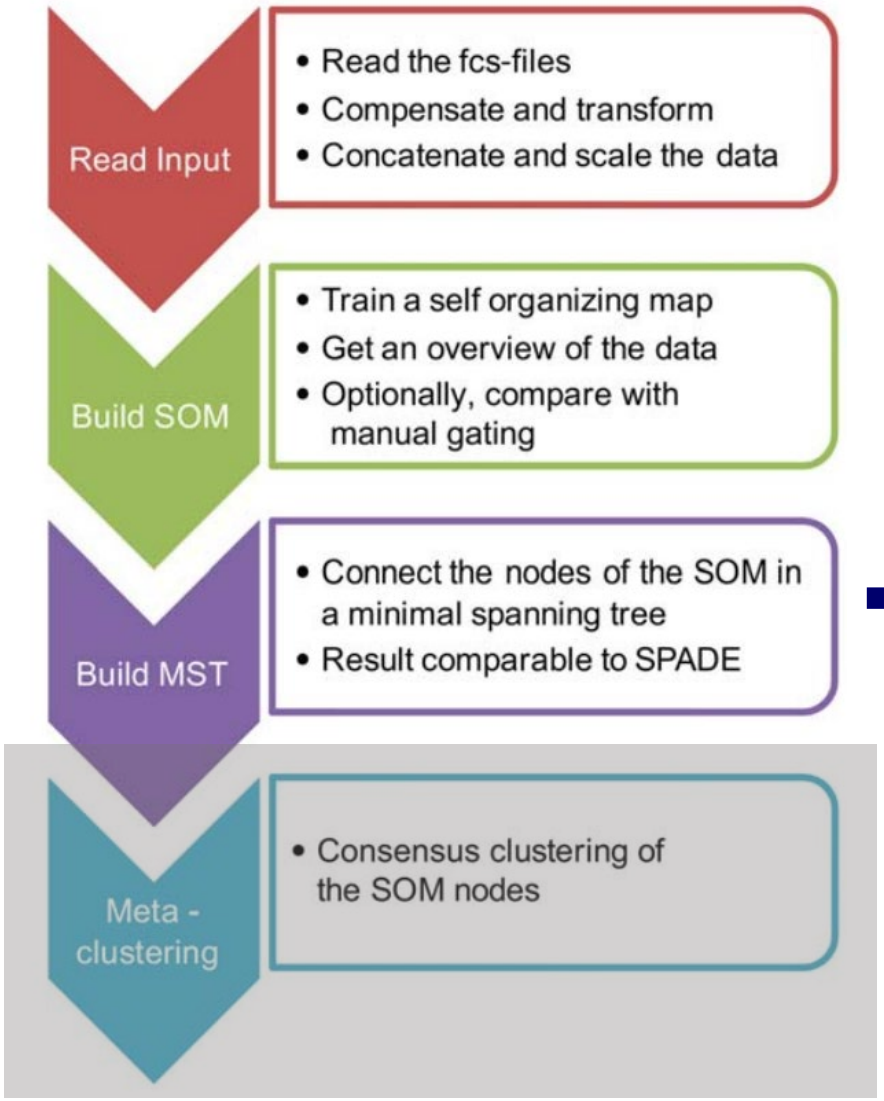
Clustering with FlowSOM: Self-organizing Maps



A SOM is created by assigning data points/cells to nodes based on their multidimensional phenotypes, updating this map repeatedly until each cell is assigned to a node with the most similar cells

Van Gassen et al., *Cytometry A* 2015
See also, FlowSOM does well in a comparison of clustering tools: Weber & Robinson, *Cytometry A* 2017

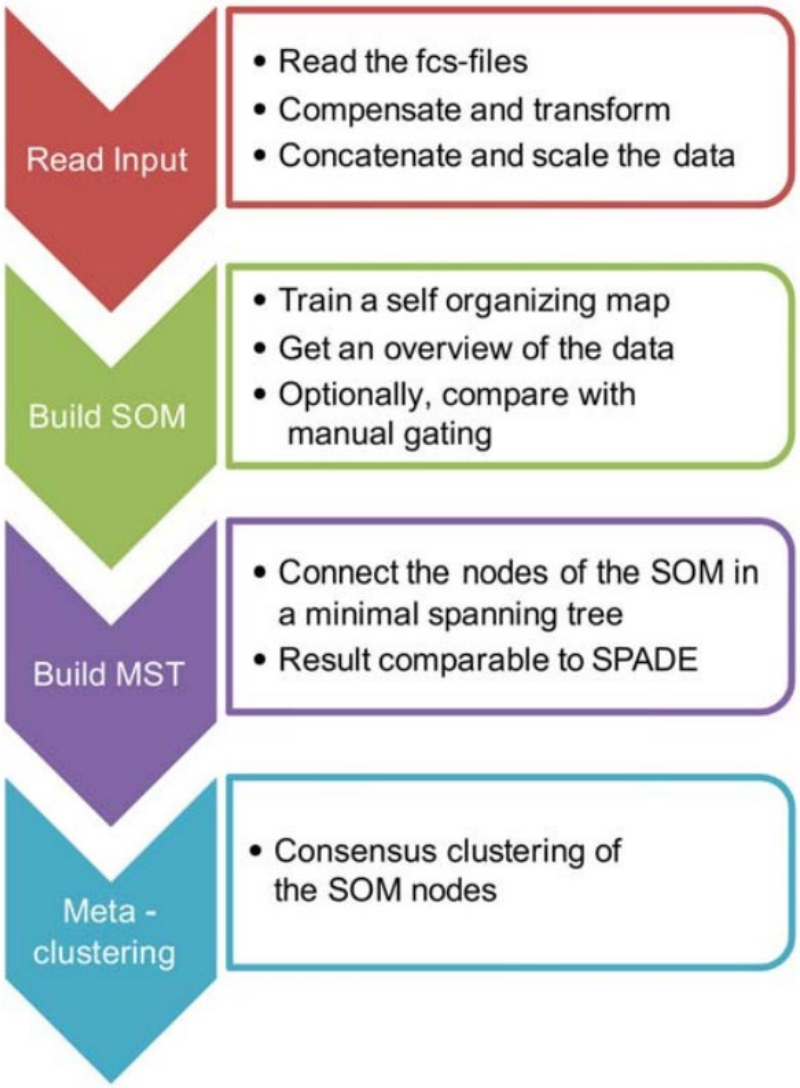
Clustering with FlowSOM: Self-organizing Maps



The next step is to arrange the nodes along a minimal spanning tree (MST), so that nodes that are most similar are closest on the tree
not used in our visualization

Van Gassen et al., *Cytometry A* 2015
See also, FlowSOM does well in a comparison of clustering tools: Weber & Robinson, *Cytometry A* 2017

Clustering with FlowSOM: Self-organizing Maps



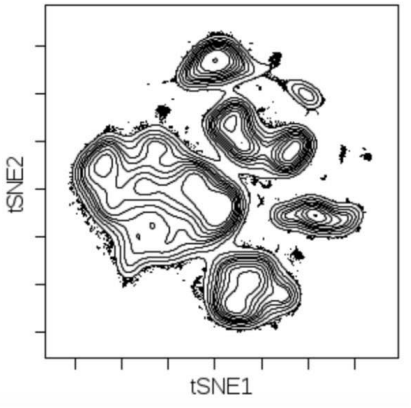
Finally, similar nodes are combined based on the number of desired clusters defined by the user. This desired number can be based on prior knowledge or a specific goal (i.e. minimizing intracluster variance)

Van Gassen et al., *Cytometry A* 2015

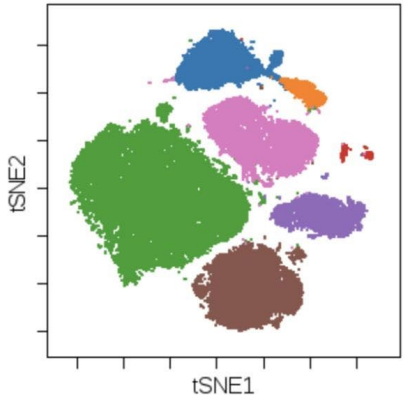
See also, FlowSOM does well in a comparison of clustering tools: Weber & Robinson, *Cytometry A* 2017

Clustering with FlowSOM: Self-organizing Maps

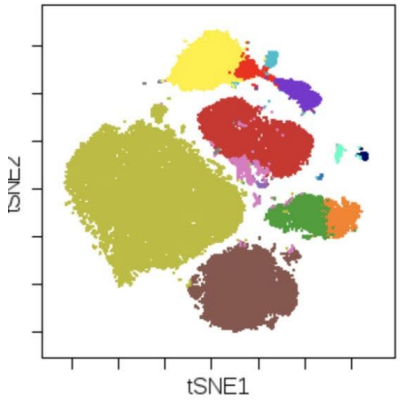
Contour plot of viSNE map



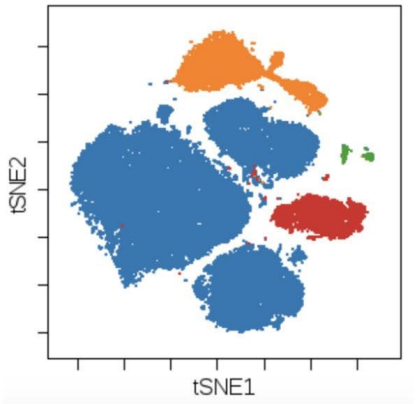
FlowSOM metaclusters overlaid on viSNE map



 # metaclusters = 7



 # metaclusters = 15



 # metaclusters = 4

01_PBMC_workflow_example.rmd

Run FlowSOM

````
```{r run_FlowSOM}
# Time <10 sec

# create flowFrame for FlowSOM input (using umap axes as input)
matrix <- as.matrix(umap.data)
metadata <-
  data.frame(name = dimnames(matrix)[[2]],
             desc = paste('UMAP', dimnames(matrix)[[2]]))
metadata$range <- apply(apply(matrix, 2, range), 2, diff)
metadata$minRange <- apply(matrix, 2, min)
metadata$maxRange <- apply(matrix, 2, max)
flowframe <- new("flowFrame",
                     exprs = matrix,
                     parameters = AnnotatedDataFrame(metadata))

# implement the FlowSOM on the data by running the line below (to see help for FlowSOM, type "FlowSOM -- enter" in console)
fsom <-
  FlowSOM(
    flowframe,      # input flowframe
    colsToUse = c(1:2),  # columns to use
    nClus = 10,          # target number of clusters (this can be changed)
    seed = overall_seed  # set seed for reproducibility
  )
FlowSOM.clusters <-
  as.matrix(fsom[[2]][fsom[[1]]$map$mapping[, 1]])
```

```{r plot_clusters}
# Time <10 sec

# plot FlowSOM clusters on UMAP axes
ggplot(UMAP.plot) + coord_fixed(ratio=graphical.ratio) +
  geom_point(aes(x=x, y=y, color=FlowSOM.clusters), cex = 0.5) +
  labs(x = "UMAP 1", y = "UMAP 2",title = "FlowSOM Clustering on UMAP Axes
       color = "Cluster") + theme_bw() +
  guides(colour = guide_legend(override.aes = list(size=4)))+
  labs(caption = "Data from Diggins et al., Nat Methods 2017, 14: 275-278 \nFlow Repository: FR-FCM-ZY63") +
  theme(panel.grid.major = element_blank(),
              panel.grid.minor = element_blank())
```
````

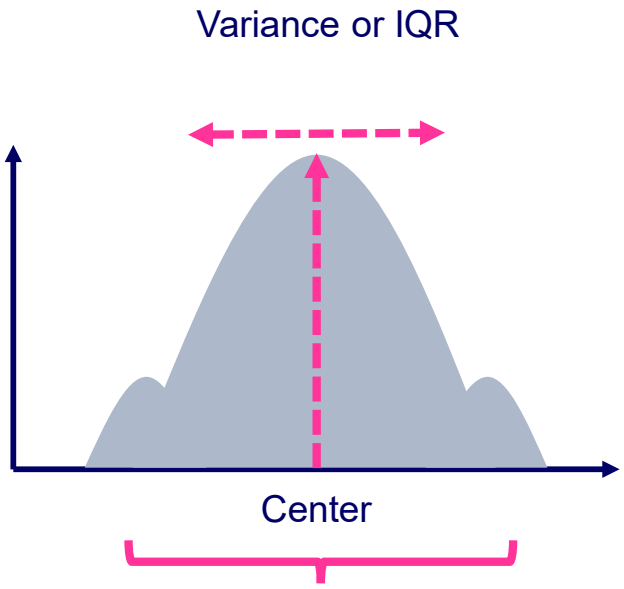
This section performs FlowSOM clustering on the UMAP results

You can choose the parameters the clustering is performed on (UMAP axes vs. measured markers) as well as a seed and desired number of clusters

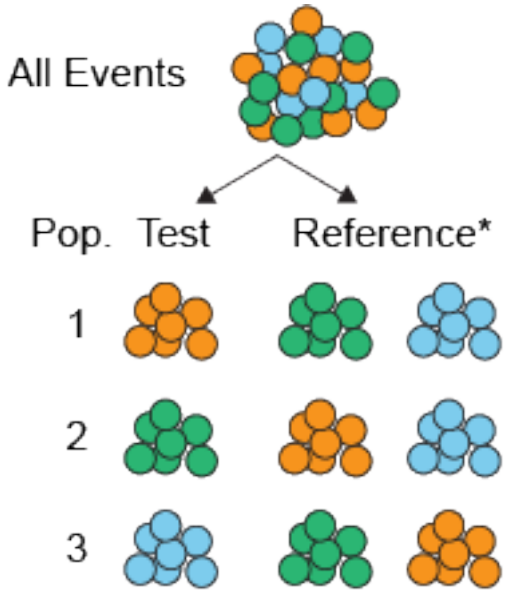
This section plots the identified clusters back onto the UMAP axes and generates a plot (a colored version of the UMAP plot from before)

Marker Enrichment Modeling Analysis Identifies Markers that are Specifically Expressed or Lacking on Populations

MEM accounts for variance and median of markers to identify enriched features on subsets of cells



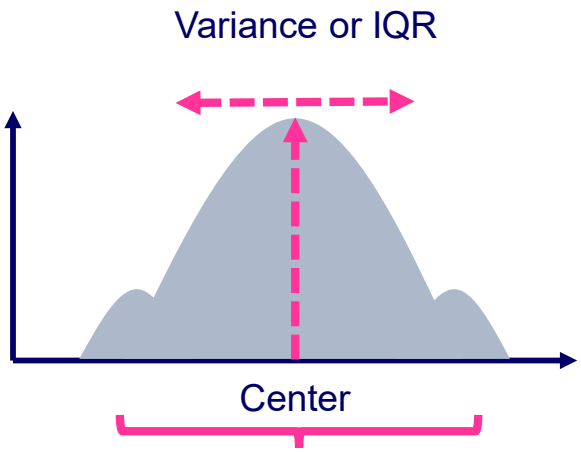
Shape (skewness, symmetry
peaks, outliers, etc.)



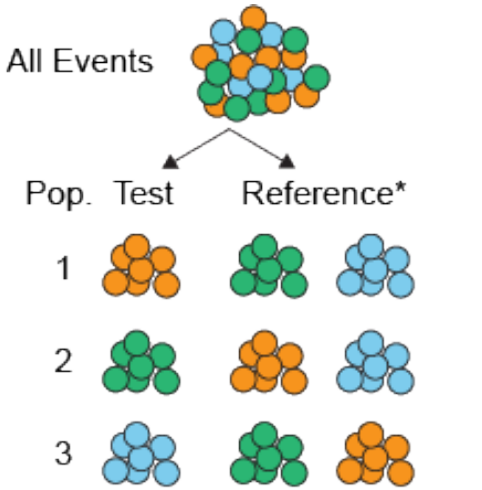
*All non-population as reference

Adapted from Kirsten Diggins

MEM Quantifies Relative Enrichment by Combining Magnitude and Interquartile Range



Shape (skewness, symmetry
peaks, outliers, etc.)



*All non-population as reference

MEM label

▲HLADR^{+10} CD20^{+9} CD19^{+7} IgM^{+5} CD34^{+3} CD45RA^{+3} CXCR4^{+2} CD47^{+2} CD33^{+2}
▼CD7^{-2}

Linear transformation to -10 to +10

If $MAG_{test} - MAG_{ref} < 0$, MEM = -MEM

Adapted from Kirsten Diggins

01_PBMC_workflow_example.rmd

Run MEM

```r
```{r run_MEM}
# Time ~30 sec

# Run MEM on the FlowSOM clusters found from using UMAP axes
cluster = as.numeric(as.vector((FlowSOM.clusters)))
MEM.data = cbind(transformed.chosen.markers, cluster)

MEM.values = MEM(
  MEM.data,              # input data (last column must contain cluster values)
  transform = FALSE,     # data is already scaled in this case
  cofactor = 1,
  choose.markers = FALSE,
  markers = "all",       # use all transformed, chosen markers from previous selection
  choose.ref = FALSE,    # reference will be all other cells
  zero.ref = FALSE,
  rename.markers = FALSE,
  new.marker.names =
"CD19,CD117,CD11b,CD4,CD8,CD20,CD34,CD61,CD123,CD45RA,CD45,CD10,CD33,C
,CD69,CD15,CD16,CD44,CD38,CD25,CD3,IgM,HLA-DR,CD56", # rename channels labels
  file.is.clust = FALSE,
  add.fileID = FALSE,
  IQR.thresh = NULL
)

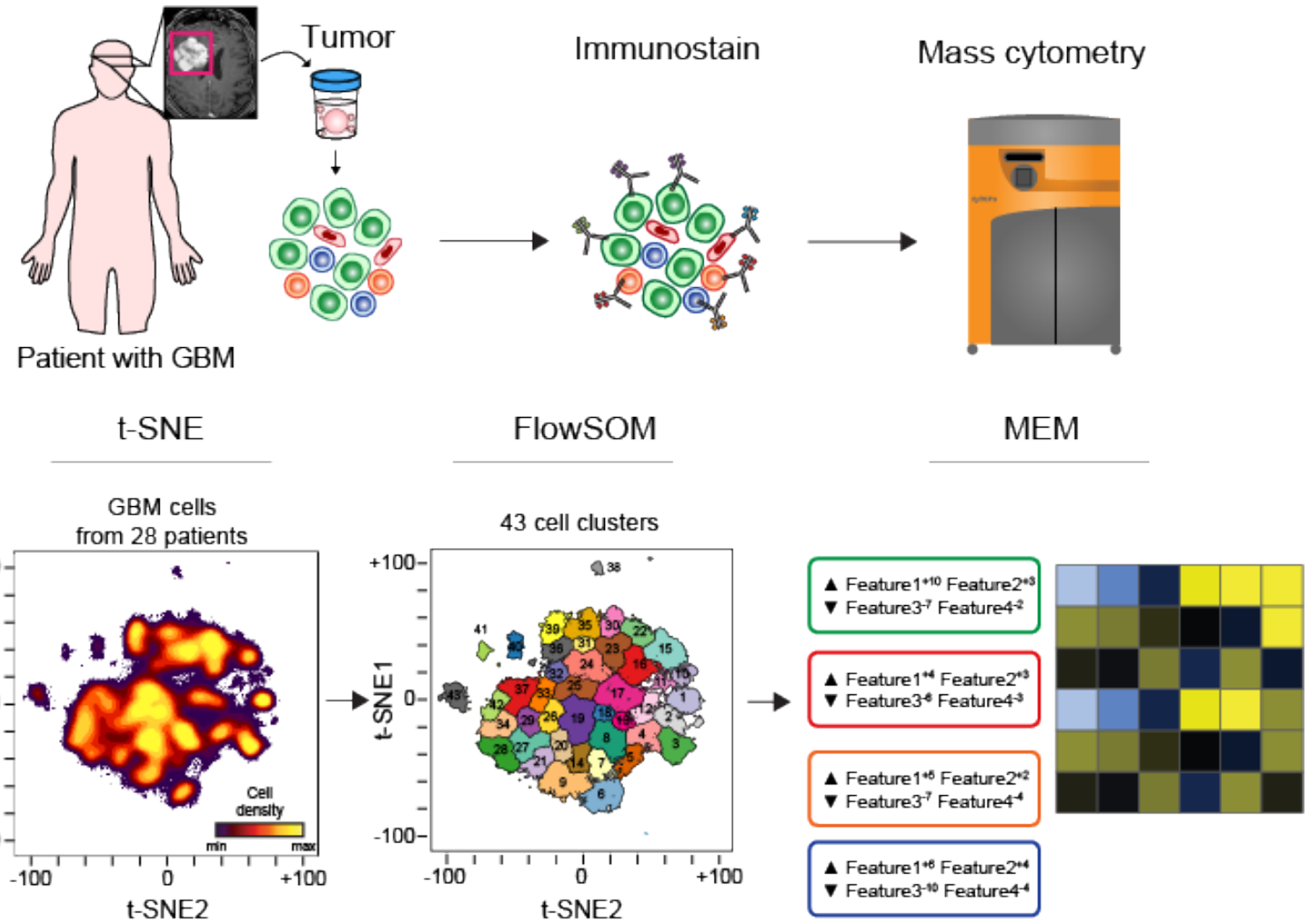
# build MEM heatmap and output enrichment scores
build.heatmaps(
  MEM.values,                  # input MEM values
  cluster.MEM = "both",        # dendrogram for columns and rows
  display.thresh = 3,          # display threshold for MEM scores
  newwindow.heatmaps = FALSE,
  output.files = TRUE,         # makes txt and PDF files for heatmap and MEM scores
  labels = TRUE,               # include labels in heatmap
  only.MEMheatmap = FALSE
)
```
```

This section performs MEM analysis on the identified FlowSOM clusters

You can choose the markers for the MEM analysis as well as their names and the reference population

This section produces heatmaps and MEM (enrichment) scores

t-SNE, FlowSOM, and MEM can be Used in a Data Analysis Workflow



https://www.biorxiv.org/content/10.1101/632208v1

Acknowledgements

Irish Lab

Jonathan Irish
Sierra Barone
Todd Bartkowiak
Caroline Roe
Madeline Hayes

Past Irish Lab Members

Nalin Leelatian
Kirsten Diggins
Jocelyn Gandelman
Allison Greenplate
Deon Dixie
Cara Wogsland

Ihrie Lab

Rebecca Ihrie
Justine Sinnaeve
Akshitkumar Mistry
Asa Brockman
Laura Winalski
Bret Mobley
Ethan Chervonski