Chapter 13

Identifying gene markers associated to cell subpopulations.

Maria Luisa Ratto and Luca Alessandri

Molecular Biotechnology Center, Turin, Turin, Italy; l.alessandri@unito.it

Abstract

An important point of the analysis of a single cell RNA experiment is the identification of the key elements, i.e. genes, characterizing each cell subpopulation cluster. In this chapter, we describe the use of sparsely-connected-autoencoder, as a tool to convert single cell clusters in pseudo-RNAseq experiments to be used as input for differential expression analysis, and the use of COMET as tool to depict cluster specific gene markers.

Key words Feature selection, comet, sparsely connected autoencoder.

---

1 Introduction

Single cell RNA sequencing (scRNAseq) offers the possibility to identify cell sub-populations, based on the single cell transcriptome. Clustering is the most used instrument to aggregates cells in groups, based on their transcription profile [1]. Notably, Yu and coworkers [1], highlight that good performance on cell clustering does not necessarily guarantee high accuracy in depicting different cell types. Furthermore, clustering algorithms can depict tiny differences among very similar cells, which are due to confounding factors, not necessarily linked to a functional difference among clusters, e.g. cell cycle, pre-apoptotic cells, etc. On the other hand, cell type annotation, based on previously known cell-type specific markers [2], might define different cell types within the same

cluster, raising the possibility that annotation was imprecise. For this reason, it is useful to integrate basic cell type annotation with cluster-specific gene markers, depicted by the analysis of cluster-specific transcriptomics data. In this chapter, we describe the use of two different approaches: SCA [3-5] (Sparsely Connected Autoencoder) and COMET [6] (COmbinatorial Marker dEtection from single-cell Transcriptomics).

SCA can be used to generate cluster-specific pseudo-bulks resembling RNAseq data [3]. The overall idea of this approach is that genes characterizing a specific cluster will be the non-noise signal caught by the hidden layer of SCA.

COMET is a statistical method, based on XL-minimal HyperGeometric test (XL-mHG test), depicting cluster specific genes.

In this chapter, we show how to use the above-mentioned approaches for the analysis of scRNAseq clusters.

2 Materials

2.1 Minimal hardware/software requirements

The analyses described in the present chapter required a workstation with at least 64 Gb RAM, 2Tb SSD, 8 CPUs. All analytical steps require a UNIX-like environment (See **Note 1** and **2**).

2.2 Exemplary dataset

The exemplary dataset RNA-5c is available at the GitHub repository https://github.com/kendomaniac/single_cell_transcriptomics/tree/main/tian_et_al_2018. It includes the human lung adenocarcinoma cell lines A549, H1975, H2228, H838, and HCC827. It is derived from GEO repository ID: GSM3618014 [7]. The count table is associated with cell names and cell lines, e.g. Lib90_00000.HCC827, Lib90_00002.H838. RNA-5c includes 1242 cells of A549, 436 cells belong to H1975, 749 cells of H2228, 879 cells belong to H838, and 598 cells of HCC827 cells.

The exemplary dataset 390c_wctype, [2], is available at the Github repository https://github.com/kendomaniac/single_cell_transcriptomics.git. This dataset refers to the sample

390c of the immune-gut atlas transcriptome and, in the counts table, cell names incorporate cell type assignment, i.e TGACTAGGTTCCACAA.Treg. This dataset is part of 41,650 cells isolated from the caecum, transverse colon and sigmoid colon of 5 individuals [2], which is one of the datasets constituting the immune-gut atlas transcriptome (see **Note 3**). All data are available as figshare repository: 10.6084/m9.figshare.19391234.

3 Methods

3.1 rCASC installation

rCASC [5] is a single cell analysis suite providing a computation reproducible infrastructure based on docker containers. The installation of rCASC is very simple. In a R session, type the code indicated below (see **Note 4**).

```
# Open an R session and type the following commands
library(devtools)
install_github("kendomaniac/rCASC", ref="master")
library(rCASC)
```

rCASC provides an infrastructure to evaluate the overall quality of the clustering result via the cell stability score (CCS, see **Note 5**).

3.2 Data cleanup, filtering, and clustering

Before performing any analysis, we strongly suggest checking the overall quality of the data. Specifically, we suggest running the *mitoRibo* function implemented in rCASC. Furthermore, the dataset needs to be imputed using SAVER tool [8] and annotated with *scannoByGtf* rCASC function. To reduce the size of the count table under analysis, we suggest filtering the dataset, keeping only the subset of genes being the most expressed and characterized by the largest variance among cells. This filtering can be performed using the *topx* function available in rCASC.

```
# Folder where the counts table is located.
WD = "/mnt/cold1/calogero/5-lung_cell-lines/scRNAseq_5-lung_cell-lines"
INPUT="saver_GSM3618014_gene_count.csv"
SEPARATOR=","
GTF="Homo_sapiens.GRCh38.101.gtf"
SCRATCH="/scratch"

setwd(WD)
library(rCASC)
#filtering, selecting 5k most expressed genes
```

```
topx(group = "docker", file=paste(getwd(),
paste("filtered_annotated_saver_ribomito", INPUT, sep="_"), sep="/"),
threshold=5000, separator=SEPARATOR, type="expression")
#filtering, selecting the 2.5k most variant genes
topx(group = "docker", file=paste(getwd(),
paste("filtered_expression_filtered_annotated_saver_ribomito", INPUT,
sep="_"), sep="/"), threshold=2500, separator=SEPARATOR, type =
"variance")
system("mkdir VandE")
file.rename(from=paste("filtered_variance_filtered_expression_filtered_an
notated_saver_ribomito", INPUT, sep="_"), to="VandE.csv")
system("mv VandE.csv VandE")
```

In this exemplary analysis, the Louvain modularity clustering, implemented in Seurat [9], is used to aggregate cells in clusters for RNA-5c and 390c samples, but the it is not mandatory the use of this specific clustering tool.

```
library(rCASC)
seuratPCAEval(group = "docker", scratch.folder=SCRATCH,
file=paste(getwd(), "VandE/VandE.csv", sep="/"), separator=",", logTen =
0, seed = 111)

#Running clustering
seuratBootstrap(
  group = "docker",
  scratch.folder= "/scratch",
  file=paste(getwd(),"VandE.csv", sep="/"),
  nPerm=40,
  permAtTime=10,
  percent=10,
  separator=",",
  logTen = 0,
  pcaDimensions=10,
  seed = 111,
  sparse = FALSE,
  format = "NULL",
  resolution = 0.8
)
```

The above function generates the following path: ~/Results/sample_names/number_clusters. In the folder defined by the number of clusters, there is a pdf ending with _Stability_Plot.pdf (Fig. 1).

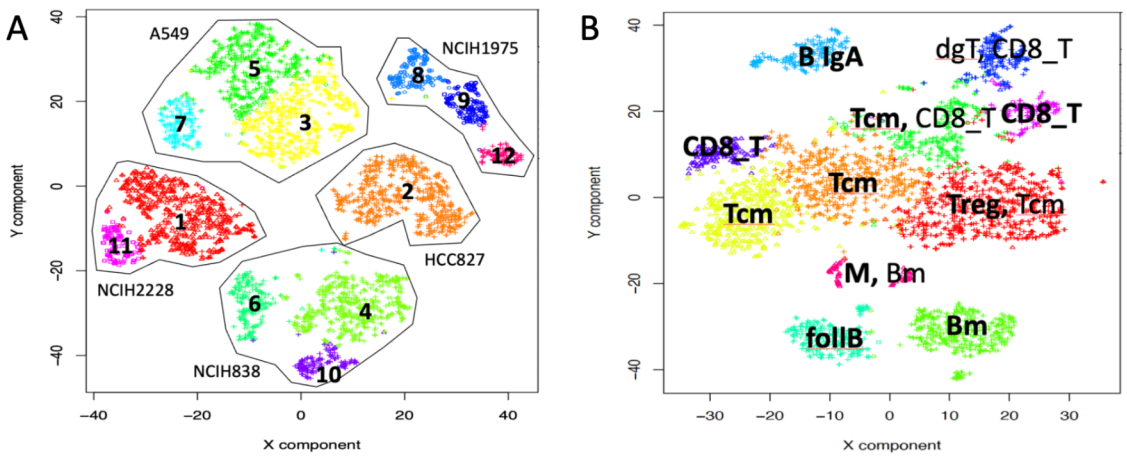

FIG. 1

The clustering in Fig. 1A shows the presence of one cluster constituted by a single cell type, clusters 2 (HCC827 cell line), as instead all other cells lines are split in different clusters. Cell lines are a typical instrument for in vitro analysis of cell biological activity. In the field of oncology, cell lines are frequently used to study the responsivity to a specific drug. Drugs response experiments are not done at single cell level usually, thus IC50 cannot be linked to a specific cell subpopulation response. scRNAseq offers the possibility to investigate if genes, specifically associated to cell subpopulations of a cell line, are different facet of the same die or represent a peculiar feature of the subpopulation, which might be related to a specific biological phenotype of the cell line, i.e. drug resistance.

The clustering shown in Fig. 1B, show the presence of a heterogeneous organization of immunological cell types in the detected clusters:

i. clusters sharing different cell types (Treg and Tcm in red cluster and Tcm, CD8_T in green cluster, Fig. 1B);

ii. same cell type in different clusters (Tcm in yellow, orange, green and red clusters, Fig. 1B).

In the gut immune atlas [2] cell type annotation is done using the transcription expression, at single cell level, of previously known cell type gene markers and not by CITE-seq [10], which would have produced more robust results. Thus, it is necessary to investigate if erroneous assignment of the

cell type is the reason why different cell types are associated to the same cluster (Fig. 1B, red and green clusters). It is also necessary to evaluate if Tcm cells present in different clusters (Fig. 1B yellow, orange, and green clusters) are transcriptionally different, thus representing sub-populations of the same cell type.

3.3 Sparsely connected autoencoders for pseudo-bulk generation

The function *autoencoder4pseudoBulk* transforms each of the clusters in a RNAseq pseudo-bulk, which can be used to depict cluster specific genes. The *permutation* parameter indicates the number pseudo-bulks, which are generated.

```
library(rCASC)
WD = "/mnt/cold1/calogero/5-lung_cell-lines/scRNAseq_5-lung_cell-
lines/VandE"
SEPARATOR=","
SCRATCH="/scratch"
CLS=12
PROJECTNAME="LUNG5CLS"

CLSOUT="/mnt/cold1/calogero/5-lung_cell-lines/scRNAseq_5-lung_cell-
lines/VandE/Results/VandE/12/VandE_clustering.output.csv"

autoencoder4pseudoBulk(group="docker",
    scratch.folder=SCRATCH,
    file=paste(getwd(),"VandE.csv", sep="/"),
    separator=SEPARATOR,
    permutation=20,
    nEpochs=1000,
    projectName=PROJECTNAME,
    bN=CLSOUT
)
```

The output of *autoencoder4pseudoBulk* function is a file called total.csv, located in the folder permutation. As part of the output, it is also generated a tSne plot of the pseudo-bulks for the RNA-5c dataset, Fig. 2.

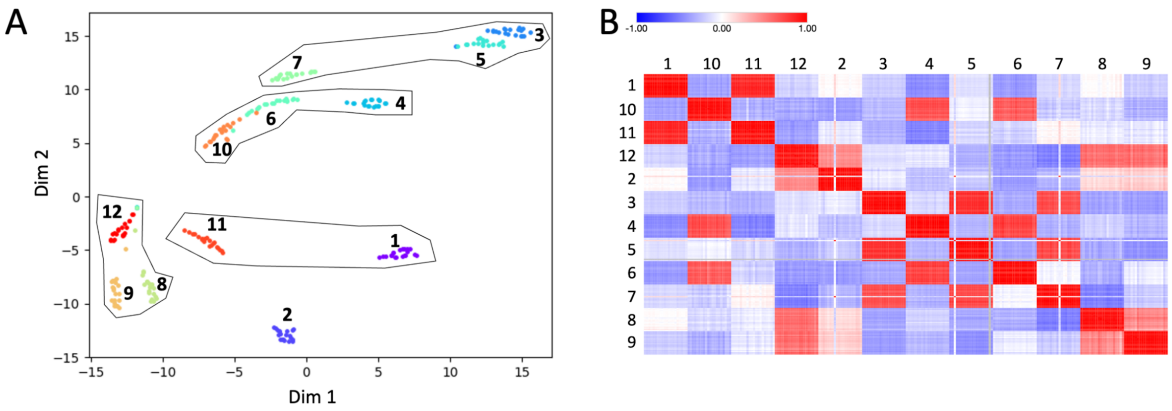

FIG. 2

The pseudo-bulks tSne plot, Fig. 2A, provides a clear description of the cell line heterogeneity, i.e. HCC827 (cluster 2) is homogeneous, as instead all the other cell lines show a certain amount of heterogeneity. Similarity analysis among clusters, using Pearson correlation, Fig. 2B, indicates high similarity among clusters belonging to the same cell line, but in the case of cluster 12, it also shows a certain similarity with cluster 2, which belong to a different cell line.

3.3 ANOVA-like on RNA-5c pseudo-bulks data

To depict genes specific to each cluster, we can perform differential expression analysis using edgeR ANOVA-like [11]. Pseudo-bulk data are converted in CPM (see **Note 6**), to provide a better similarity to RNAseq expression data, but retaining sample to sample variability.

```
counts2cpmAE <- function(fileIN, sep = ",", prefix){
      tmp <- read.table(fileIN, sep=sep, header=T)
      tmp <- t(tmp)
      tmp <- as.data.frame(tmp)
      col.names <- as.character(tmp[1,])
      tmp <- tmp[2:dim(tmp)[1],]
      names(tmp) <- col.names
      tmp <- t(tmp)
      tmp <- as.data.frame(tmp)
      col.sum <- apply(tmp, 1, sum)
      tmp1 <- tmp/col.sum
      tmp1 <- t(tmp1)
      tmp1 <- tmp1 * 1000000
      write.table(tmp1, paste(prefix, "cpm.csv", sep="_"), sep=",",
col.names=NA)
      write.table(log2(tmp1 + 1), paste(prefix, "log2cpm.csv", sep="_"),
sep=",", col.names=NA)
}
sample.folder="VandE"
pbAE.folder="LUNG5CLS"
```

```
dir(paste("./",sample.folder,"/Results/",pbAE.folder,"/permutation/",sep=
""))
whereis.total=paste("../",sample.folder,"/Results/",pbAE.folder,"/permuta
tion/total.csv",sep="")
prefix=pbAE.folder
counts2cpmAE(fileIN=whereis.total, sep=",", prefix=prefix)
```

To perform ANOVA-like, a reference sample, is made by averaging all the clusters.

```
#Ordering columns, replicates of the same clusters are placed near to
each other
logcpm <- read.table("LUNG5CLS_log2cpm.csv", sep=",", header=T,
row.names=1)
tmp.n <- sapply(strsplit(names(logcpm), "\\."), function(x)x[1])
logcpm <- logcpm[,order(tmp.n)]
write.table(logcpm, "LUNG5CLS_log2cpm_ordered.csv", sep=",",
col.names=NA)

cpm <- read.table("LUNG5CLS_cpm.csv", sep=",", header=T, row.names=1)
tmp.n <- sapply(strsplit(names(cpm), "\\."), function(x)x[1])
cpm <- cpm[,order(tmp.n)]
write.table(gcpm, "LUNG5CLS_cpm_ordered.csv", sep=",", col.names=NA)

#building a reference groups for ANOVA-like
tmp.n <- sapply(strsplit(names(cpm), "\\."), function(x)x[1])
tmp.nu <- unique(tmp.n)
tmp.lst <- list()
for(i in tmp.nu){
  if(i == "X1"){
    tmp.lst[[i]] <- grep('^X1$', tmp.n)
  } else{
    tmp.lst[[i]] <- grep(i, tmp.n)
  }
}
tmp.lst <- as.data.frame(tmp.lst)

ref.cl <- list()
for(i in 1:dim(tmp.lst)[1]){
  clmns <- as.numeric(unlist(tmp.lst[i,]))
  ref.cl[[i]] <- trunc(apply(cpm[,clmns], 1, mean))
}
ref.cl <- as.data.frame(ref.cl)
names(ref.cl) <- paste("ref", seq(1,dim(tmp.lst)[1]), sep=".")
names(ref.cl) <- paste(names(ref.cl), "Cov.0", sep="_")
ref.cl.na <- apply(ref.cl, 2, function(x)(sum(is.na(x))/length(x)))
ref.cl <- ref.cl[,which(ref.cl.na < 0.2)]

for(i in 1:dim(tmp.lst)[2]){
  names(cpm)[tmp.lst[,i]] <- paste(names(cpm)[tmp.lst[,i]], "_Cov", i,
sep="")
}
#Adding covariates to column names
covar <- sapply(strsplit(names(cpm), "_"), function(x)x[2])
cpm <- cpm[,order(covar)]

#Removing columns with more than 20% of 0s, these columns refer to low
SCA quality models
cpm.cl.na <- apply(cpm, 2, function(x)(sum(is.na(x))/length(x)))
cpm <- cpm[,which(cpm.cl.na < 0.2)]
```

```
#Formatting rownames to be compatible with docker4seq functions
symb <- sapply(strsplit(rownames(cpm.new), "\\."), function(x)x[2])
ensembl <- sapply(strsplit(rownames(cpm.new), "\\."), function(x)x[1])
rownames(cpm.new) <- paste(symb, ensembl, sep=":")
cpm.new <- data.frame(ref.cl, cpm)
write.table(cpm.new, "cpm_new.txt", sep="\t", col.names=NA)
#Running ANOVA-like
library(docker4seq)
anovaLike(
  group = "docker",
  file=paste(getwd(), "cpm_new.txt", sep="/"),
  logFC.threshold = 1,
  FDR.threshold=-0.1,
  logCPM.threshold = 4,
  plot = FALSE
)

anova.results <- read.table("ANOVAlike_cpm_new.txt", sep="\t", header=T,
row.names=1)
anova.results <- anova.results[which(anova.results$FDR <= 0.1),]
names(anova.results) <- gsub("groupsCov", "cl",names(anova.results))

write.table(anova.results, "filtered_ANOVAlike_cpm_new.txt", sep="\t",
col.names=NA)
write.table(anova.results[,grep("cl", names(anova.results))],
"filtered_ANOVAlike_cpm_new_4_clustering.txt", sep="\t", col.names=NA)
```

The output of this analysis are two files: filtered_ANOVAlike_cpm_new.txt, which contains the overall result of clusters comparison with respect to the reference set (see **Note 7**), and the file filtered_ANOVAlike_cpm_new_4_clustering.txt (see **Note 8**), which can be used to perform a hierarchical clustering using online tools, e.g. Morpheus from Broad Institute (https://software.broadinstitute.org/morpheus/).

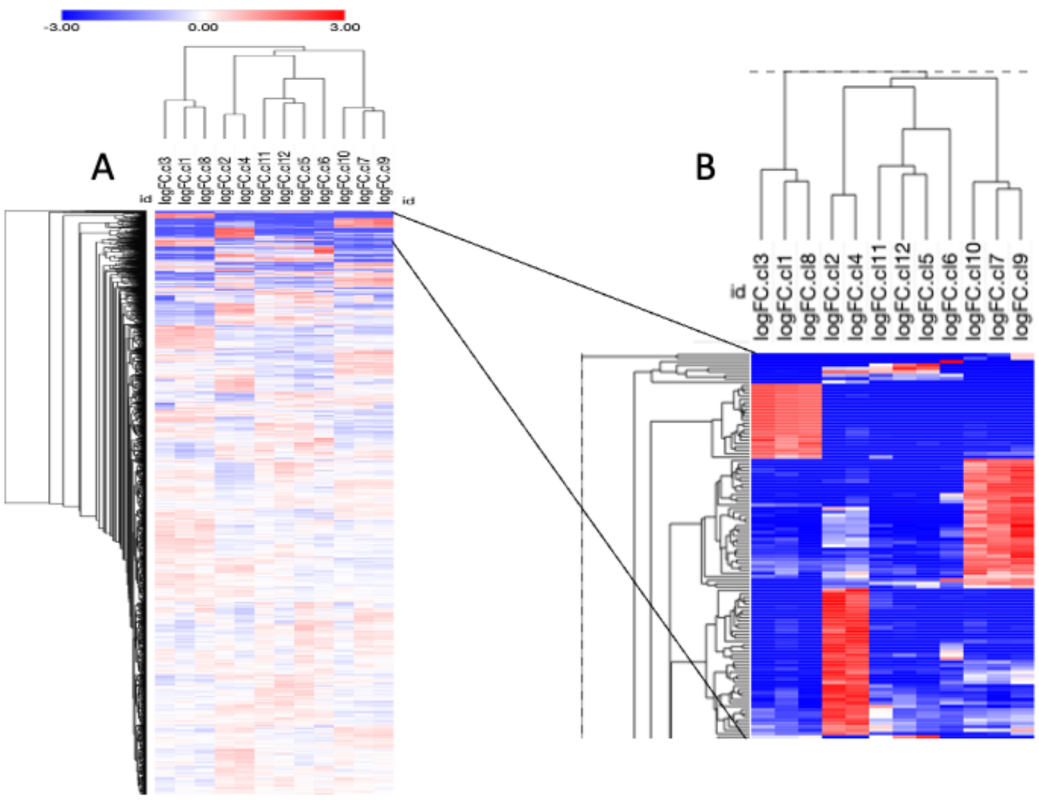

FIG. 3

In Fig. 3, are shown the results of the hierarchical clustering performed on pseudo-bulks from the RNA-5c analysis. Notably, cluster specific genes, Fig. 3B, show a different cluster-to-cluster similarity with respect to those depicted using the full transcriptome, Fig. 1A.

3.4 DESeq2 on pseudo-bulks data from Gut Immune Atlas sample 390c.

In sample 390c from gut immunocell atlas, Fig. 1B, we observe that central-memory T cells (Tcm) are spread over four clusters: red, orange, yellow and green. The task of this section is to understand how Tcm from different clusters are significantly transcriptionally different.

```
#Editing clustering.output file to create a set of new clusters
encompassing only Tcm derived respectively from red, orange, yellow and
green cluster.
cls.df <-
read.table("/data/390c/Results/saver_390c_wctype/11/saver_390c_wctype_clu
stering.output.csv", sep=",", header=T, row.names=1)
#red cluster cl1
#orange cluster cl2
#yellow cluster cl3
#green cluster cl5

#Ordering cells based on the belonging cluster.
cls.df <- cls.df[order(cls.df$Belonging_Cluster),]
```

```
#stripping cell type annotation from the rowname
ctype <- sapply(strsplit(rownames(cls.df),"\\."), function(x)x[2])
cls.df$ctype <- ctype

#creating fake clusters for Tcm in cluster 1, 2, 3 and 5.
cls.df$Belonging_Cluster[which(cls.df$ctype == "Tcm" &
cls.df$Belonging_Cluster == 1)] <- 12
cls.df$Belonging_Cluster[which(cls.df$ctype == "Tcm" &
cls.df$Belonging_Cluster == 2)] <- 13
cls.df$Belonging_Cluster[which(cls.df$ctype == "Tcm" &
cls.df$Belonging_Cluster == 3)] <- 14
cls.df$Belonging_Cluster[which(cls.df$ctype == "Tcm" &
cls.df$Belonging_Cluster == 5)] <- 15
write.table(cls.df, "/data/390c/saver_390c_wctype_clustering.output.csv",
col.names=NA, sep=",")

library(rCASC)
autoencoder4pseudoBulk(
  group = "docker",
  scratch.folder="/scratch",
  file="/data/390c/saver_390c_wctype.csv",
  separator=",",
  permutation=20,
  nEpochs=1000,
  patiencePercentage = 5,
  seed = 1111,
  projectName="Tcm",
  bN="/data/390c/saver_390c_wctype_clustering.output.csv",
  lr = 0.01,
  beta_1 = 0.9,
  beta_2 = 0.999,
  epsilon = 1e-08,
  decay = 0,
  loss = "mean_squared_error",
  regularization = 10
)
```

The *autoencoder4pseudoBulk* function result table total.csv is in ~/Results folder/Tcm/permutation/ folder.

To detect the presence of cluster specific genes, pseudo-bulk data are transformed in CPM. Then, pseudo bulk for red, orange ,yellow and green clusters are analyzed with DESeq2 [12], performing all possible permutations. The gene names of the differential expressed genes, depicted from each comparison, are then combined, and used to filter the table encompassing all the pseudo-bulks for the red, orange, yellow and green cell clusters.

```
# Transforming total.csv autoencoder4pseudoBulk result.
setwd("/data/390c/Results/Tcm/permutation")
counts2cpmAE <- function(fileIN, sep = ",", prefix){
      tmp <- read.table(fileIN, sep=sep, header=T)
      tmp <- t(tmp)
      tmp <- as.data.frame(tmp)
      col.names <- as.character(tmp[1,])
      tmp <- tmp[2:dim(tmp)[1],]
```

```
      names(tmp) <- col.names
      tmp <- t(tmp)
      tmp <- as.data.frame(tmp)
      col.sum <- apply(tmp, 1, sum)
      tmp1 <- tmp/col.sum
      tmp1 <- t(tmp1)
      tmp1 <- tmp1 * 1000000
      write.table(tmp1, paste(prefix, "cpm.csv", sep="_"), sep=",",
col.names=NA)
      write.table(log2(tmp1 + 1), paste(prefix, "log2cpm.csv", sep="_"),
sep=",", col.names=NA)
}
counts2cpmAE(fileIN="total.csv", sep=",", prefix="tcm")
cpm <- read.table("tcm_cpm.csv", sep=",", header=T, row.names=1)
cpm <- trunc(cpm)
#setting rownames compatibile with docker4seq
cpm.embl <- sapply(strsplit(rownames(cpm), "\\."), function(x)x[1])
cpm.symb <- sapply(strsplit(rownames(cpm), "\\."), function(x)x[2])
rownames(cpm) <- paste(cpm.symb, cpm.embl, sep=":")

red <- cpm[,grep("X12\\.", names(cpm))]
names(red) <- sub("^X", "red",names(red))
orange <- cpm[,grep("X13\\.", names(cpm))]
names(orange) <- sub("^X", "orange",names(orange))
yellow <- cpm[,grep("X14\\.", names(cpm))]
names(yellow) <- sub("^X", "yellow",names(yellow))
green <- cpm[,grep("X15\\.", names(cpm))]
names(green) <- sub("^X", "green",names(green))

#Calculating cluster-specific differentially expressed genes, |log2FC|≥2
and adjusted p-value ≤ 0.1. The thresholds used for differential
expression must be defined by the user. (see Note 9)
library(docker4seq)
de.pair <- function(dataset.ref, dataset.query, prefix){
  names(dataset.ref) <- paste(names(dataset.ref), "Cov.1", sep="_")
  names(dataset.query) <- paste(names(dataset.query), "Cov.2", sep="_")
  #DE
  tmp <- data.frame(dataset.ref, dataset.query)
  write.table(tmp, "comparison.txt", sep="\t", col.names=NA)

  wrapperDeseq2(
    output.folder=getwd(),
    group = "docker",
    experiment.table="comparison.txt",
    log2fc = 2,
    fdr = 0.1,
    ref.covar = "Cov.1",
    type = "gene",
    batch = FALSE
  )
  file.rename("log2normalized_counts.txt",
paste(prefix,"log2normalized_counts.txt", sep="_"))
  file.rename("DEfiltered_gene_log2fc_2_fdr_0.1.txt",
paste(prefix,"DEfiltered_gene_log2fc_2_fdr_0.1.txt", sep="_"))
  file.rename("genes2GO.txt", paste(prefix,"genes2GO.txt", sep="_"))
  file.rename("DEfull.txt", paste(prefix,"DEfull.txt", sep="_"))
}
#detecting genes differentially expressed among clusters pairs threshold
|log2FC| >= 2, adj.pval <= 0.1
de.pair(dataset.ref=red, dataset.query=orange, prefix="red_orange")
de.pair(dataset.ref=red, dataset.query=yellow, prefix="red_yellow")
```

```
de.pair(dataset.ref=red, dataset.query=yellow, prefix="red_green")
de.pair(dataset.ref=orange, dataset.query=yellow, prefix="orange_yellow")
de.pair(dataset.ref=orange, dataset.query=green, prefix="orange_green")
de.pair(dataset.ref=yellow, dataset.query=green, prefix="yellow_green")

files <- dir()
de.set <- files[grep("DEfiltered_gene_log2fc_2_fdr_0.1.txt", files)]
de.genes <- NULL
for(i in de.set){
  tmp.de <-read.table(i, sep="\t", header=T,row.names=1)
  de.genes <- c(de.genes, rownames(tmp.de))
}
de.genes <- unique(de.genes)
de.df <- data.frame(red, orange, yellow, green)
de.df <- de.df[which(rownames(de.df) %in% de.genes),]
write.table(de.df, "cluster_specific_genes.txt", sep="\t", col.names=NA)
write.table(log2(de.df + 1), "log2_cluster_specific_genes.txt", sep="\t",
col.names=NA)
```

In Fig. 4, it is shown the hierarchical clustering of the gene subset of Tcm pseudobulks encompassing only differentially expressed genes. The analysis, using enrichR [13] and the Panglao database [14], of the gene signature in the black frame, Fig. 4, assign the red and green cluster to Treg cells. Thus, using the above-described approach, cell type assignment of red and green clusters could be refined.

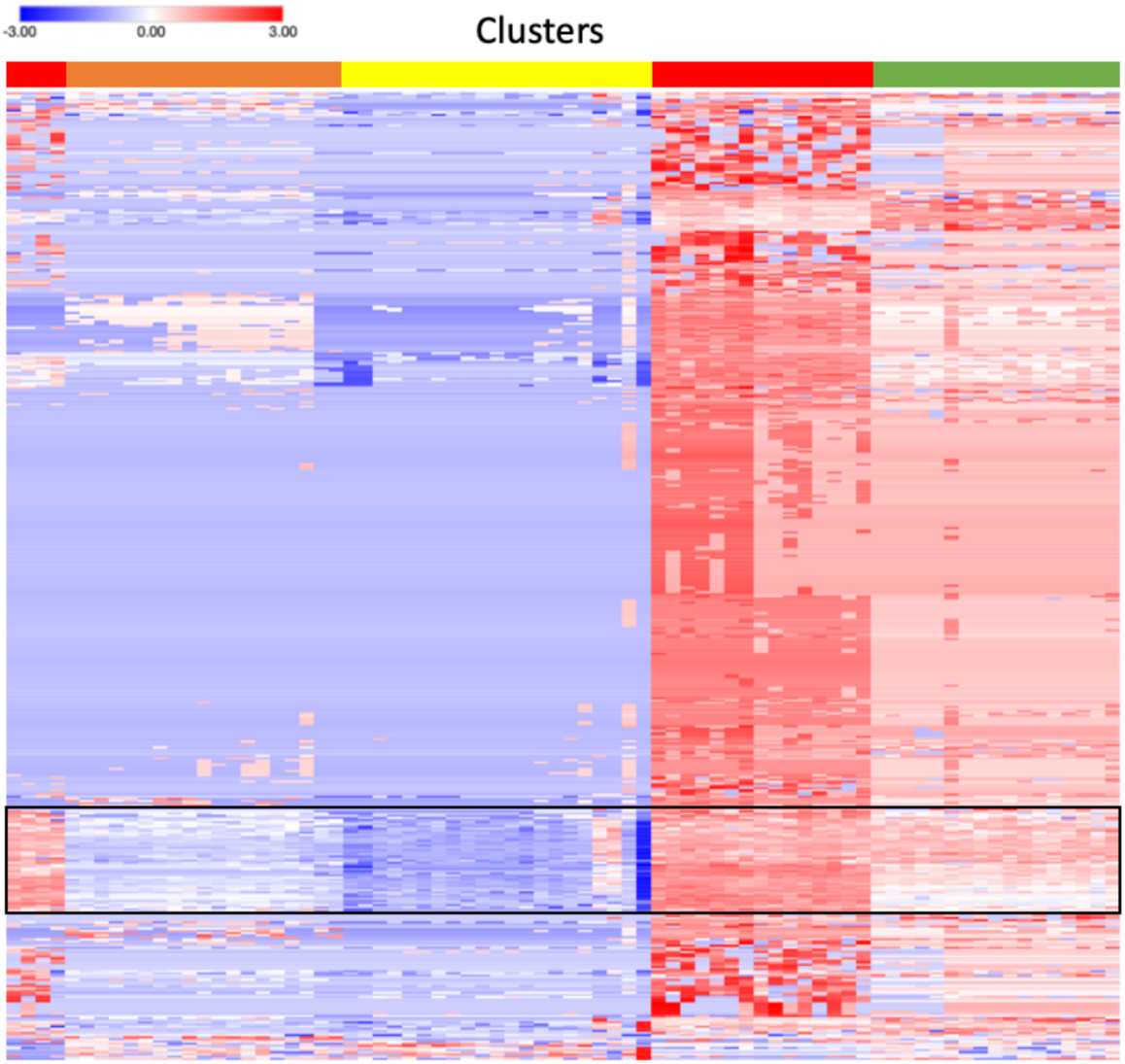

FIG. 4

3.5 COMET a tool to identify cluster specific genes

COMET is a statistical method designed to select panels of gene markers from single-cell RNA-seq data [6]. COMET uses the minimal Hypergeometric test to rank gene marker panels characterizing predefined cell populations (clusters) using single-cell RNA-seq data. COMET is available as standalone tool and it is implemented in rCASC [5], or as web application (http://www.cometsc.com/index). COMET is particularly effective in depicting few genes, from 1 to 4, which efficiently label a cell cluster. Thus, it results particularly interesting to identify set of genes to be used to further study and isolate cells belonging to a specific cluster.

```
WD = "/mnt/cold1/calogero/5-lung_cell-lines/scRNAseq_5-lung_cell-
lines/VandE"
SEPARATOR=","
SCRATCH="/scratch"
CLS=12
CC="/mnt/cold1/calogero/5-lung_cell-lines/scRNAseq_5-lung_cell-
line/annotated_GSM3618014_gene_count_cellCycle.csv"
CLSOUT="/mnt/cold1/calogero/5-lung_cell-lines/scRNAseq_5-lung_cell-
lines/VandE/Results/VandE/7/VandE_clustering.output.csv"

setwd(WD)
library(rCASC)
cometsc(
        group="docker",
        file=paste(getwd(), "VandE.csv", sep="/"),
        scratch.folder=SCRATCH,
        threads=CLS, X=0.15, K=1, counts="True",
        skipvis="False", nCluster=CLS, separator=SEPARATOR
)
```

The main output of COMET are two folders: outputdata and outputvis. Outputdata contains comma separated files with the genes ranked as the most important for characterizing the cluster under analysis. Outputvis, provides images for the 100 top most representative cluster-marker genes. In outputdata tables the most interesting columns are TP and TN, which ranks from 0 to 1. The best markers are those showing values near 1 for both TP and TN. TP equal to 1 means that the gene can discriminate very efficiently the cells in the cluster, under analysis, from the other cells. TN equal to 1 mean that the gene marker does not associate any cell outside the cluster to the cluster under analysis. Examples of COMET outputs are shown in Fig. 5 (see **Note 10**).

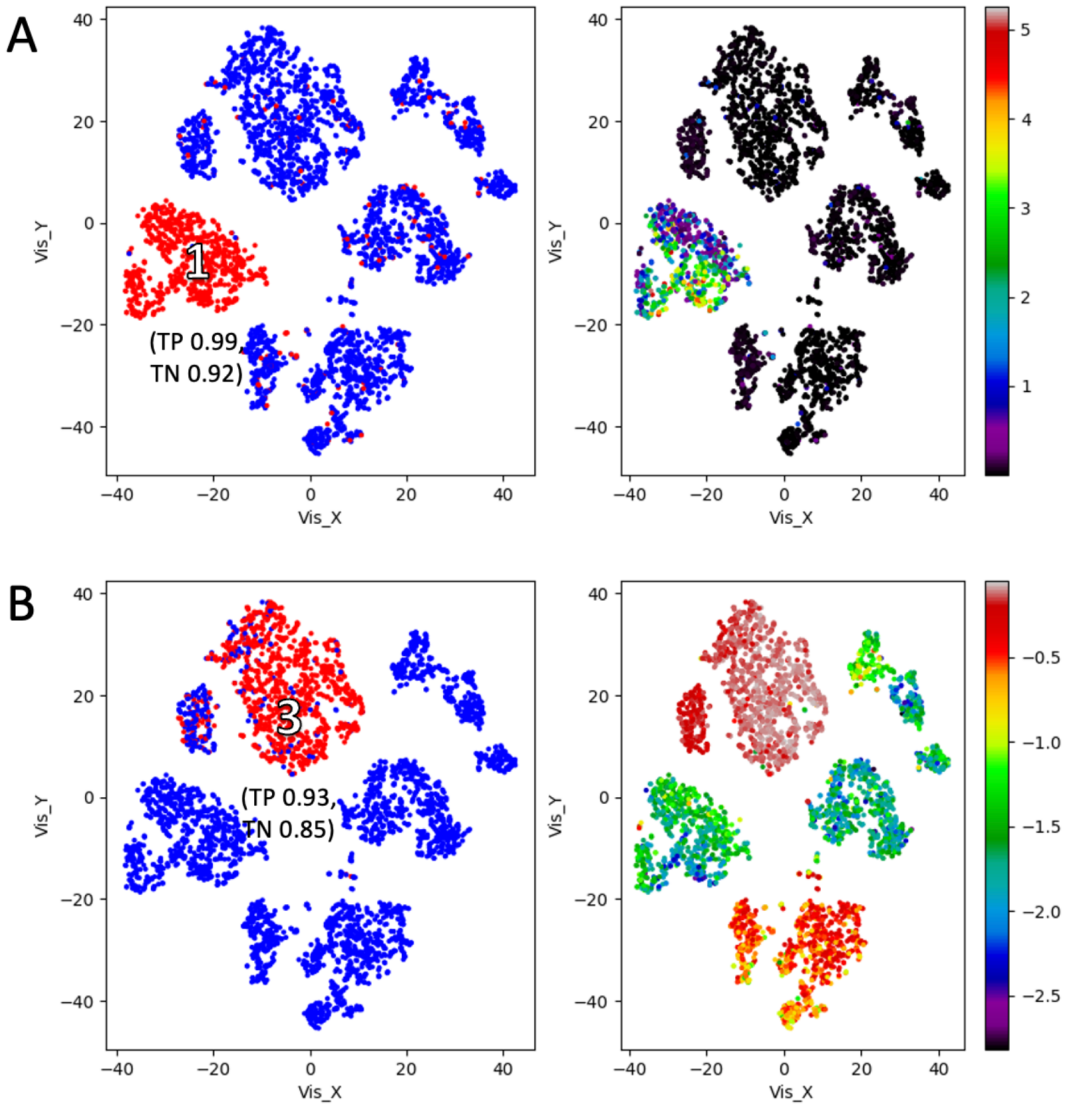

FIG.5

It is important to note that, in some clusters, many of the top ranked marked are defined as “NEGATION” indicating that the expression of that genes is higher in all clusters, but in the cluster under consideration. Usually, the presence of many expression negation genes is an indication that the cluster is not so different from the other clusters, since it is not possible to identify marker genes specifically expressed in the cluster under analysis. In these cases, it would be useful to perform autoencoder analysis (see Paragraph. 3.4) to identify which are the most similar clusters.

4 Note

1. The installation of docker is required. Docker is available for Windows, Mac and UNIX infrastructures at https://docs.docker.com/get-docker/.
2. R installation is required (https://cran.r-project.org/). Specifically, it is required the installation of devtools library.
3. The gut atlas datasets are available at https://www.gutcellatlas.org/.
4. The description of all rCASC functions is available at https://kendomaniac.github.io/rCASC/reference/index.html. The execution of the "*downloadContainers()*" command is not mandatory, since the first time a new docker container is required for an analysis it is downloaded from the repbioinfo docker.io repository. The first time a rCASC function is used, it retrieves from docker.io/repbioinfo repository the docker image required for the analysis. Thus, internet connection is required.
5. CSS (cell stability score) describes cluster stability, by estimating the persistence of a cell in a specific cluster upon jackknife resampling. A detailed description of the CSS metric is available in rCASC paper supplementary data [5].
6. Usually, CPM are not used as input to estimate differential expression, however, in this specific case we are using CPM transformation to convert the output of the sparsely-connected-autoencoder in a measure that is more like to what is expected to be a gene expression dataset.
7. The reference dataset is at the denominator of the fold change, thus, genes that have a positive fold change are the genes that are representative of a specific cluster with respect to the average reference cluster.
8. After having loaded to Morpheus the dataset, do not perform any transformation, since the data is already representing variation of each cluster with respect to the reference set.
9. We suggest testing combination of adjusted p-values (0.1, 0.05. 0.001) and absolute $\log_2$FC (1, 1.5, 2). Ideally the combination of adjusted p-values and |$\log_2$FC|, should produce less than 100 genes in total, to depict the main players for each cluster.

10. Detailed description of COMET algorithm can be found at https://hgmd.readthedocs.io/en/latest/

References

1. Yu L, Cao Y, Yang JYH, Yang P (2022) Benchmarking clustering algorithms on estimating the number of cell types from single-cell RNA-sequencing data. Genome Biol 23 (1):49. doi:10.1186/s13059-022-02622-0
2. James KR, Gomes T, Elmentaite R, Kumar N, Gulliver EL, King HW, Stares MD, Bareham BR, Ferdinand JR, Petrova VN, Polanski K, Forster SC, Jarvis LB, Suchanek O, Howlett S, James LK, Jones JL, Meyer KB, Clatworthy MR, Saeb-Parsy K, Lawley TD, Teichmann SA (2020) Distinct microbial and immune niches of the human colon. Nat Immunol 21 (3):343-353. doi:10.1038/s41590-020-0602-z
3. Alessandri L, Ratto ML, Contaldo SG, Beccuti M, Cordero F, Arigoni M, Calogero RA (2021) Sparsely Connected Autoencoders: A Multi-Purpose Tool for Single Cell omics Analysis. Int J Mol Sci 22 (23). doi:10.3390/ijms222312755
4. Alessandri L, Cordero F, Beccuti M, Licheri N, Arigoni M, Olivero M, Di Renzo MF, Sapino A, Calogero R (2021) Sparsely-connected autoencoder (SCA) for single cell RNAseq data mining. NPJ Syst Biol Appl 7 (1):1. doi:10.1038/s41540-020-00162-6
5. Alessandri L, Cordero F, Beccuti M, Arigoni M, Olivero M, Romano G, Rabellino S, Licheri N, De Libero G, Pace L, Calogero RA (2019) rCASC: reproducible classification analysis of single-cell sequencing data. Gigascience 8 (9). doi:10.1093/gigascience/giz105
6. Delaney C, Schnell A, Cammarata LV, Yao-Smith A, Regev A, Kuchroo VK, Singer M (2019) Combinatorial prediction of marker panels from single-cell transcriptomic data. Mol Syst Biol 15 (10):e9005. doi:10.15252/msb.20199005
7. Tian L, Dong X, Freytag S, Le Cao KA, Su S, JalalAbadi A, Amann-Zalcenstein D, Weber TS, Seidi A, Jabbari JS, Naik SH, Ritchie ME (2019) Benchmarking single cell RNA-sequencing analysis pipelines using mixture control experiments. Nat Methods 16 (6):479-487. doi:10.1038/s41592-019-0425-8
8. Huang M, Wang J, Torre E, Dueck H, Shaffer S, Bonasio R, Murray JI, Raj A, Li M, Zhang NR (2018) SAVER: gene expression recovery for single-cell RNA sequencing. Nat Methods 15 (7):539-542. doi:10.1038/s41592-018-0033-z
9. Butler A, Hoffman P, Smibert P, Papalexi E, Satija R (2018) Integrating single-cell transcriptomic data across different conditions, technologies, and species. Nat Biotechnol 36 (5):411-420. doi:10.1038/nbt.4096
10. Stoeckius M, Hafemeister C, Stephenson W, Houck-Loomis B, Chattopadhyay PK, Swerdlow H, Satija R, Smibert P (2017) Simultaneous epitope and transcriptome measurement in single cells. Nat Methods 14 (9):865-868. doi:10.1038/nmeth.4380
11. Robinson MD, McCarthy DJ, Smyth GK (2010) edgeR: a Bioconductor package for differential expression analysis of digital gene expression data. Bioinformatics 26 (1):139-140. doi:10.1093/bioinformatics/btp616
12. Love MI, Huber W, Anders S (2014) Moderated estimation of fold change and dispersion for RNA-seq data with DESeq2. Genome Biol 15 (12):550. doi:10.1186/s13059-014-0550-8
13. Chen EY, Tan CM, Kou Y, Duan Q, Wang Z, Meirelles GV, Clark NR, Ma'ayan A (2013) Enrichr: interactive and collaborative HTML5 gene list enrichment analysis tool. BMC Bioinformatics 14:128. doi:10.1186/1471-2105-14-128
14. Franzen O, Gan LM, Bjorkegren JLM (2019) PanglaoDB: a web server for exploration of mouse and human single-cell RNA sequencing data. Database (Oxford) 2019. doi:10.1093/database/baz046

Figures legend

Fig. 1 Seurat clustering of scRNAseq data. A) RNA-5c dataset at 0.8 resolution for RNA5c. Clusters belonging to a specific cell line are delimited by a black continuous line. B) sample 390c at 0.6 resolution. The Figure contains the main type of immunological cells constituting each cluster. In case there are two or more cell types, associated to a cluster, bold indicates the most represented cell type. Cell types: B cells producing IgA (B IgA), delta/gamma T cell (dgT), CD8 T cells (CD8_T), central memory T cells (Tcm), T regulatory cells (Treg), monocytes (M), B memory cells (Bm), follicular B cells (follB).

Fig. 2 Pseudo bulks. A) tSne plot of RNA-5c pseudo-bulks generated by sparsely-connected-autoencoder. Pseudo-bulks belonging to a specific cell line are delimited by a black continuous line. B) Pearson-based similarity score of RNA-5c Z-score log2CPM pseudo-bulks.

Fig. 3 Hierarchical clustering for ANOVA-like perfomed on RNA-5c pseudobulk. A) Hierarchical clustering performed on the $\log_2$ fold change generated comparing each RNA-5c pseudo bulks with respect to a reference dataset given by means of the overall pseudobulks, Clustering parameters: Euclidean distance and average linkage. B) Zoom on the top of the clustering results.

Fig. 4 Hierarchical clustering for genes detected a differentially expressed comparing to each other the pseudobulks for Tcm in red, orange, yellow and green cluster for sample 390c. Clustering parameters: Euclidean distance and average linkage.

Fig. 5 Examples of COMET outputs. A) Example of a top ranked genes with optimal TP and TN scores. B) Example of expression negation, the gene is expressed at higher level in all sample, but the cluster under consideration.