RESOLV3D

An R package for
resolving single cell RNA sequencing data in 3 dimensions

Kulkarni A.& Konopka G.

November 2016

1 Introduction

Advances in single cell transcriptome sequencing have elucidated the underlying heterogeneity within cell populations with greater resolution than ever before. The ability to identify constituent cell types and subtypes, crucial in understanding the composite organ, has been regarded as one of the most significant insights to be gained from single cell RNA sequencing (scRNA-seq). Traditional methods to distinguish cell types are based on the expression of small sets of marker genes. Using scRNA-seq, classifications can now take into account global transcriptome similarity among the population of cells. We implemented a novel unbiased classification and clustering approach accompanied by an additional correlation step to identify cell types and marker genes for each cell type. Here, we present R package `RESOLV3D` for implementing this approach. This document serves as an introduction and tutorial for the use of `RESOLV3D` R package.

2 Before you start

2.1 Installation

Using `RESOLV3D` package requires few existing R packages including but not limited to `reshape2` (Wickham, 2007), `ggplot2` (Wickham, 2009), `randomcoloR` (Ammar, 2016), `Hmisc` (Harrell Jr., 2016), `Rtsne` (van der Maaten and Hinton, 2008; van der Maaten, 2014; Krijthe, 2015), `apcluster` (Frey and Dueck, 2007; Bodenhofer et al., 2011), `plot3D` (Soetaert, 2013a), `rgl` (Daniel Adler, 2013), `plot3Drgl` (Soetaert, 2013b) & `goseq` (Young et al., 2010). Make sure to have most recent R ($\geq$ 3.1) installed before installing dependencies. Follow instructions (below) to install the dependencies. Working *X11-forwarding* and a free software *ImageMagick* is required for one of the functions of this package to run properly. Instructions to setup *X11-forwarding* are available through `https://www.x.org/wiki/`. Follow instructions by visiting `https://www.imagemagick.org` to download, install and use *ImageMagick*.

```
# Install dependencies from CRAN
install.packages(c("ggplot2", "reshape2", "randomcoloR", "Hmisc",
   "Rtsne", "apcluster", "plot3D", "rgl"))
# Install dependencies from Bioconductor
source("https://bioconductor.org/biocLite.R")
biocLite("goseq")
```

Once the dependencies are installed successfully, install `RESOLV3D` package from source as below. `RESOLV3D` package will be made available to CRAN and/or Bioconductor in near future.

```
# Install RESOLV3D from source without invoking R
R CMD INSTALL RESOLV3D_0.1.0.tar.gz
```

```
# Install RESOLV3D from source invoking R
install.packages("path-to-source/RESOLV3D_0.1.0.tar.gz", type =
    "source", repos = NULL)
```

Once successfully installed, follow instructions to invoke R and load the `RESOLV3D` package.

```
# Invoke R with specified memory
R --max-ppsize=500000
# Load R package
library(RESOLV3D)
```

2.2 Citing `RESOLV3D`

The work underlying `RESOLV3D` package is in preparation for a manuscript. In the meantime, users are encouraged to cite it as below.

Kulkarni A. and Konopka G. (2016). `RESOLV3D`: An R package for resolving single cell RNA sequencing data in three dimensions. R package version 0.1.

3 Quick Start

If you have a tab-seperated single cell expression data matrix (rows as *genes* and columns as *cells*) for at least 100 cells (package will support less than 100 cells in near future) and a list of background genes ready, a quick run can save you some time. You will also need `runRESOLV3D.R` and `RESOLV3D.sh` scripts (provided with this package) for a quick run. The shell script requires seven command-line input parameters i.e. (i) tab-seperated data file, (ii) prefix for output, (iii) maximum iterations, (iv) expression cutoff, (v) genome-version, (vi) key type for genes & (vii) file with list of background genes to be provided in the same order as mentioned. Run the following command carefully replacing with appropriate input values.

```
#run following command from working directory
#make sure all four required files are available in the directory
#(a) input file
#(b) background genes list
#(c) runRESOLV3D.R and
#(d) RESOLV3D.sh

sh ./RESOLV3D.sh example_rpkm.txt testExample 5000 0.5 hg19
   geneSymbol background_genes.txt
```

4 Details

Following are the steps to analyze single cell RNA-seq data using RESOLV3D package in R. In addition, a wrapper R script and shell scripts are also provided to automate the analysis. Since, this package uses already existing packages, plot(s) can be customized as required using data generated by this package. Also, package will generate RData files to save intermediate data. These RData files can be used in future to make additional analysis or plot(s).

4.1 Input data

The typical output of any RNA-seq data analysis such as RPKM/CPM/TPM/UMI matrix where rows are *genes* and columns are *samples* is the best example of input data for RESOLV3D. In case of single cell RNA-seq data, *samples* are replaced by *cells*.

	$Cell_a$	$Cell_b$	$Cell_c$	$Cell_d$	$Cell_e$ ...
$Gene_1$	X_{a1}	X_{b1}	X_{c1}	X_{d1}	X_{e1}
$Gene_2$	X_{a2}	X_{b2}	X_{c2}	X_{d2}	X_{e2}
$Gene_3$	X_{a3}	X_{b3}	X_{c3}	X_{d3}	X_{e3}
$Gene_4$	X_{a4}	X_{b4}	X_{c4}	X_{d4}	X_{e4}
...	...	...	...	...	...

4.2 Set working directory & input parameters

Before starting the analysis using RESOLV3D package, set a working directory having the input data file(s). Also, set the input parameters such as input file (ifn), output prefix (ofp), maximum iterations (itr), expression cutoff (expcutoff), etc. as shown in an example below.

```
# Invoke R with specified memory and load library
R --max-ppsize=500000
library(RESOLV3D)
# Set working diretory
setwd("/path/to/working/directory/")
# Set input parameters
ifn <- "sample_rpkm.txt"
ofp <- "sample1"
itr <- 1000
expcutoff <- 0.5
```

4.3 Validate and filter data

The first step in the single cell RNA-seq data analysis using RESOLV3D is to validate and filter the dataset to remove genes and cells with no or very low expression.

Filtering step will remove the genes with expression value(s) less than user-provided cutoff. Also, cells are discarded if no gene(s) is expressed above cutoff. The intended file format is a matrix with rows as *genes* and columns as *cells*. This step returns a filtered data frame and also generates a histogram showing number of expressed genes per cell (Fig. 1).

```
# validate and filter raw dataset
newdt1 <- validatedata(ifn, ofp, as.numeric(expcutoff)) # or
newdt1 <- validatedata("sample_rpkm.txt", "sample1", 0.5)
```

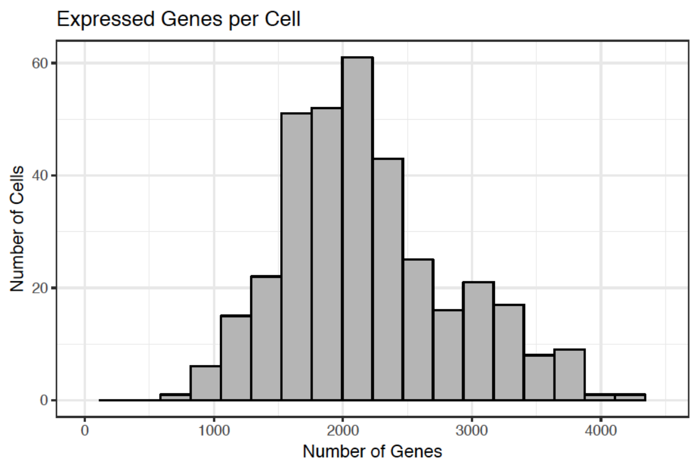


Figure 1: Histogram showing number of expressed genes per cell.

4.4 Resolve cells in three dimensions

This is the one-step and most critical part of analysis to resolve single cells in three dimensional space in an unbiased manner. First, the filtered dataset is used to separate the cells using t-Stochastic Neighbor Embedding (tSNE) in three dimensions. More the number of cells in the data, more number of iterations will properly resolve the cells. Post separating the cells in three dimensions, a combination of affinity propagation (AP) (Frey and Dueck, 2007; Bodenhofer et al., 2011) and k-means (KM) clustering identifies and assigns the cluster labels to the separated cells respectively. AP identified optimal number of clusters to be assigned from the resolved cells in X, Y and Z dimensions by passing the message between data points. And, KM clustering assigns the cluster labels to each cell. This step returns a data frame with cells resolved in three dimensions along with assigned cluster labels. This step also creates a 3-D scatter plot (Fig. 2) showing resolved cells along X, Y and Z coordinates and a barplot (Fig. 3) showing number of cells per cluster. This can be the time consuming step based on the number of cells to be analyzed. See *additional plots* section of this tutorial for alternative 3-D scatter plot options.

```
# resolve single cells in three dimensional space
newdt2 <- clustercells(newdt1, ofp, as.numeric(itr)) # or
newdt2 <- clustercells(newdt1, "sample1", 10000)
# save data as RData for future use, can also be saved as text file
save(newdt2, file="sample1_main.RData")
```

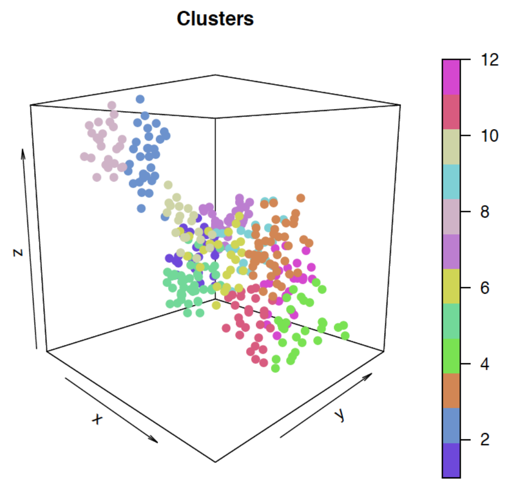


Figure 2: 3-D scatter plot showing cells resolved using RESOLV3D.

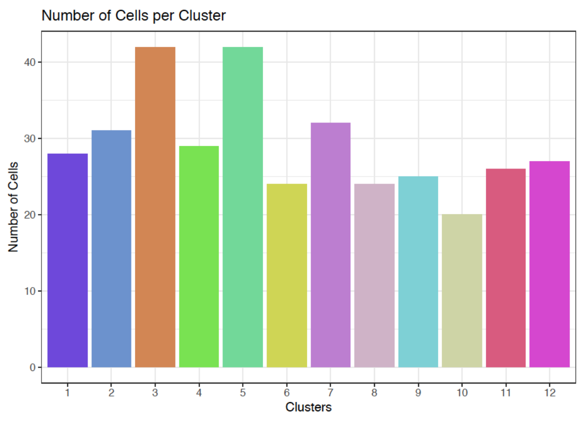


Figure 3: Barplot showing number of cells assigned per cluster using RESOLV3D.

4.5 Extract genes

This step extracts list(s) of genes specific to each detected cluster or cellular population. First, genes that are enriched in each cluster are identified with expression profile specific to cells of the cluster(s) under study against remaining cells (Fig. 4). Cluster-enriched genes are then tested for Spearman correlation. Significantly and highly correlated genes are then selected as cluster-specific genes. Cluster-specific gene lists are saved in the working directory and can be further used for gene ontology analysis to identify associated cell type.

```
# extract cluster-specific gene(s) or gene list(s)
extractgenes(newdt2)
```

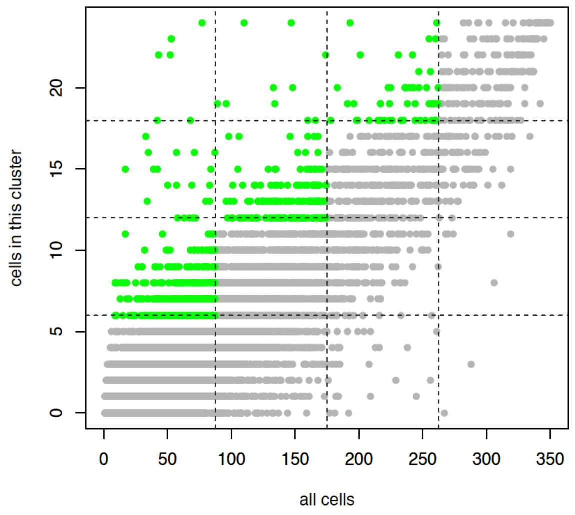


Figure 4: Scatter plot showing cluster-enriched genes (green) against genes expressed across remaining cells (grey).

4.6 Gene Ontology (GO) analysis

This step performs gene ontology (GO) analysis using gene list(s) generated by *extractgenes*. In order to perform GO analysis, a user-created background gene(s) list is required. This step returns a data frame with enriched GO terms and corresponding B&H adjusted p-value. Users are encouraged to filter the GO terms using B&H adjusted p-value to retrieve significant GO terms. These GO terms corresponding to each cluster-specific gene list(s) are an indicative of associated cell type. Alternative GO analysis method(s), e.g. ToppGene, are recommended in order to identify associated cell type.

```
# GO analysis
cluster1GO <- getsetgo("genes_cluster_1_postCor_genes_top50.txt",
   "background_genes.txt", ofp, "hg19", "geneSymbol")
save(cluster1GO, file="sample1_go.RData")
```

4.7 Create animation

This is an accessory function to create GIF animation of 3-D resolved cells. A working *X11-forwarding* and *ImageMagick* software is required for this function to work properly.

```
# create animation
animate(ofp)
```

4.8 Gene expression profile

This is an accessory function to create gene expression profile for gene(s) of interest or list of gene(s) across 3-D resolved cells. This function will create three types of plots for the user-provided gene(s) including (i) 3-D scatter plot (Fig. 5), (ii) bar plot (Fig. 6) and (iii) violin plot (Fig. 7) highlighting gene expression profile across resolved groups of cells. Additionally, user(s) can create 3-D scatter plot by rotating the viewing angle(s) to best represent the gene expression profile as shown in the *Additional plots* section of this tutorial below.

```
# create gene expression profile for gene of interest, e.g. STMN2
plotgene("STMN2", ofp)

# create gene expression profile using a gene list(s)
for(gene in genelist) {
plotgene(gene, ofp)
}
```

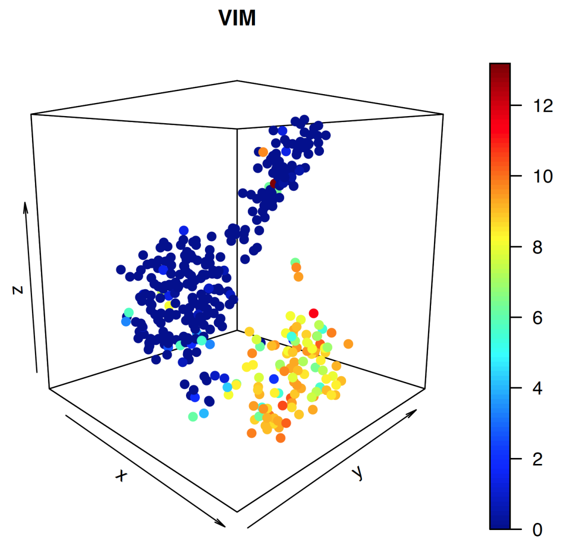


Figure 5: 3-D scatter showing gene expression profile across all clusters for gene **VIM**.

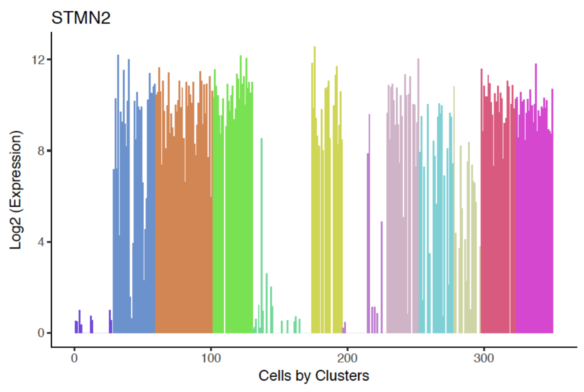


Figure 6: Bar plot showing gene expression profile across all clusters for gene **STMN2**.

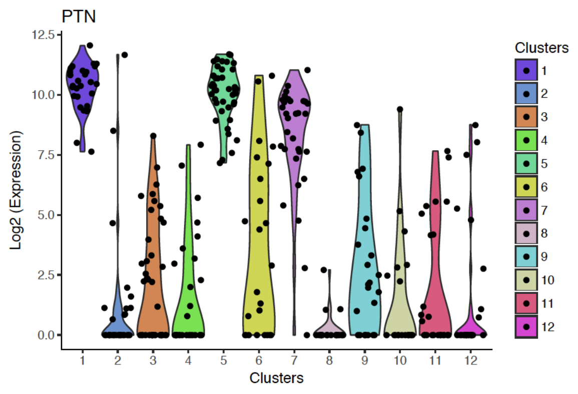


Figure 7: Violin plot showing gene expression profile across all clusters for gene **PTN**.

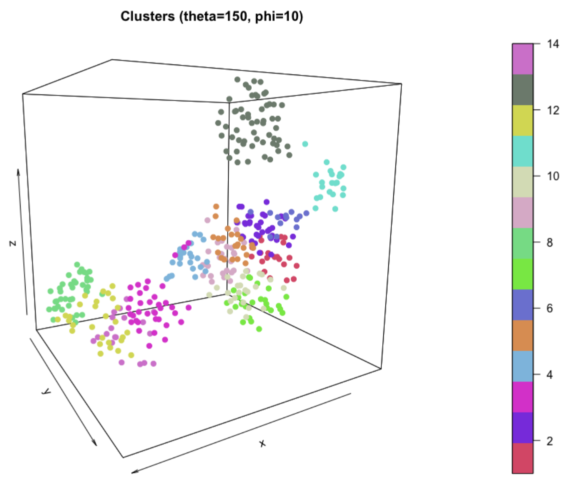
4.9 Additional plots

In addition to plots mentioned above in this tutorial, additional plots can be generated using the output data frame(s) such as `newdt2`. Below is an example (Fig. 8, left panel) to generate alternative 3-D scatter plot showing resolved groups of cells. Similarly, alternative viewing angles can be used to plot gene expression profile as a 3-D scatter plot is shown below (Fig. 8, right panel).

```
# additional 3-D scatter plot(s) with custom viewing angle(s)
# change the value(s) of "theta" or "phi" in "scatter3D" command
load("validatedata.RData")
load("clustercells.RData")
load("sample1_main.RData")
```

```
scatter3D(newdt2[,2], newdt2[,3], newdt2[,4], colvar = newdt2[,1], pch
   = 20, col = colpalette[1:max(optimalclusters)], cex = 1.5, theta =
   150, phi = 10, main = "Clusters (theta = 150, phi = 10)")

mygene <- "DLX2"
scatter3D(newdt2[,2], newdt2[,3], newdt2[,4], colvar =
   newdt2[,as.character(mygene)], pch = 20, cex = 1.5, theta = 150, phi
   = 10, main = paste(mygene, ", theta=150, phi=10", sep=""))
```



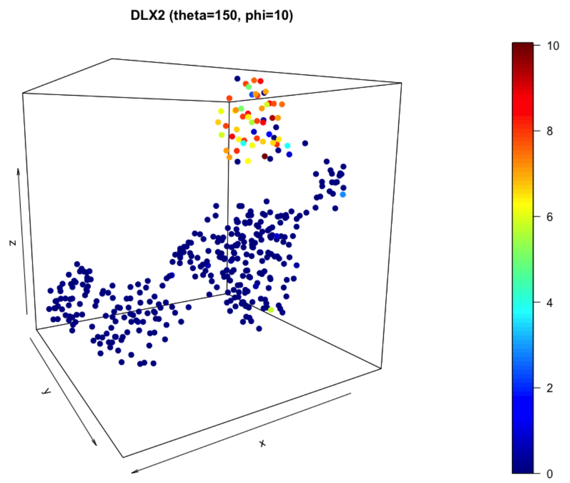


Figure 8: 3-D scatter plot showing different viewing angles for resolved clusters (top) and highlighting *DLX2* gene expression across resolved cells (bottom).

In addition, enriched GO terms can also be represented as a scatter plot using output of `getsetgo` (Fig. 9).

```
load("sample1_go.RData")
ggplot(cluster1GO, aes(x=Number_Input_Genes, y=-log10(Adj_p_val),
    label=GO_Term, size=-log10(Adj_p_val), colour=Category)) +
    geom_point() + geom_text(size = 2, colour = "black", vjust = -2) +
    xlim(0, max(Number_Input_Genes)) + ylim(0, max(-log10(Adj_p_val)))
```

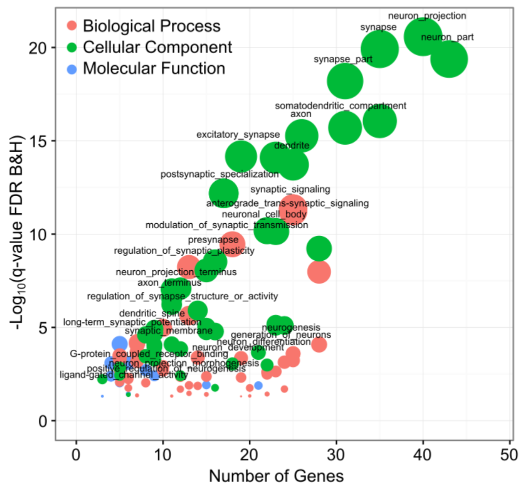


Figure 9: Scatter plot showing enriched GO terms.

References

Ammar, R. (2016). An r package for generating attractive and distinctive colors.

Bodenhofer, U., A. Kothmeier, and S. Hochreiter (2011). Apcluster: an r package for affinity propagation clustering. *Bioinformatics 27*, 2463–2464.

Daniel Adler, D. M. (2013). rgl: 3d visualization using opengl.

Frey, B. J. and D. Dueck (2007). Clustering by passing messages between data points. *Science 315*, 972–977.

Harrell Jr., F. E. (2016). Harrell miscellaneous.

Krijthe, J. H. (2015). *Rtsne: T-Distributed Stochastic Neighbor Embedding using Barnes-Hut Implementation.* R package version 0.11.

Soetaert, K. (2013a). plot3d: Tools for plotting 3-d and 2-d data.

Soetaert, K. (2013b). plot3drgl: Plotting multi-dimensional data using rgl.

van der Maaten, L. (2014). Accelerating t-sne using tree-based algorithms. *Journal of Machine Learning Research 15*, 3221–3245.

van der Maaten, L. and G. Hinton (2008). Visualizing high-dimensional data using t-sne. *Journal of Machine Learning Research 9*, 2579–2605.

Wickham, H. (2007). Reshaping data with the reshape package. *Journal of Statistical Software 21*(12), 1–20.

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis.* Springer-Verlag New York.

Young, M. D., M. J. Wakefield, G. K. Smyth, and A. Oshlack (2010). Gene ontology analysis for rna-seq: accounting for selection bias. *Genome Biology 11*, R14.