Clustering analysis of Retinal Bipolar Cell Drop-seq data

Introduction

This is an R Markdown script that outlines the key steps involved in identifying clusters of Bipolar Cell types from the *Vsx2*-GFP Drop-seq dataset published in Shekhar et al., "Comprehensive classification of retinal bipolar neurons by single-cell transcriptomics",*Cell*, 2016. The main input required is the expression matrix of transcript counts (genes x cells). Most of the steps were implemented on an Apple MacBook Pro (2.6 GHz, 16 GB of memory) using the RStudio IDE. However, certain computational or memory-intensive steps (e.g. Batch Correction or t-SNE embedding) were run on a computing cluster (Intel(R) Xeon(R) CPU, 2.67GHz, 100 GB of memory). These are flagged below, and for ease of implementation we provide output files from these steps that can be directly read in by the user. Background details of the methods are described in the **Supplementary Experimental Procedures** accompanying the paper. Before beginning this tutorial,

- please install the following packages in your local R library - `sva`, `igraph`, `ggplot2`, `grid`, `Matrix`
- please ensure that the accessory files - `class.R`, and `bipolar_data_Cell2016.Rdata` - are available in the your working directory in RStudio

First change to the directory containing the above files, and set it as your working directory. Next, load required packages and functions,

```
source("class.R")
```

Data preprocessing

Load the data file containing the expression matrix `bipolar_dge`, and the pre-calculated matrices `pca.scores`, `pca.load` and `tsne.y` (for memory/computation intensive steps).

```
load("bipolar_data_Cell2016.Rdata")
print(dim(bipolar_dge))
```

```
## [1] 24904 37122
```

```
bipolar_dge[1:5, 1:2]
```

```
##               Bipolar1_CCCACAAGACTA Bipolar1_TCGCCTCGTAAG
## 0610005C13Rik                     0                     0
## 0610007P14Rik                     0                     0
## 0610009B22Rik                     0                     1
## 0610009E02Rik                     0                     0
## 0610009L18Rik                     0                     0
```

The raw matrix thus consists of 24,904 genes and 44,994 cells. Next, remove libraries that contain more than 10% mitochondrially derived transcripts,

```
mt.genes = grep("mt-", rownames(bipolar_dge), value = TRUE)
cells.use = colnames(bipolar_dge)[colSums(bipolar_dge[mt.genes, ])/colSums(bipolar_dge) <
    0.1]
bipolar_dge = bipolar_dge[, cells.use]
dim(bipolar_dge)
```

```
## [1] 24904 37122
```

Initialize S4 object and explore data

Initialize single cell data as an S4 class object. Only cells where > 500 genes are detected are considered. Among the selected cells, only genes that are present in > 30 cells *and* those having > 60 transcripts summed across all the selected cells are considered,

```
#Execution time ~ 10-15 minutes
dsq.bip=scDrop(count.data=bipolar_dge)
dsq.bip=initialize(dsq.bip, min.genes = 500, min.cells = 30, min.counts=60)
```

```
## [1] "Initializing S4 object"
## [1] "Median normalizing counts and log-transforming"
## [1] "z-scoring each gene"
```

```
print(dim(dsq.bip@data))
```

```
## [1] 13166 27499
```

Thus the filtered matrix contains 13,166 genes and 27,499 cells. Our data consists of 6 experimental samples labeled "Bipolar1"-"Bipolar6". "Bipolar1"-"Bipolar4" are replicates from the first experimental batch, while "Bipolar5"-"Bipolar6" are replicates from the second experimental batch. Let's examine the number of cells from each of these samples,

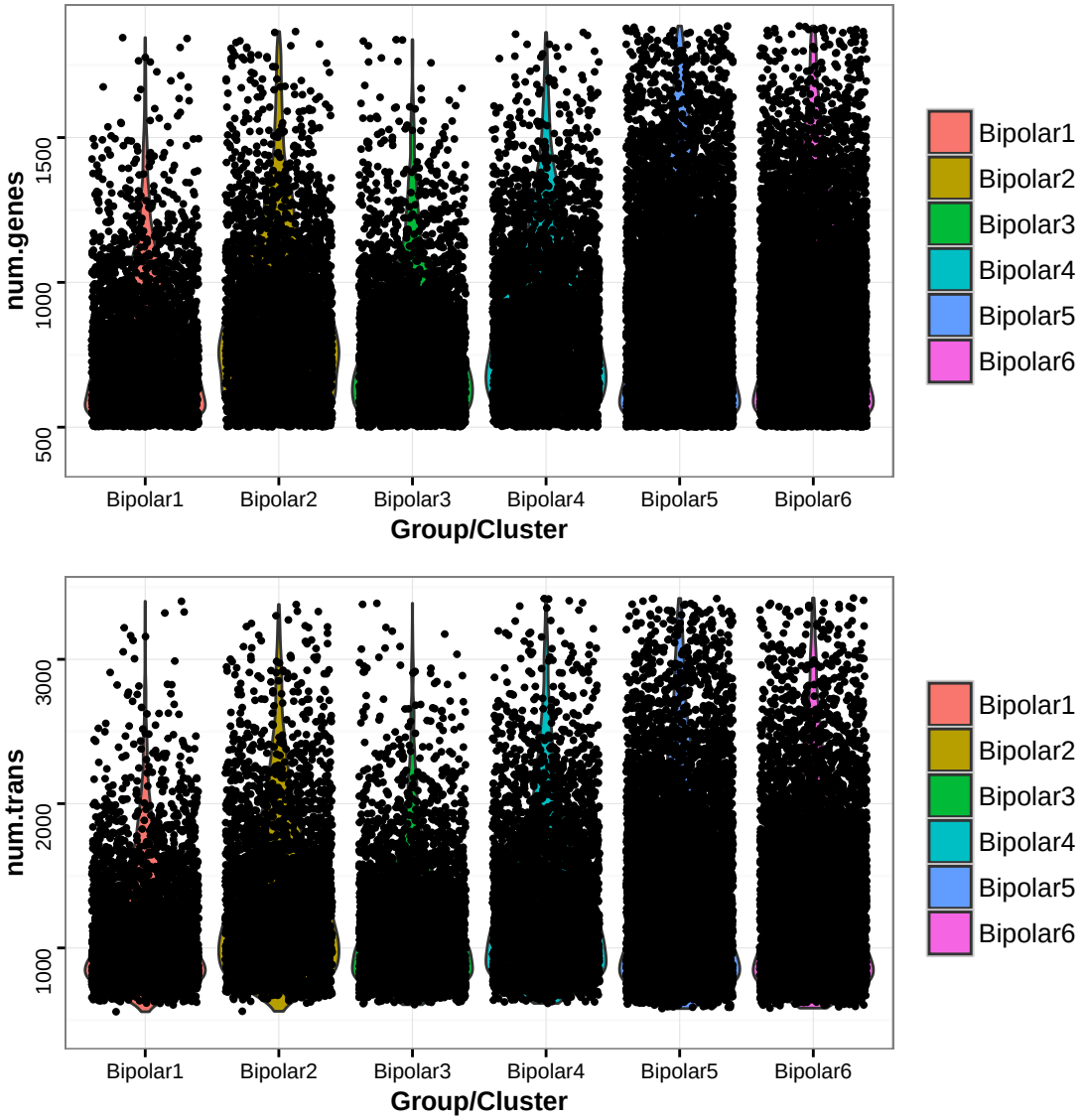
```
table(dsq.bip@meta$sample)
```

```
##
## Bipolar1 Bipolar2 Bipolar3 Bipolar4 Bipolar5 Bipolar6
##     3055     3419     3662     3851     7129     6383
```

We can examine the genes and transcripts per cell (**Figure S1A**)

```
violinplot(dsq.bip,c("num.genes","num.trans"))
```

```
## Loading required package: grid
```



Batch correction and PCA

Next, we perform batch correction using ComBat (Johnson et al., *Biostatistics*, 2007) and the batch-corrected expression matrix is then reduced using PCA, and the top 100 PCs are stored.

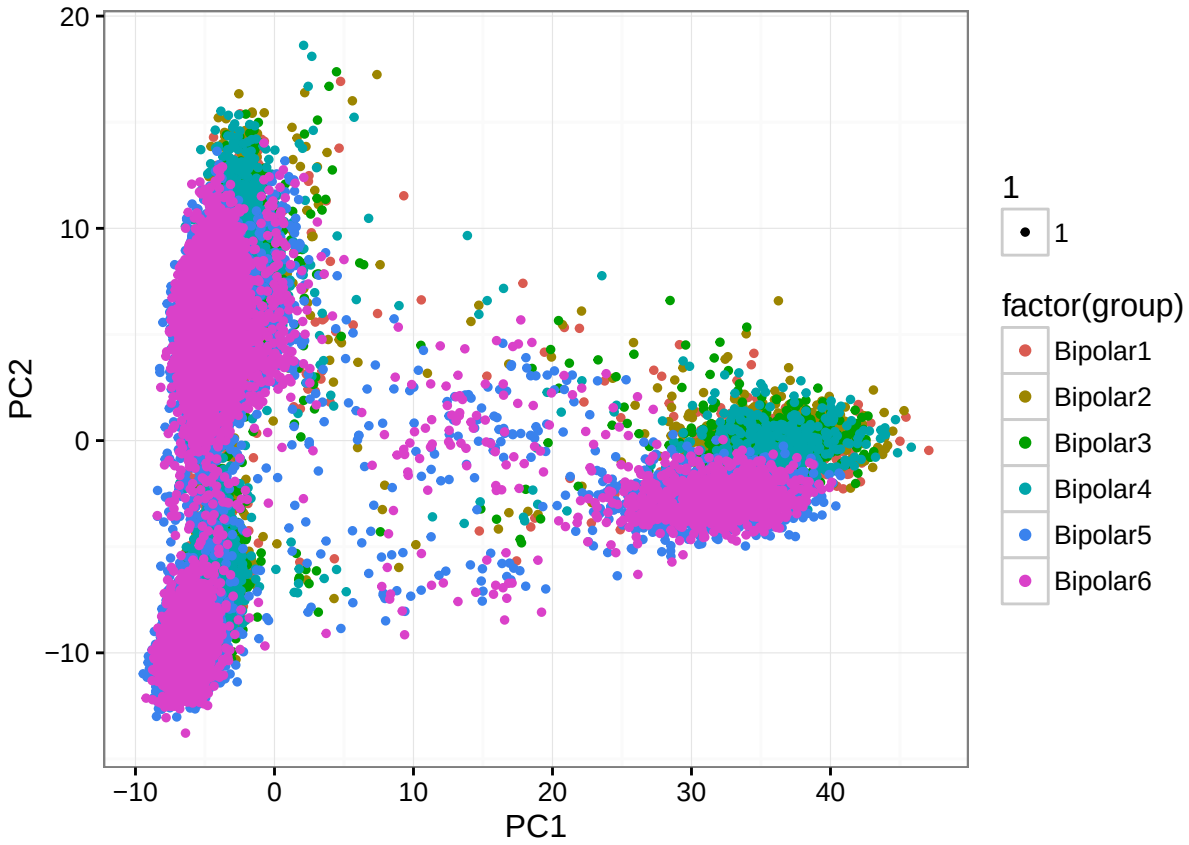
```
batchname = as.character(dsq.bip@meta$sample)
batchid = rep(1,length(batchname))
batchid[batchname=="Bipolar5"] = 2
batchid[batchname=="Bipolar6"] = 2
dsq.bip=doBatchCorrection(dsq.bip, batch.cov=batchid)
dsq.bip=doPCA(dsq.bip,pcs.store=100)
```

Both steps above are computation and/or memory intensive, and we recommend that they be run on a cluster with at least 30 GB memory. The batch corrected expression matrix is stored in `dsq.bip@batch.data`. The scores and the loadings from the PCA are stored in `dsq.bip@pca.scores` and `dsq.bip@pca.load`. For users that wish to proceed with the analysis without executing the above steps, we directly read in the PC scores and PC loadings corresponding to the top 100 PCs from the provided data in `bipolar_data_Cell2016.Rdata`,

```
dsq.bip@pca.scores = pca.scores
dsq.bip@pca.load = pca.load
```

Visualize PC-scores as a scatter plot (**Figure S1I**),

```
data.plot = dsq.bip@pca.scores
data.plot$group = dsq.bip@group
ggplot(data.plot, aes(x = PC1, y = PC2)) + geom_point(aes(colour = factor(group),
    size = 1)) + scale_color_hue(l = 55) + scale_size(range = c(1, 1)) + theme_bw()
```



Visualize in tSNE

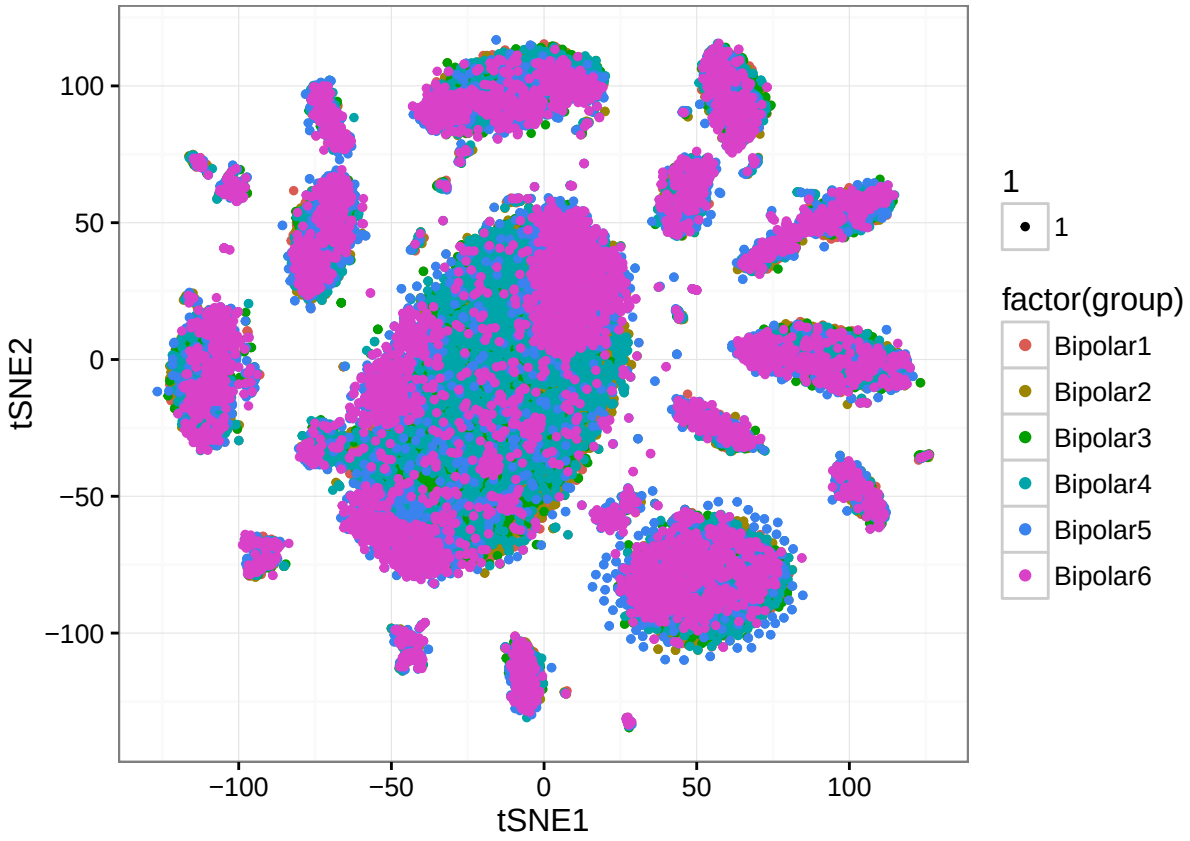
A permutation test (cf. **Methods and Resources, Figure S1J**) enabled us to determine the statistically significant number of PCs as 37. The scores of the cells corresponding to these PCs can be used to embed the single cells on a 2D map using t-distributed Stochastic Neighbor Embedding or tSNE, such that cells with similar gene expression have similar coordinates (van der Maaten and Hinton, *Journal of Machine Learning Research*, 2008). This was implemented in python for 2000 steps using the author's original code (https://lvdmaaten.github.io/tsne/) on a single computing node (36 hours running time), after disabling the initial PCA reduction step. Users can consider using Barnes Hut SNE (van der Maaten, *Journal of Machine Learning Research*, 2014), which is an approximate, faster alternative to t-SNE. Here we directly read in the embedded values and store in the slot `dsq.bip@tsne.y`,

```
dsq.bip@tsne.y = tsne.y
```

Note that the above step is *entirely dispensable* as these coordinates are used strictly for visualization, not clustering. The exact coordinates also depend on the stochastic initialization, so users are likely to obtain

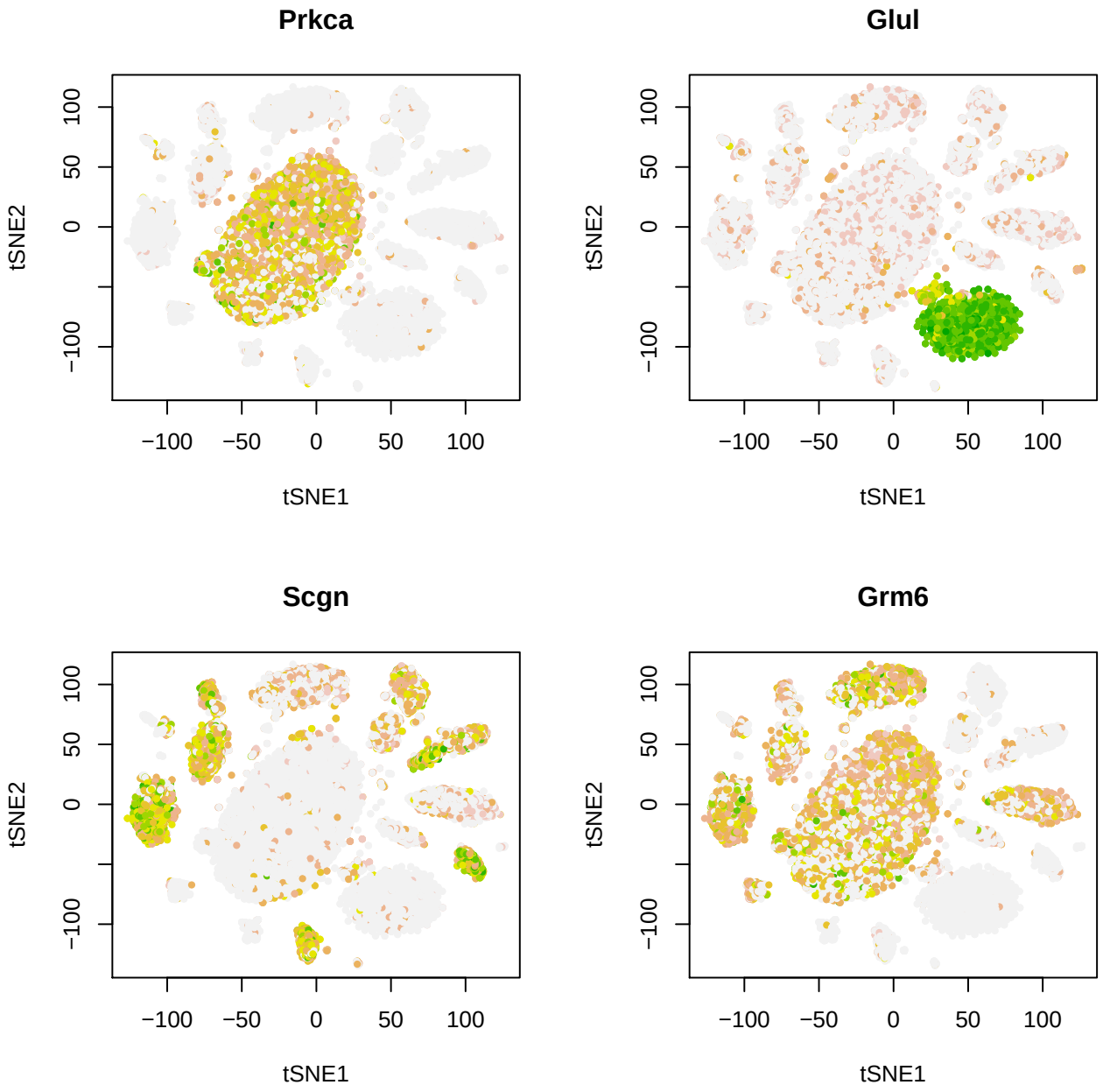
slightly different visualizations each time. Next, we visualize single cell expression based on tSNE coordinates (**Figure S1K**),

```
data.plot = dsq.bip@tsne.y
data.plot$group = dsq.bip@group
ggplot(data.plot, aes(x = tSNE1, y = tSNE2)) + geom_point(aes(colour = factor(group),
    size = 1)) + scale_color_hue(l = 55) + scale_size(range = c(1, 1)) + theme_bw()
```



As before, cells are colored based on their sample of origin. We can color each cell by the expression levels of marker genes,

```
gene.expression.scatter(dsq.bip, c("Prkca", "Glul","Scgn", "Grm6"))
```



Louvain-Jaccard Clustering

Thus, we can see that distinct point clouds on the tSNE map largely correspond to well-known cell types. These include rod bipolar cells (*Prkca*$^{+}$ *Scgn*$^{-}$), Muller Glia (*Glul*$^{+}$ *Prkca*$^{-}$ *Scgn*$^{-}$), several cone bipolar clusters (*Prkca*$^{-}$ *Scgn*$^{+}$), and among these, a subset of ON Cone Bipolar Clusters (*Prkca*$^{-}$ *Scgn*$^{+}$ *Grm6*$^{+}$). Next, we cluster the cells based on their PC scores using the Louvain-Jaccard method,

```
dsq.bip = doGraph_clustering(dsq.bip, pcs.use = 1:37, num.nn = 30, do.jaccard = TRUE,
    method = "Louvain")
```

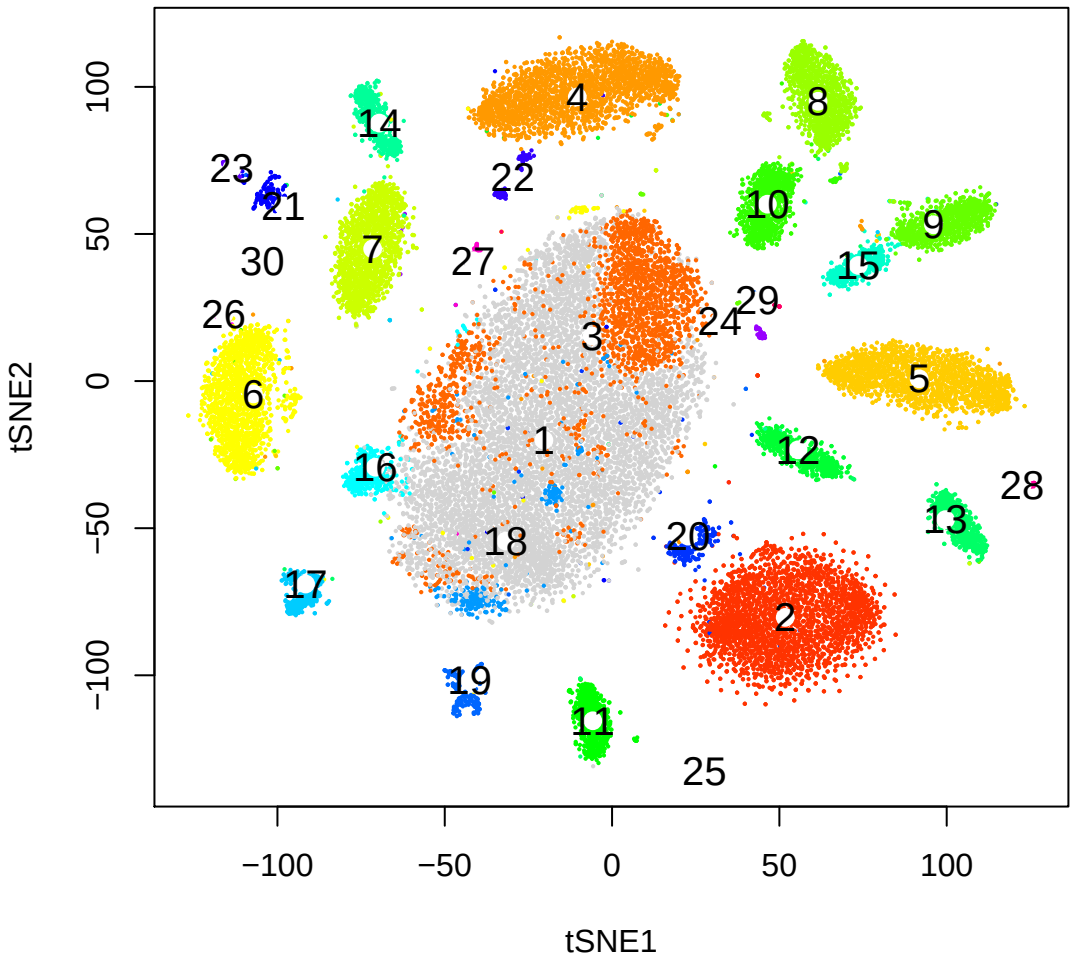
```
## [1] "Performing Louvain-Jaccard clustering. Using 30 nearest neighbors, and 37 PCs"
## [1] "Found nearest neighbors"
## [1] "Outputting clusters .."
```

```
length(table(dsq.bip@group))
```

```
## [1] 30
```

Indicating that 30 clusters are found. Next, visualize clusters on tSNE plot (**Figure S2A**),

```
plot.tsne(dsq.bip)
```



Differential expression analysis and merging of expression proximal clusters

We are now ready to perform differential expression analysis. First we extract matrices corresponding to the raw counts `Count.mat` and median normalized counts `TPM.mat`,

```
TPM.mat = exp(dsq.bip@data) - 1  #Recompute normalized TPM counts from log-transformed values
Count.mat = dsq.bip@count.data[rownames(dsq.bip@data), colnames(dsq.bip@data)]
```

Next, we examine the top genes enriched in cluster 6 in the previous tSNE map (note the cluster numbering is slightly different than **Figure 1C** as merging is yet to be performed),

```
markers.6 = markers.binom(dsq.bip, clust.1 = 6, effect.size = log(2), TPM.mat = TPM.mat,
    Count.mat = Count.mat)
head(markers.6, 10)
```

```
##        log.effect          pval nTrans_6 nTrans_rest
## Cck      4.280402  0.000000e+00    2.082       0.013
## Grin3a   3.408364 5.927952e-262    0.171       0.006
## Gm7854   3.055762 4.954127e-161    0.123       0.006
## Spock3   2.960223  0.000000e+00    0.496       0.029
## Lect1    2.783093  0.000000e+00    1.218       0.043
## Pvrl4    2.581887 1.232220e-175    0.171       0.012
## Lhx3     2.360652  0.000000e+00    0.395       0.039
## Ephx4    2.329716 2.207310e-115    0.123       0.011
## Cdh18    2.329160 1.287491e-126    0.140       0.013
## Smoc2    2.321759 4.447017e-122    0.150       0.013
```

The results suggest that this corresponds to BC6. Among the top 10 enriched markers (sorted by effect size) include *Cck* and *Lect1*, both of which are highly enriched in this cluster compared to the rest of the cells (compare the columns `nTrans_6` and `nTrans_rest`). Additionally enriched markers include *Scgn* (a broad cone BC marker), *Lhx3* (also enriched in BC1B and BC2). We can also evaluate the differentially enriched genes across two clusters (e.g. 1 and 2),

```
markers.1vs2 = markers.binom(dsq.bip, clust.1 = 1, clust.2 = 2, effect.size = log(2),
    TPM.mat = TPM.mat, Count.mat = Count.mat)
head(markers.1vs2[order(markers.1vs2$log.effect, decreasing = FALSE), ], 10)
```

```
##          log.effect pval nTrans_1 nTrans_2
## Cyba           -Inf    0    0.000    0.127
## Pros1          -Inf    0    0.000    0.124
## Lgals3    -6.615825    0    0.000    0.108
## Col9a2    -6.232358    0    0.000    0.157
## Cbr3      -6.184221    0    0.000    0.147
## Gm3764    -6.066609    0    0.001    0.283
## Slc9a3r1  -6.058875    0    0.000    0.121
## Gm26669   -6.025723    0    0.001    0.280
## Emp2      -5.954204    0    0.000    0.116
## F3        -5.940315    0    0.001    0.242
```

```
head(markers.1vs2[order(markers.1vs2$log.effect, decreasing = TRUE), ], 10)
```

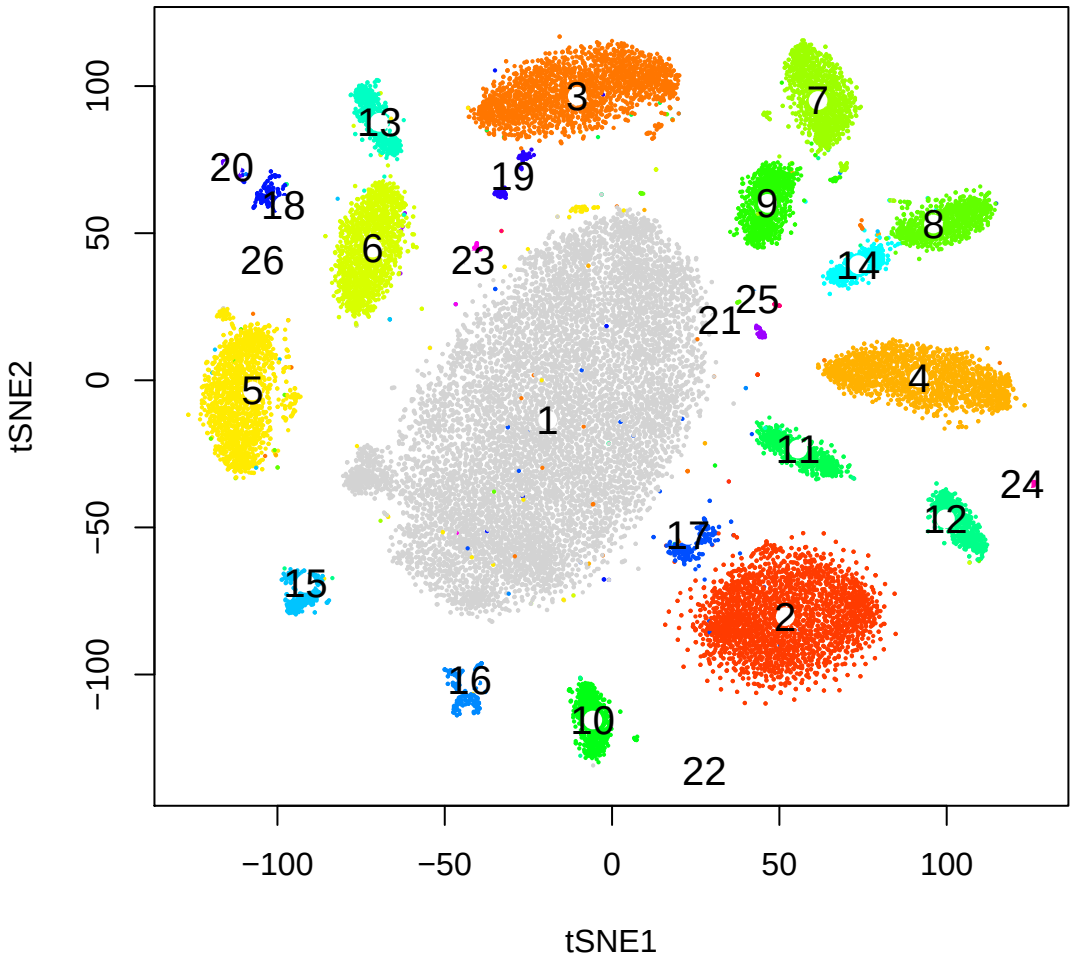
```
##               log.effect pval nTrans_1 nTrans_2
## A230077H06Rik   4.554337    0    0.198    0.002
## Slc24a4         4.547982    0    0.326    0.003
## Rapgef5         4.399130    0    0.240    0.003
## Wscd2           4.310075    0    0.213    0.002
## Kcne2           4.253009    0    0.597    0.006
## Asic4           4.194433    0    0.207    0.003
## Gzmm            4.142517    0    0.152    0.002
## Sebox           4.132954    0    0.959    0.009
## Ptprr           4.083541    0    0.453    0.005
## Lactbl1         4.045313    0    0.129    0.002
```

Indicating that cluster 1 corresponds to rod bipolar cells, while cluster 2 corresponds to Muller Glia. Next, we merge clusters that do not satisfy sufficient differential expression,

```
dsq.bip = merge.clusters.DE(dsq.bip, min.de.genes = 50, pcs.use = 1:37, TPM.mat = TPM.mat,
    Count.mat = Count.mat)
ident.louvain = dsq.bip@group
```

Next, we visualize the merged clusters on the tSNE map (same as **Figure 1C**)

```
plot.tsne(dsq.bip)
```

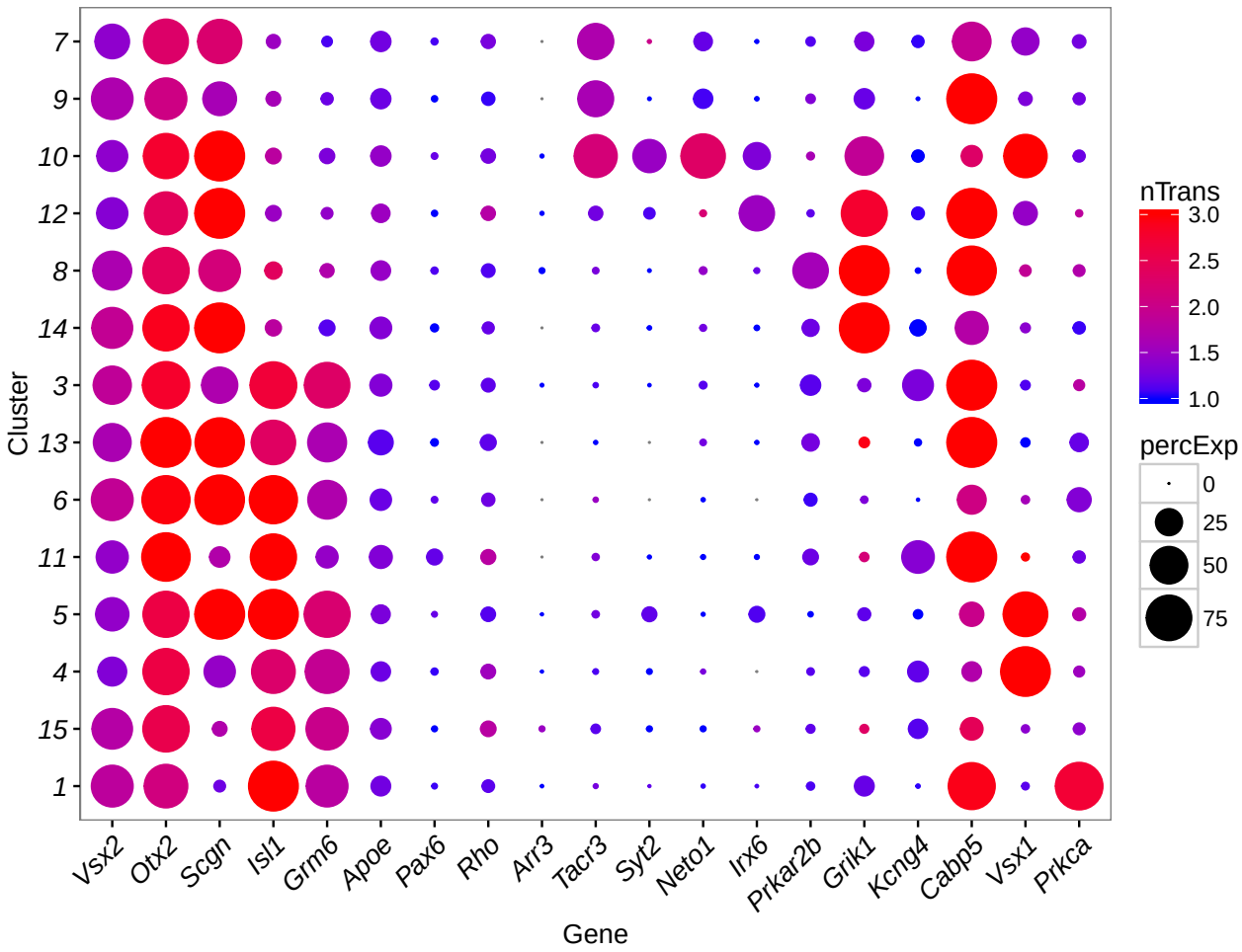


26 clusters are found.

Visualize gene expression across clusters

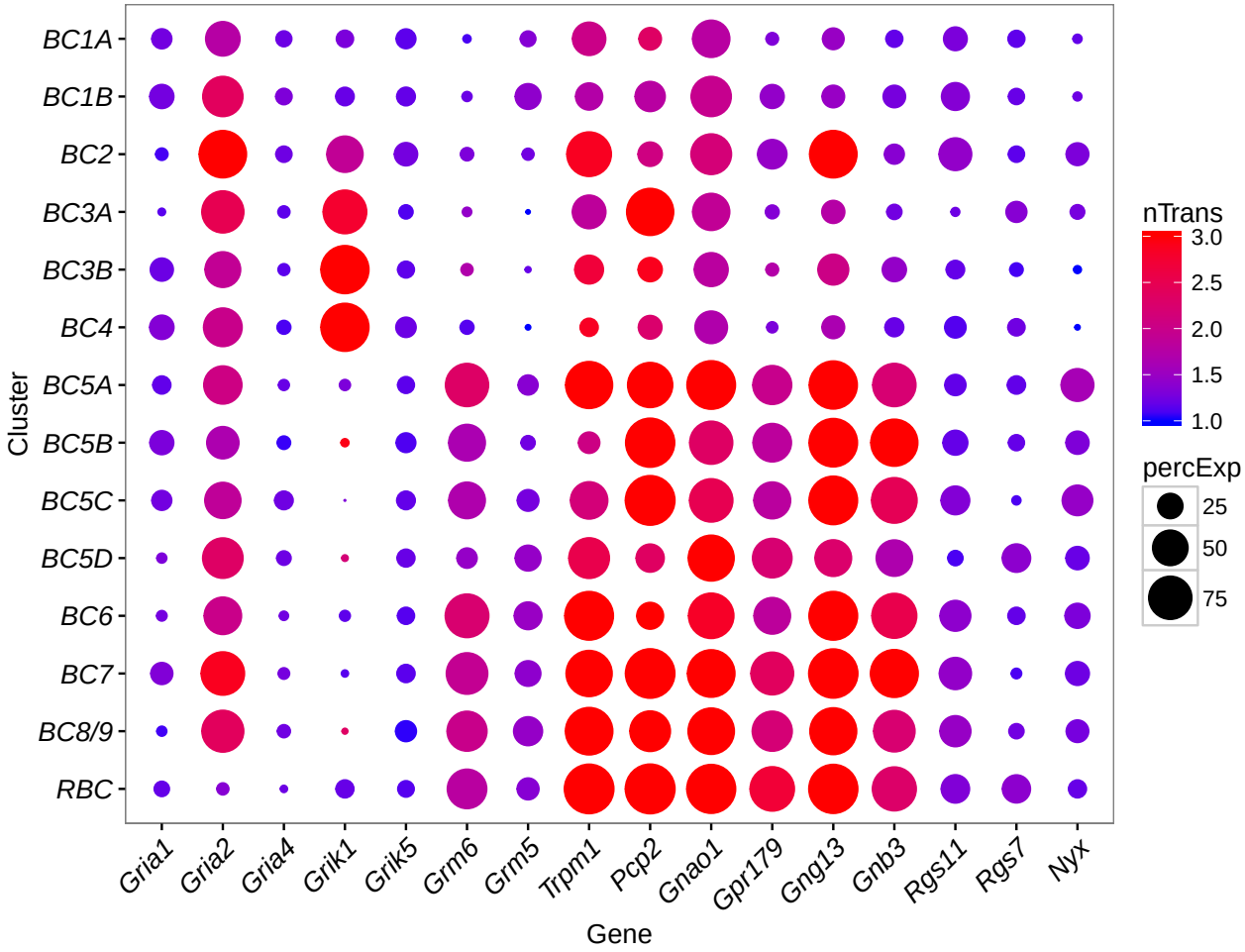
Next use the function `dot.plot` to visualize the expression of retinal markers and known bipolar markers (**Figure 1F**, upper panels),

```
dot.plot(dsq.bip, features.use = c("Vsx2", "Otx2", "Scgn", "Isl1", "Grm6", "Apoe",
    "Pax6", "Rho", "Arr3", "Tacr3", "Syt2", "Neto1", "Irx6", "Prkar2b", "Grik1",
    "Kcng4", "Cabp5", "Vsx1", "Prkca"), group.use = c(7, 9, 10, 12, 8, 14, 3, 13,
    6, 11, 5, 4, 15, 1), max.val.perc = 0.9, max.val.exp = 3, max.size = 10)
```



Note that the argument `group.use` specifies which clusters are shown. The options `max.val.perc` and `max.val.exp` limit the range of values displayed in the plot and should be set according to the limits of the dynamic range in the data. We can use `dot.plot` to visualize the glutamate receptors and components of the ON pathway (**Figure 6D**),

```
dot.plot(dsq.bip, features.use = c("Gria1", "Gria2", "Gria4", "Grik1",
    "Grik5", "Grm6", "Grm5", "Trpm1", "Pcp2", "Gnao1", "Gpr179", "Gng13",
    "Gnb3", "Rgs11", "Rgs7", "Nyx"), group.use = c(7, 9, 10, 12, 8, 14,
    3, 13, 6, 11, 5, 4, 15, 1), group.names = c("BC1A", "BC1B", "BC2",
    "BC3A", "BC3B", "BC4", "BC5A", "BC5B", "BC5C", "BC5D", "BC6", "BC7",
    "BC8/9", "RBC"), max.val.exp = 3, max.val.perc = 1, max.size = 10,
    min.perc = 0.1)
```



Note that the argument `group.names` can be used to pass user-specified names to the clusters. In our case this is based on examining expression patterns of known markers (**Table S2**).

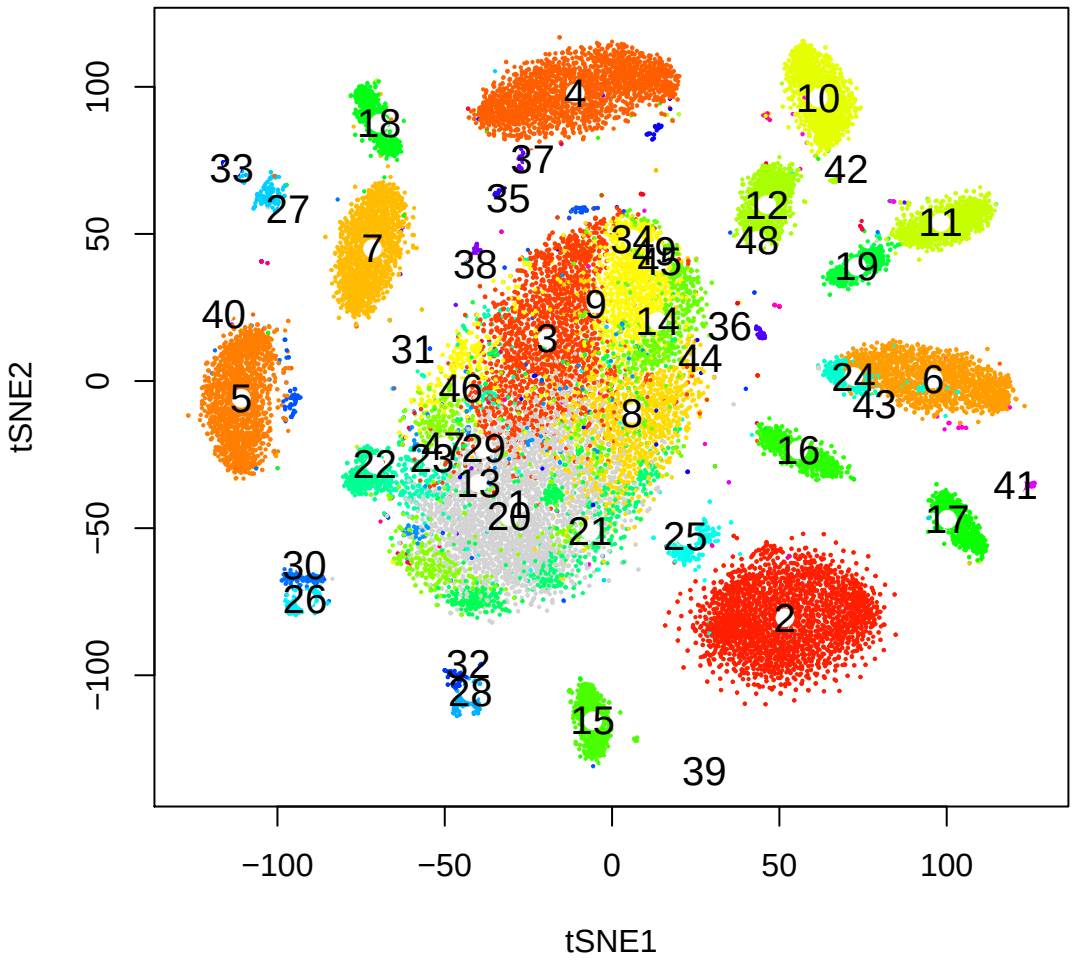
Infomap clustering

Alternatively, we can also cluster the cells using Infomap (note that the `do.jaccard` option has been set to `FALSE`),

```
dsq.bip = doGraph_clustering(dsq.bip, pcs.use = 1:37, num.nn = 30, do.jaccard = FALSE,
    method = "Infomap")
```

```
## [1] "Performing Infomap clustering. Using 30 nearest neighbors, and 37 PCs"
## [1] "Found nearest neighbors"
## [1] "Outputting clusters .."
```

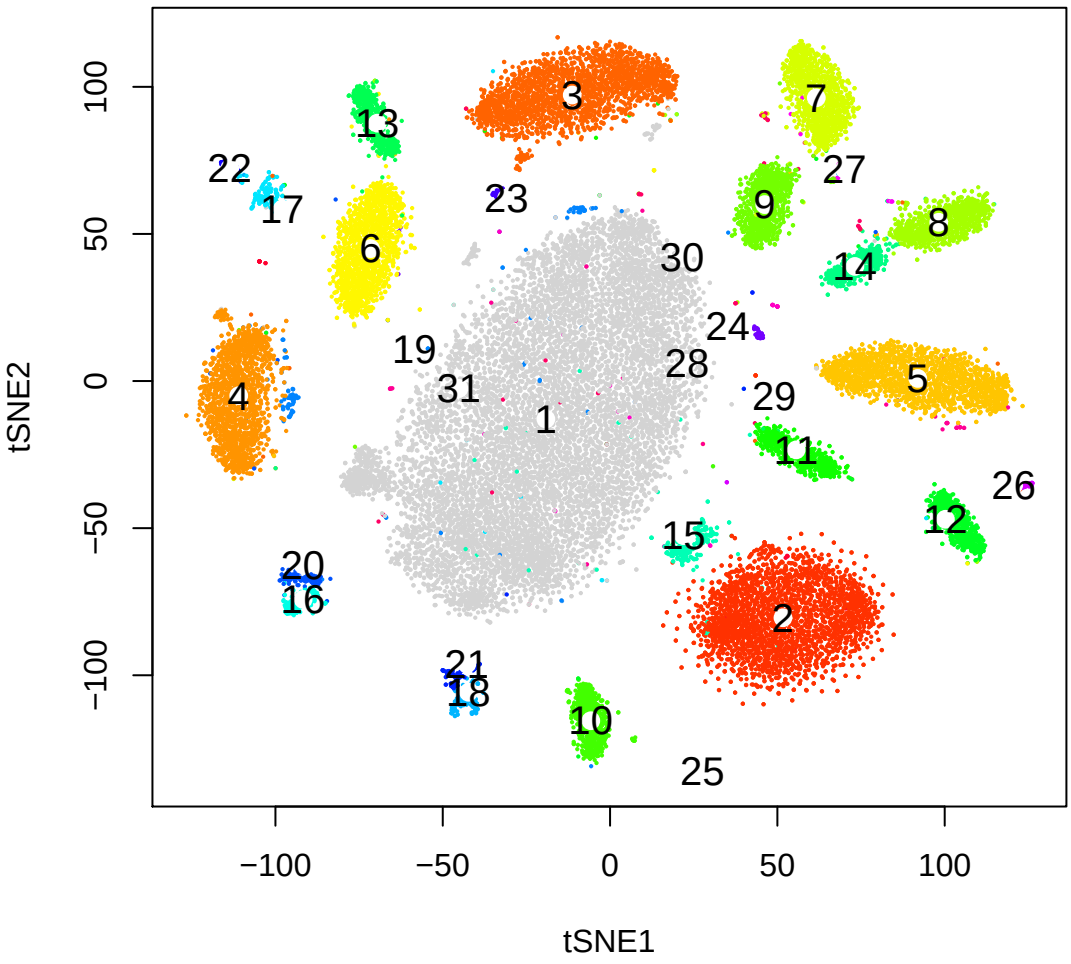
```
plot.tsne(dsq.bip)
```



This is similar **Figure S2F** in the paper (slight differences are possible because of stochasticity in the algorithm). Next we merge these clusters as before,

```
dsq.bip = merge.clusters.DE(dsq.bip, min.de.genes = 50, pcs.use = 1:37, TPM.mat = TPM.mat,
    Count.mat = Count.mat)
```

and visualize them on the tSNE map (same as **Figure 1D**),

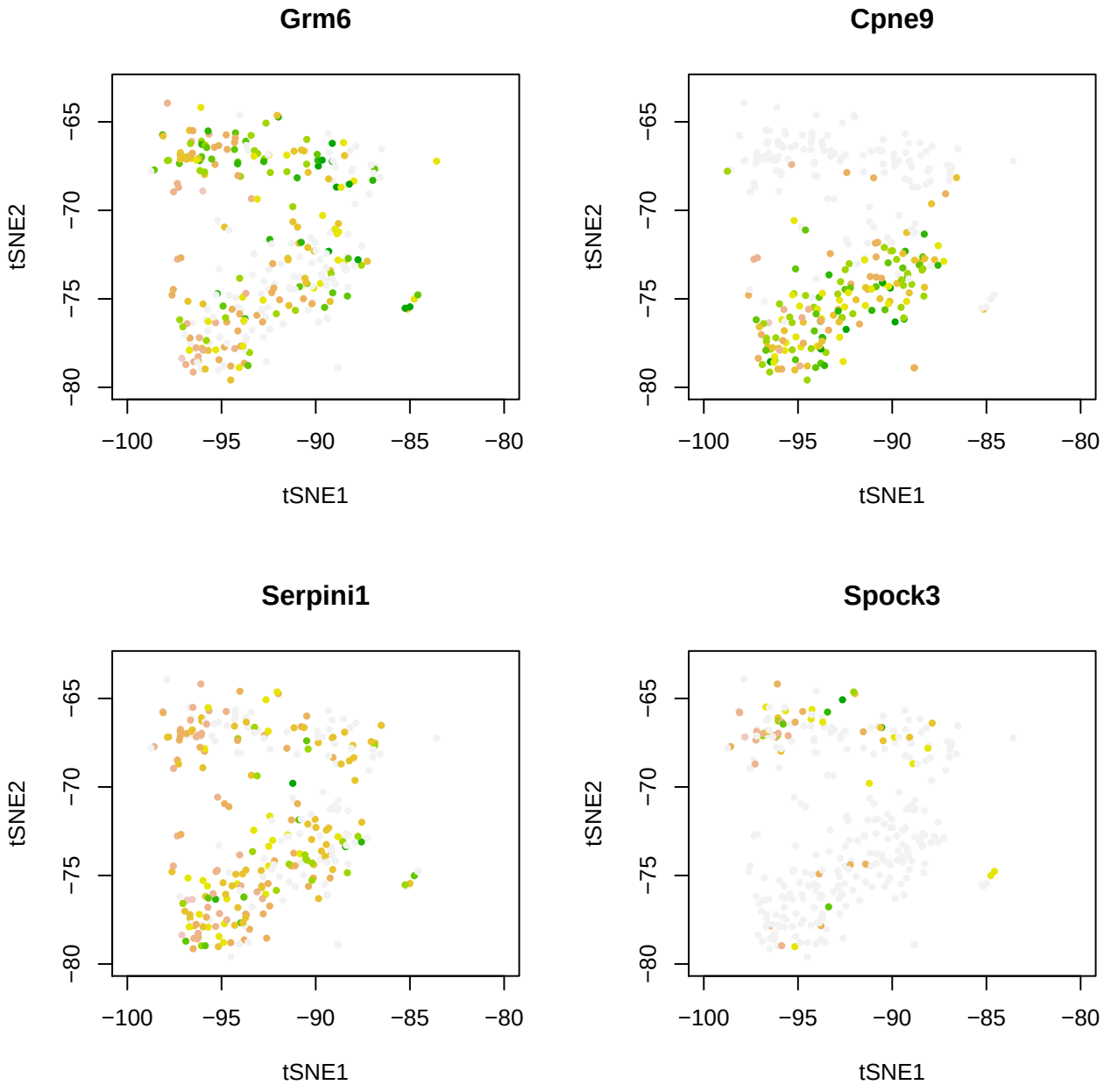



```
## Loading required package: rJava

## Loading required package: xlsxjars

## pdf
##   2

## pdf
##   2

## pdf
##   2

## pdf
##   2
```

Examination of two clusters with additional substructure found by Infomap

To examine the variation between the BC8/9 clusters identified in the paper (**Figure 5A**), we can zoom into the expression patterns of Clusters 16 and 20

```
clust.89.cells = names(dsq.bip@group[which(dsq.bip@group %in% c(16, 20))])
gene.expression.scatter(dsq.bip, c("Grm6", "Cpne9", "Serpini1", "Spock3"), cells.use = clust.89.cells,
    xlim = c(-100, -80), ylim = c(-80, -63))
```




```
markers.8vs9 = markers.binom(dsq.bip, clust.1 = 16, clust.2 = 20, effect.size = log(2),
    TPM.mat = TPM.mat, Count.mat = Count.mat)
head(markers.8vs9, 30)
```

```
##              log.effect          pval nTrans_16 nTrans_20
## Igfbp7              Inf  9.314842e-23     0.154     0.000
## Tmod1               Inf  2.065185e-15     0.122     0.000
## Bhlhe23       -3.352567  7.244434e-44     0.011     0.416
## Col1a2         3.255433  1.795847e-42     0.293     0.016
```

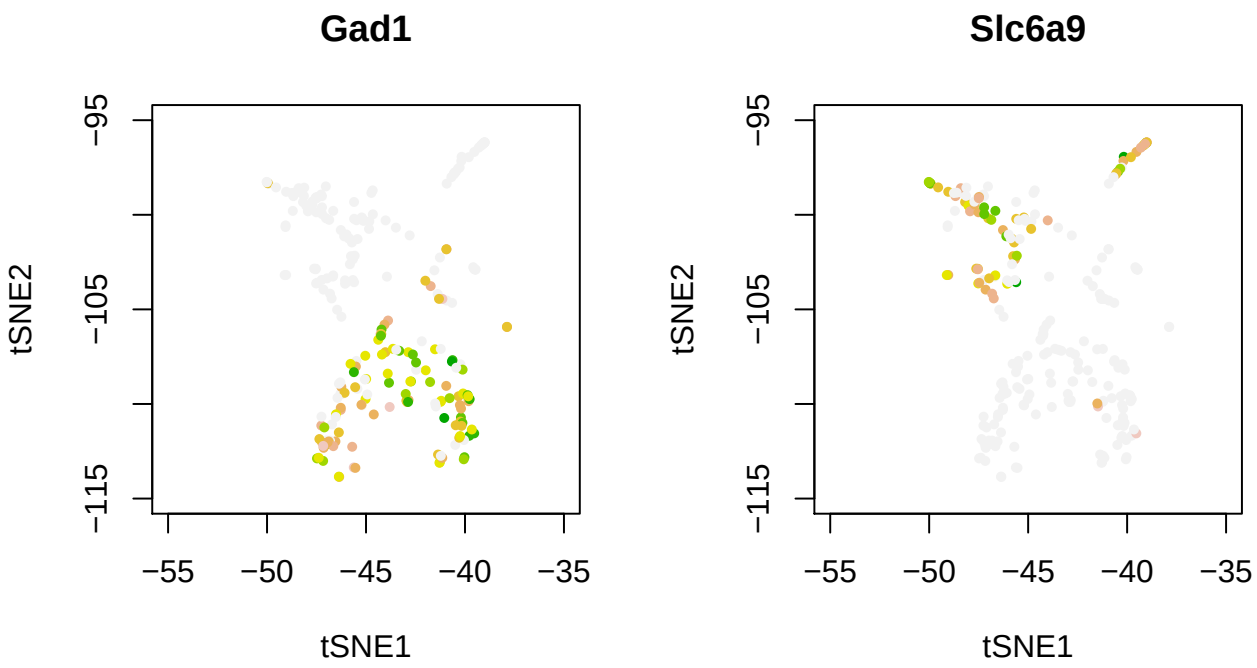
```
## Hecw1          -3.241342  5.068800e-19     0.005     0.184
## Wls            -3.047186  3.086054e-43     0.016     0.472
## Zfp804a        -3.047186  1.334254e-28     0.011     0.384
## Gabrb1         -3.047186  1.023994e-14     0.005     0.144
## Lhx3           -3.047186  1.023994e-14     0.005     0.112
## Chst9          -2.893035  5.186360e-12     0.005     0.112
## Adamtsl4        2.887709  2.268808e-25     0.186     0.008
## Dmd            -2.806023  2.303487e-20     0.021     0.232
## Tenm1           2.705387  9.399395e-39     0.356     0.016
## Ptpn13          2.682914  5.500095e-19     0.144     0.008
## Iqgap2          2.636394  6.136262e-35     0.309     0.016
## Fam19a5         2.636394  9.013167e-18     0.138     0.008
## Dusp4          -2.548194  3.851451e-14     0.011     0.184
## Cpne9           2.358191 3.654678e-142     2.553     0.152
## Six3           -2.264426  2.144349e-22     0.037     0.344
## Mdga2          -2.240710  2.602747e-17     0.027     0.264
## Dnajb2          2.230929  8.096732e-19     0.213     0.016
## Osgep          -2.199888  2.977051e-16     0.027     0.264
## Smoc1          -2.199888  2.122874e-12     0.016     0.152
## Dlg1           -2.199888  1.218294e-08     0.011     0.120
## 2610203C20Rik   2.137403  2.552622e-32     0.383     0.040
## Pcdh11x         2.137403  2.552622e-32     0.372     0.056
## Rbfox3          2.137403  2.552622e-32     0.372     0.056
## Cap2            2.076778  7.455467e-22     0.223     0.024
## Gabbr2          2.034219  7.481585e-14     0.128     0.016
## Aqp11          -2.017566  1.924092e-09     0.021     0.152
```

```
print(dim(markers.8vs9))
```

```
## [1] 325   4
```

We can also examine clusters 18 and 21, which correspond to GABAergic and Glycinergic amacrine cells (cf. **Supplementary Experimental Procedures**) based on the expression of *Gad1* and *Slc6a9*, the GABA and glycine transporters, respectively,

```
clust.amac.cells = names(dsq.bip@group[which(dsq.bip@group %in% c(18, 21))])
gene.expression.scatter(dsq.bip, c("Gad1", "Slc6a9"), cells.use = clust.amac.cells,
    xlim = c(-55, -35), ylim = c(-115, -95))
```



Acknowledgments : We gratefully acknowledge the Seurat R package developed by Dr. Rahul Satija (`http://www.satijalab.org/seurat.html`)