Project 4: Single Cell RNA-Seq Analysis of Pancreatic Cells

Salam Al-Abdullatif, Evie Wan, Mary T. Yohannes, Eetu Eklund

5/2/2020

Introduction

The pancreas is an important organ in the endocrine and digestive system of vertebrae. Its main endocrine functions include regulating blood sugar levels and secreting hormones such as insulin and glucagon. Its role in the digestive system is secreting pancreatic juice which neutralizes acid from the stomach. The dysfunction of the pancreas leads to type 1 and type 2 diabetes, pancreatitis, and cancer. Thus, it is of therapeutic and clinical importance to understand gene expression of adult beta cells.

The study by Baron et al performed droplet-based, single-cell RNA-seq on 12,000 post-mortem pancreatic cells from four human donors and four mouse strains to investigate cellular diversity in the pancreas. Unlike bulk RNA-seq, single-cell RNA-seq examines transcriptome sequence information of individual cells, providing a higher resolution of cellular differences and better understanding of individual cell functions. The study used inDrop to barcode RNA from individual cells using high-throughput droplet microfluidics. This study was able to identify previously characterized cell types, subpopulations of ductal cells with distinct expression profiles, and heterogeneity in gene regulation relating to functional maturation and ER stress (Baron et al). Moreover, the study detected disease-associated differential expression using single-cell data, providing foundation for identifying novel cell type-specific transcription factors and signaling receptors.

Our project attempts to replicate the primary findings of this study by examining human single-cell RNA-seq samples from a 51 year old female donor. We will use Seurat, which is an R package for QC and analysis of single-cell RNA-seq data to explore heterogeneity from single-cell transcriptomic measurements (Baron et al). We first created a UMI matrix with selected cells with informative barcodes using salmon alevin --- a software that uses a family of algorithms to quantify and analyze 3' tagged-end single-cell sequencing data (Stuart, Butler et al., bioRxiv, 2018). We then used Seurat to identify clusters of subpopulations, marker genes and cell types, as well as novel marker genes. Finally, we performed enrichment analysis using metascape. Top enriched genes found by metascape were inserted into PanglaoDB to search for cell types matching these marker genes.

Data and Methods

The raw scRNA-seq data from Baron et. al. data was accessed and downloaded through NCBI GEO accession number GSE84133. The full dataset includes 6 samples: four human, and two mouse. The sample selected for analysis in this report is the 51-year old female human donor, who is identified by sample number GSM2230758. The cells from this sample were sequenced with Illumina HiSeq 2500 and the inDrop platform. There are three runs associated

with this sample: SRR3879604, SRR3879605, SRR3879606. The processed barcode files were then converted into read counts by parsing through the fastq output with a bash script and counting the number of UMI reads per unique barcode. The distribution of reads across barcodes was then plotted, and barcodes corresponding to low frequency counts were then removed from downstream analysis. The mean counts for the three runs were found to be 385, 256, and 258 counts per barcode. The filtered barcodes were chosen by finding counts which are two standard deviations above the mean counts in each run.

The rationale behind this filtering process is that we want to eliminate reads with infrequent barcodes since they may be uninformative. High count barcodes may imply an abundance of duplicates created in PCR, so they are filtered in this step of data curation, while low count cells are left to be filtered in future steps. The remaining barcodes which pass the selected threshold were saved to a whitelist, which is then passed on to Salmon Alevin to generate a UMI counts matrix.

Salmon is capable of identifying transcripts without previously aligned reads with the use of Salmon's indexing function. It is recommended to build a decoy-aware transcriptome file, which we create by downloading the reference human genome and transcriptome from GENCODE, extracting the genome targets from the metadata, and concatenating the genome and transcriptome. Using the concatenated file and decoys text file, we create a Salmon indexing file for use with alevin.

Another requirement to run salmon alevin is the transcript-to-gene mapping file, which we created from the human reference annotation file from GENCODE. Finally, Salmon alevin was run on each SRR number, generating a UMI counts matrix for each run. Across the three runs, there were less than 15 barcodes with no mapped reads in each run. The salmon output logs show that as many as 40% of reads were thrown away because of noisy barcodes, so that may indicate that better filtering may have been beneficial when whitelisting the barcodes.

After the UMI count matrix was generated, Bioconductor package Seurat was used for further quality control and subpopulation identification. The UMI matrix includes 26,047 genes and 7253 cells. Salmon alevin counts were used to create the Seurat object, which contains both matrix data and analysis such as PCA and clustering. First, low-quality cells will be eliminated based on specified QC metrics. In Figure 1, the nFeature_RNA volcano plot shows the distribution of the number of genes detected in each cell. The majority of cells have between 200-4000 genes detected. Low nFeature indicates empty droplets or dead cells. Very high nFeature indicates multiplets with abnormally high gene count.

Mitochondrial genes proportion is another indicator of cell quality, because low-quality cells often have extensive mitochondrial contamination. We therefore filtered out cells with unique gene counts greater than 4000 or less than 200 and with mitochondrial genes proportion higher than 5%. Figure 2 and 3 demonstrate feature feature relationship. We can see that nFeature and nCount are highly correlated with correlation coefficient of 0.98 while percent

mitochondria is not correlated with nCount. Thus, we will include only nFeature and percent mitochondria in the filtering metrics. The number of genes and cells left after this step are respectively 24986 and 4764.

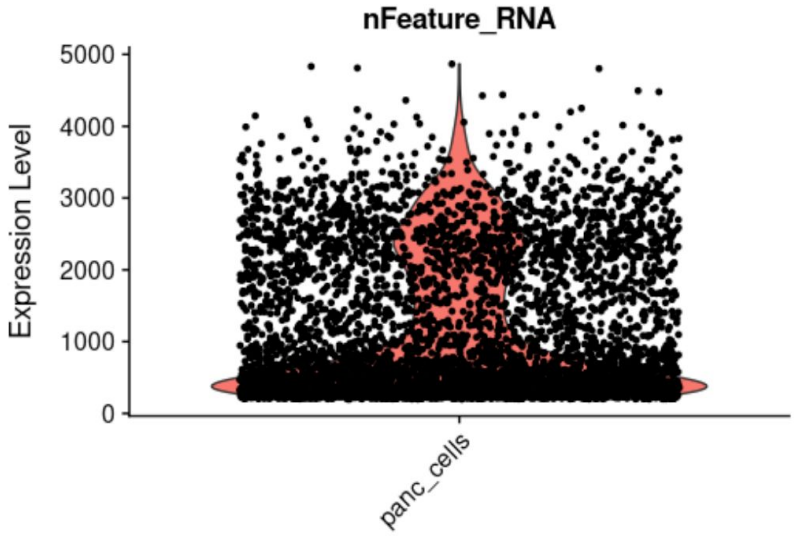

Figure 1. Distribution of cells with different number of genes

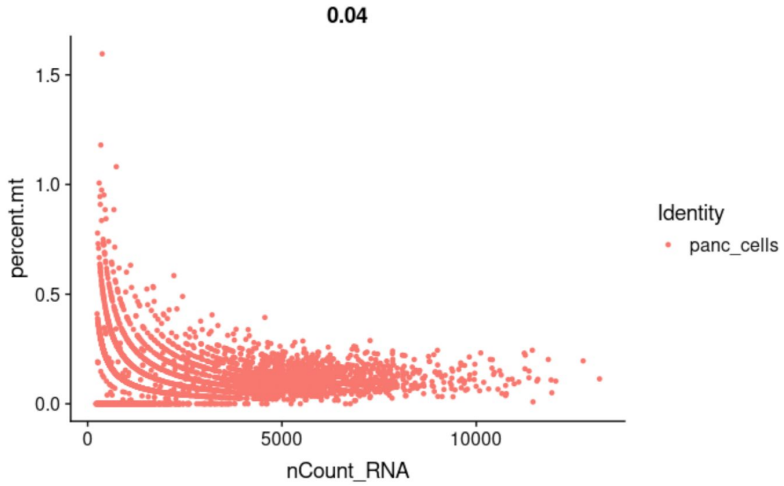

Figure 2. nCount_RNA and mitochondria percentage are not correlated

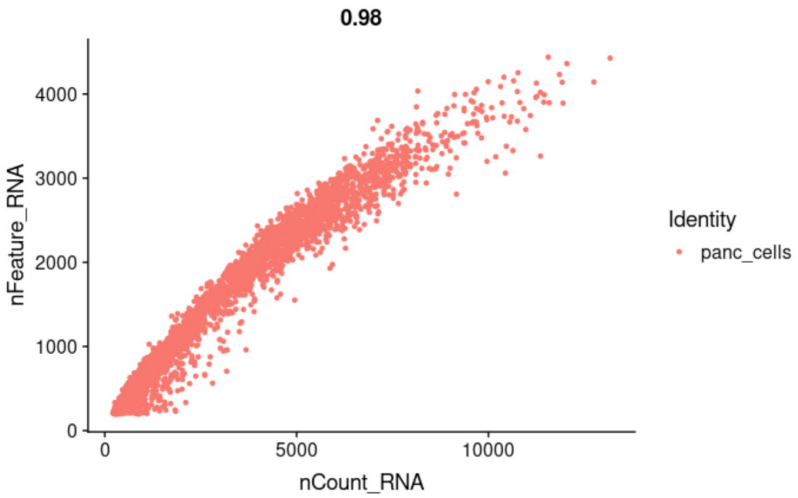

Figure 3. nCount_RNA and nFeature_RNA are highly correlated

After low-quality cells have been removed, data was normalized using the global-scaling normalization method “LogNormalize” to normalize gene expression measurements for each cell by the total expression and then multiplied by a scale factor(10000). After normalization, genes that have high cell-to-cell variation were then selected. In other words, these genes are highly expressed in some cells and lowly expressed in others. The top 10% most variable genes were selected. Figure 4 shows the 2499 most variable genes identified.

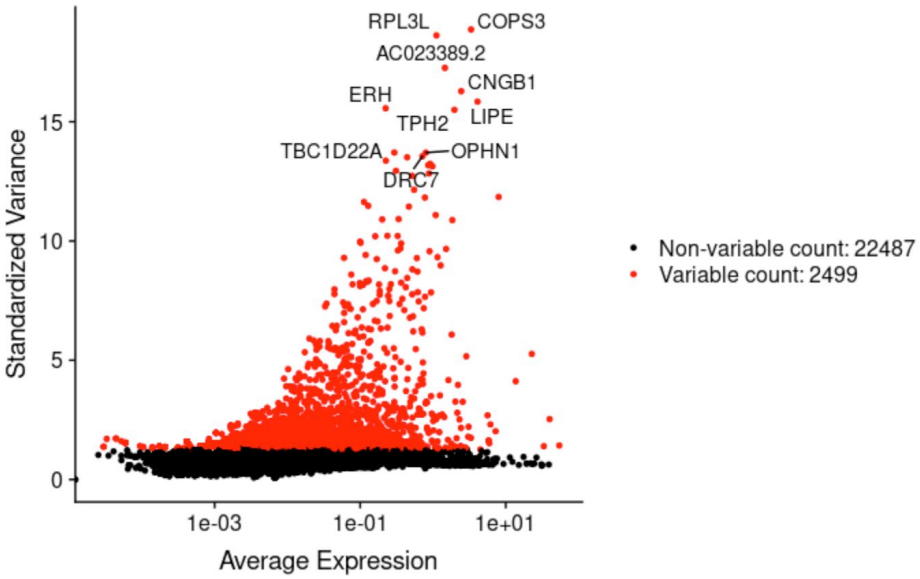

Figure 4. Genes that exhibit high cell-to-cell variation shown in red

Next, linear transformation scaling was applied to prepare the data for dimensional reduction. The expression of each gene was scaled to make mean expression and variance across cell zero, giving equal weight to each gene for downstream analyses. PCA was then performed on the scaled data. Elbow plot was used to visualize PCs based on the percentage of variance explained by each PC. The plot shows an “elbow effect” around PC10-11, indicating that the majority of true signal is captured in the first 10 PCs. We chose the first 10 PCs for downstream analyses. Finally, we applied Seurat’s graph-based clustering approach to identify cell clusters. FindNeighbors function was used to construct the KNN graph. Then, FindClusters function apply modularity optimization techniques to iteratively group cells together. Resolution was set to 0.85, which is roughly the middle of the recommended range. Cells were grouped in 15 clusters using such methods. The number of genes and cells selected are respectively 24,986 and 4770.

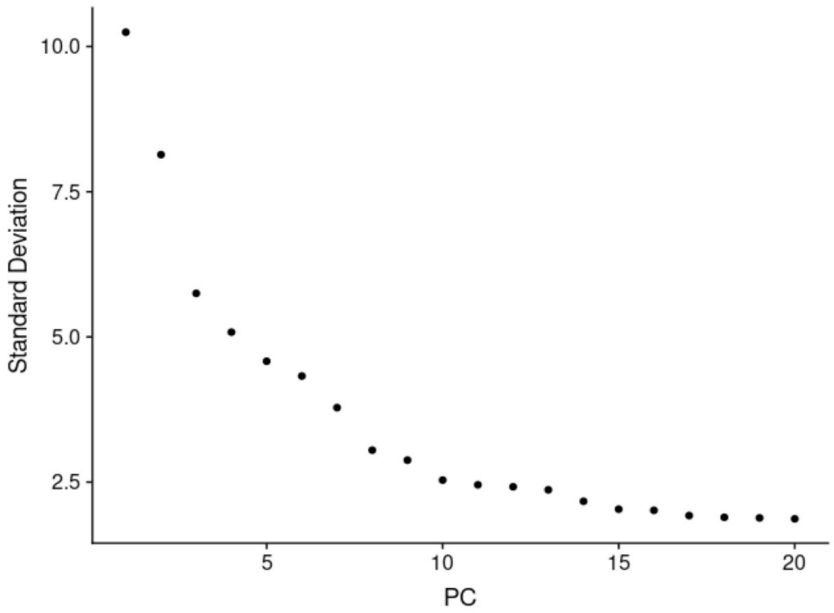

Figure 5. Elbow plot shows that “elbow effect” occurs near PC10

The RDS file containing a saved Seurat object with processed and clustered counts was loaded into R using the “readRDS” function and parts of the Seurat - Guided Clustering Tutorial from Satija lab (Stuart et al) were used to guide the following analyses. Marker genes for each cluster were identified using FindAllMarkers, a differential expression function from the Seurat package. The analysis only took positive values into consideration with a minimum pct value of 0.25 and a logfc threshold of 0.25. Then, known genes, which were used in the Baron et al paper to find cell types, were checked for being cluster-specific in our dataset by searching for them in our result dataset. For the known marker genes that were not cluster-specific, a heatmap of their expression values was used to determine which cluster they were differentially expressed in and to what extent. This helped to classify the known marker genes to their

respective clusters so that each cluster could be categorized into its respective cell type; cell type classification of clusters was based on their marker genes.

There were some clusters that did not have not have any of the known marker genes indicated in the Baron et al paper. Thus, other differentially expressed marker genes within those clusters with the highest positive ave_logFC were searched in the PanglaoDB database (Franzén et al 2019) to determine their cell types. A list of the cluster-specific differentially expressed genes was also searched in the database to validate the cell type choices of each cluster. A violin plot using the VlnPlot function was executed on features to explore the presence of duplet clusters in our dataset.

Once the cell type of each cluster was determined using known marker genes from the paper and PanglaoDB, it was added as a column into our dataset. Since there were problems with installing the UMAP package, t-SNE was used to generate a projection and scatter plot of the clustered cells and the clusters were labeled with their respective cell types that were previously assigned to them. This figure was then compared to Figure 1D in the Baron et al paper. The top two marker genes in each cluster were then visualized using the "DoHeatmap" function to see differences. The clustered heatmap of log normalized UMI counts for those marker genes across all cells was then compared to Figure 1B of the Baron et al paper. Furthermore, other differentially expressed genes in the clusters that were just as discriminative of cell type as the marker genes (novel marker genes) were identified by pre-filtering the features that are detected at <50% frequency in either of the clusters and had an adjusted p-value of <0.05. From that result, previously identified known marker genes were dropped and the final output of each cluster was combined into one big dataset for further analysis.

A filter of p-value adjusted < $5*10^{-6}$ was used on the marker genes for our gene set enrichment analysis. This adjusted our genes from 181853 marker genes to 137894 marker genes. Prior to adjustment, clusters 0, 1, and 3 had 3, 5, and 0 genes respectively. After adjustment, clusters left were 2, 4-14. Each of these clusters were run through Metascape to perform enrichment analysis. Not all clusters received enrichment scores. Clusters 2, 6, 8-14 got enrichment results. The top enriched genes for these clusters were input into pangloaDB to find the top matched cell types. Additionally, genes within the sampled tissue of pancreatic islets were searched for to find the cell types corresponding to the top ranked genes (most highly expressed marker genes) within the pancreas.

Results and Discussion

After the processing of the UMI counts matrix, there were 4,770 cells in total in the RDS file and 15 clusters were identified. Each cluster had a different number of cells: 813, 780, 556, 526, 354, 311, 273, 265, 201, 194, 160, 143, 98, 61, and 50. There were 14 known marker genes in the Baron et al paper namely GCG, INS, SST, PPY, GHRL, CPA1, KRT19, RGS5, PDGFRA, VWF, SDS, TPSAB1, TRAC, and SOX10. Genes INS and PYY were filtered out prior to Seurat analysis and thus were not considered in the succeeding analyses. However, INS-IGF2, locus that includes two alternatively spliced read-through transcript variants which align to the INS gene in the 5' region and to the IGF2 gene in the 3' region, was present in our RDS dataset. Therefore, we decided to use INS-IGF2 in place of INS and removed PYY (marker gene for Gamma cells) from our known-marker-genes list.

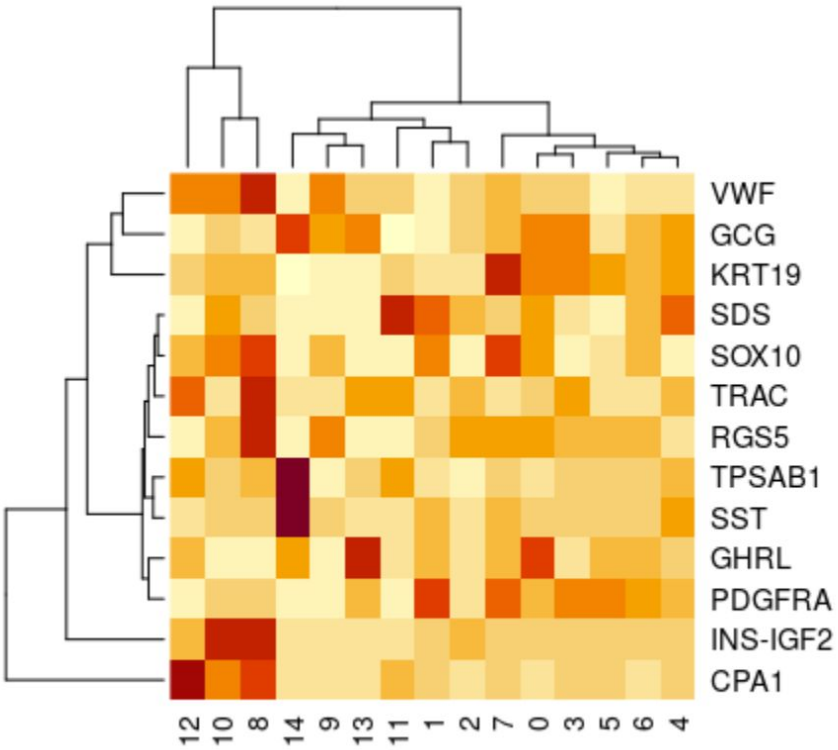

Figure 6. Heatmap of the expression values of known marker genes that were not cluster-specific in our analysis; used to classify marker genes to their respective clusters so that each cluster could be categorized into its respective cell type

Table 1. 15 of our clusters and their respective cell types determined by the Baron et al paper, heatmap visualization and PanglaoDB database

Cluster	Cell_types
0	Germ cells
1	Germ cells
2	Epsilon
3	Germ cells
4	T-cells
5	Activated Stellate
6	Ductal, Schwann
7	Alpha cells
8	Germ cells
9	Hepatocytes
10	Macrophages
11	Acinar, Cytotoxic T, Endothelial, Quiescent Stellate
12	Beta cells
13	Beta cells
14	Delta cells, Mast

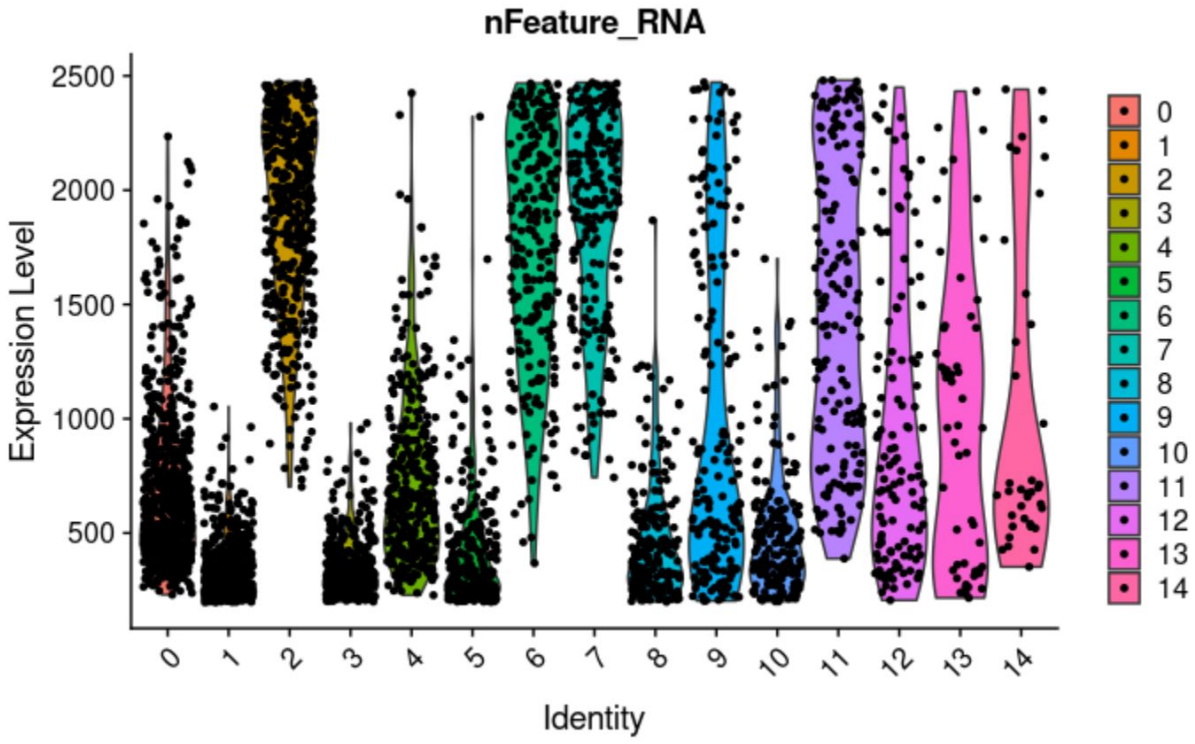

Figure 7. Violin plot of features used to explore the presence of duplet clusters in our dataset; produced using the VlnPlot function

Among the marker genes identified in the Baron et al paper, INS-IGF2 and CPA1 were the only marker genes that were cluster-specific in our analysis and identifying which major pancreatic cell types they associate with was not challenging to find. Marker genes of 6 clusters (2, 5, 6, 7, 10, 11, 12, 14) were identified using their expressions displayed on the heatmap (Figure 6). 7 clusters (0, 1, 3, 4, 8, 9, 13) did not have any of the known marker genes indicated in the Baron et al paper and PanglaoDB was used to classify and label with their respective cell types (Table 1). Clusters 2, 6, 8 and 11 had more than one marker genes and thus identified with multiple cell types. This might have been due to the presence of cell mixtures within those clusters - duplet clusters; this can be seen in the violin plot (Figure 7) to an extent.

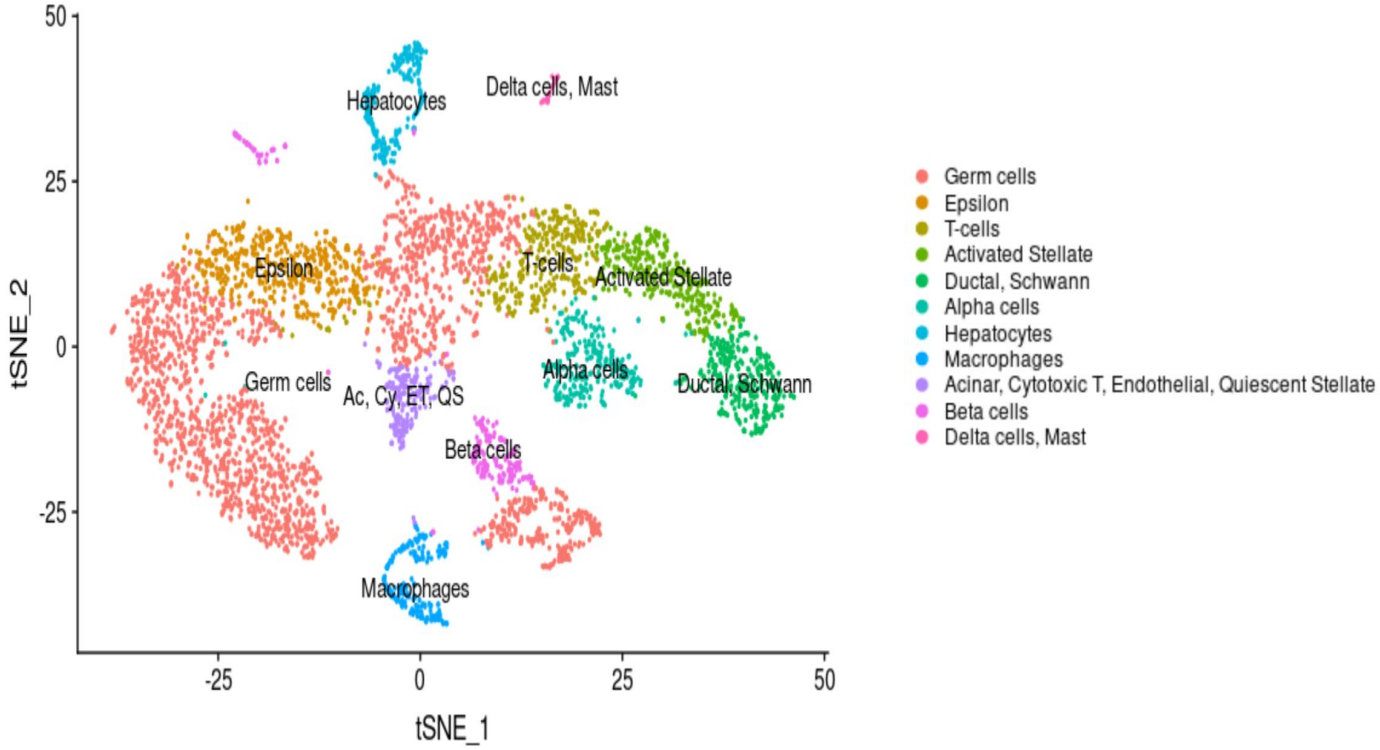

Figure 8. t-SNE plot of the clustered cells colored by clusters and labeled with respective cell types

As compared to Figure 1D in the Baron et al paper, our t-SNE plot (Figure 8) indicates we are able to successfully identify cell subpopulations within the dataset. Similar to the published results, we are able to identify beta cells, delta cells, alpha cells, and ductal cells. However, our projection shows a smaller subpopulation of beta cells and delta cells, with a higher representation of germ cells. One potential reasoning for these results is that the whitelisted cell barcodes did not filter enough of the uninformative count reads, leaving a high proportion of germ cells in the dataset.

Figure 9. Clustered heatmap of log normalized UMI counts for the top two marker genes of each cluster across all cell types; visualized using “DoHeatmap” function

We visualized expression of the top differentially expressed genes in Figure 9. Compared to the heatmap published in Baron et al Figure 1B, we see a clear expression pattern in differentially expressed genes. We selected the two top features per cluster for visualization, and we can see that most of our clusters show specific expression of the top differentially expressed genes, but we can also see some shared expression patterns among some clusters such as cluster 0 and cluster 2. The clear distinction between expression levels across clusters suggests that we were successful in identifying cell subpopulations. The shared expression patterns between clusters suggests that we may have produced more clusters than distinct cell subtypes; however, delineating cell clusters with more precision requires a more in-depth understanding of the cell types represented in the cell population. To more closely match the results from Baron et al, we would need to devise a method for accurately selecting unique markers, but the results of the heatmap in Figure 9 suggest we have sufficiently distinct clusters and biomarkers.

Additionally, across all clusters, 2132 genes were identified as being novel marker genes; 14 in cluster 0, 2 in cluster 1, 36 in cluster 2, 35 in cluster 4, 11 in cluster 5, 113 in cluster 6, 62 in cluster 7, 24 in cluster 8, 392 in cluster 9, 4 in cluster 10, 605 in cluster 11, 255 in cluster 12, 275 in cluster 13, and 304 in cluster 14. There were no novel marker genes identified in Cluster 3.

Table 2. Matched cell type for each cluster with enriched genes found by Metascape. Top 5 enriched genes were used to search for matching cell types. Last column includes the inferred cell type of the most expressed gene within Pancreatic Islets.

Cluster	Top 5 Enriched Genes	Process	Top Matched Cell Type	Top Cell Type with Top Ranked Gene in Pancreatic Tissue
2	MAPT,KIF3A,KIF1B	axonal transport	Neurons	Beta cells
6	C1QC,GSN,TLL1,ATG12, WDR1	cellular component disassembly	Macrophages	Beta cells
8	ID4,INSM1,TEAD3,CDON	neural precursor cell proliferation	Astrocytes	Pancreatic stellate cells
9	CSNK2A1,IGFBP1,PPM1B, PSMB10,PSMD10	regulation of Wnt signaling pathway	Macrophages	Beta Cells
10	SMARCA5,KAT6B,CDCA5	DNA packaging	Germ Cells	Beta Cells
11	SLC11A2,ATP2C2,SLC39A 14,ATP2C1,SLC6A12	manganese ion transmembrane transport	Hepatocytes	Pancreatic Stellate Cells
12	GLB1,PFKL,PGAM2,RPE,I DUA,B4GALT3	monosaccharide catabolic process	Germ Cells	Beta Cells
13	DHX15,HSP90AB1,HTR2A ,OPRM1,NPFF	response to alkaloid	Neurons	Beta Cells
14	IL4,MECP2,NDUFB7,SDH D,VCP	cellular respiration	Macrophages	Alpha Cells

Alpha cells, beta cells, and macrophages match the cell types found in the Baron et al. study (Table 2). More cell types matching the study were also found when looking through the cell types associated with pancreatic islets. These were not included since they were not the highest ranked (most highly expressed marker gene) matching a pancreatic islet. These cell types include Acinar cells, Delta Cells, Endothelial Cells, Gamma cells, and Ductal cells.

A too stringent adjusted p-value cutoff ($< 5*10^{-6}$) may have been used in filtering, which would have removed some clusters entirely and resulted in no enrichment scores within clusters with too few cells. A much less stringent p-value cutoff (0.05) was also used, but this resulted in a loss of some clusters and no enrichment scores for a few more. Without any filtering in place, cluster 0 included only 2 genes, cluster 1 included 4, and cluster 3 included 0 genes (.csv was slightly changed after doing analysis, 3 clusters still only had <6 after update). These clusters do not give enrichment any results due to too few genes included. Since a loss of some clusters and some enrichment scores happened with any filtering using the p-value, or no filtering at all, it seemed to be expected. So a low adjusted p-value cutoff ($< 5*10^{-6}$) was used to be more confident in the fold changes of expression in our marker genes, which might differentiate some clusters better (this may have not been the right way to go, causing us to lose too much data). It is possible some marker genes were lost to this, causing our results to not differentiate as well as it should. This could be why Beta cells are such a prevalent cell type in our results. Beta cells

make up most of pancreatic islets, so a loss of some marker genes could make more of our results match Beta cells rather than other cell types not as prevalent in the pancreas.

Conclusion

The results from our report show that we have the ability to analyze single cell RNA sequencing data and analytically find cell types and subpopulations of interest. Comparing our results to the Baron et al. study, we find a few common results. Beta cells, Alpha cells, and Macrophages match the cell types in the study. Without restricting the data to only the top ranked gene expressed, we also find Acinar cells, Delta cells, Endothelial cells, Gamma cells, and Ductal cells. Beta cell is the most common cell type found in our study, which is also the most common cell type in the pancreas. This could be due to different filtering thresholds accidentally removing marker genes from our results, or some clusters not differentially expressing any marker genes, which could cause a lack of clear boundaries between cell clusters and subtypes. The whitelisting process, where the barcodes with infrequent reads were removed, could also affect the accuracy of our analysis. The presence of too many noisy barcodes could have affected Salmon alevin's ability to properly map transcripts, which could therefore influence the later steps of the project. Overall, this is a demonstration of the power of single cell RNA-seq and its ability to discover novel cell subtypes.

References

1. Baron M, Veres A, Wolock SL, et al. A Single-Cell Transcriptomic Map of the Human and Mouse Pancreas Reveals Inter- and Intra-cell Population Structure. *Cell Syst*. 2016;3(4):346-360.e4. doi:10.1016/j.cels.2016.08.011

2. Oscar Franzén, Li-Ming Gan, Johan L M Björkegren, PanglaoDB: a web server for exploration of mouse and human single-cell RNA sequencing data, *Database*, Volume 2019, 2019, baz046, https://doi.org/10.1093/database/baz046

3. Tim Stuart, Andrew Butler, Paul Hoffman, Christoph Hafemeister, Efthymia Papalexi, William M. Mauck III, Marlon Stoeckius, Peter Smibert, Rahul Satija bioRxiv 460147; doi: https://doi.org/10.1101/460147

R Markdown Code

Programmer Code

```
txi_04 <- tximport(files04, type="alevin")$counts

## reading in alevin gene-level counts across cells

txi_05 <- tximport(files05, type="alevin")$counts

## reading in alevin gene-level counts across cells

txi_06 <- tximport(files06, type="alevin")$counts

## reading in alevin gene-level counts across cells

txi_05 <- txi_05[rownames(txi_04), ]

txi_06 <- txi_06[rownames(txi_05), ]

txi_all <- cbind(txi_04, txi_05, txi_06)

#convert id to gene name

#matrix04

panc_cells <- CreateSeuratObject(counts = txi_all, project = "panc_cells", min.cells
= 3, min.features = 20)

## Warning: Non-unique cell names (colnames) present in the input matrix,

## making unique

txi04_ct <- as.data.frame(panc_cells@assays$RNA@data@Dimnames[[1]])

eid04 <- (txi04_ct[,1])

write.table(eid04, "eid04.txt", row.names = FALSE, quote = FALSE)

eid04_file <- read.delim("/projectnb2/bf528/users/group_5/project_4/evie/mart_export
(8).txt")

genes04 <- eid04_file$Gene.name

genes04 <- as.character(genes04)

panc_cells@assays$RNA@data@Dimnames[[1]] <- genes04

panc_cells@assays$RNA@counts@Dimnames[[1]] <- genes04

panc_cells[["percent.mt"]] <- PercentageFeatureSet(panc_cells, pattern = "^MT-")
```

```
#before_filter <- subset(genes_paper, !genes_paper %in%
panc_cells@assays$RNA@counts@Dimnames[[1]])

#INS" "PPY" not in list

panc_cells <- subset(panc_cells, subset = nFeature_RNA > 200 & nFeature_RNA < 4000 &
percent.mt < 5)

VlnPlot(panc_cells, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol =
2)
```

```
#visualize feature-feature relationships

plot1 <- FeatureScatter(panc_cells, feature1 = "nCount_RNA", feature2 = "percent.mt")

plot2 <- FeatureScatter(panc_cells, feature1 = "nCount_RNA", feature2 =
"nFeature_RNA")

plot1
```

```
plot2
```

normalization

```
panc_cells <- NormalizeData(panc_cells, normalization.method = "LogNormalize",
scale.factor = 10000)
```

features that exhibit high cell-to-cell variation

```
panc_cells[["RNA"]]@meta.features <- data.frame(row.names =
rownames(panc_cells[["RNA"]]))

ntf_04 = round(0.1*nrow(panc_cells[["RNA"]]), digits = 0)

panc_cells <- FindVariableFeatures(panc_cells, selection.method = "vst", nfeatures =
ntf_04)

top10 <- head(VariableFeatures(panc_cells), 10)

plot3 <- VariableFeaturePlot(panc_cells)

plot4 <- LabelPoints(plot = plot3, points = top10, repel = TRUE, xnudge = 0, ynudge =
0)

## Warning: Using `as.character()` on a quosure is deprecated as of rlang 0.3.0.

## Please use `as_label()` or `as_name()` instead.

## This warning is displayed once per session.

plot4
```

```
## Warning: Transformation introduced infinite values in continuous x-axis
```

Scale Data

```
all.genes.04 <- rownames(panc_cells)

panc_cells <- ScaleData(panc_cells, features = all.genes.04)

## Centering and scaling data matrix
```

identify clusters

with each PC essentially representing a 'metafeature' that combines information across a correlated feature set.

```
panc_cells <- RunPCA(panc_cells, features = VariableFeatures(object = panc_cells))

#print(panc_cells[["pca"]], dims = 1:10, nfeatures = 10)

#VizDimLoadings(panc_cells, dims = 1:2, reduction = "pca")

#DimPlot(panc_cells, reduction = "pca")

DimHeatmap(panc_cells, dims = 1:5, cells = 500, balanced = TRUE)
```

```
ElbowPlot(panc_cells)
```

```
panc_cells <- FindNeighbors(panc_cells, dims = 1:10)

## Computing nearest neighbor graph

## Computing SNN

panc_cells <- FindClusters(panc_cells, resolution = 0.75)

## Modularity Optimizer version 1.3.0 by Ludo Waltman and Nees Jan van Eck

## Number of nodes: 4764

## Number of edges: 164611

## Running Louvain algorithm...

## Maximum modularity in 10 random starts: 0.8784

## Number of communities: 15

## Elapsed time: 0 seconds

file04_clusters <- Idents(panc_cells)

head(Idents(panc_cells), 5)
```

```
## GACCTATTCAATGCGACTA AAATCAGAACTAGTGCTTC ATAGTGGACACGAGTGTAC
##                   6                   0                   6
## GATGGAGGAGAGTGTCAGG GAGCGGAAACTCCGATACG
##                   6                   6
## Levels: 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14

saveRDS(panc_cells, file = "programmer_output.rda")

clusters <- as.data.frame(Idents(panc_cells))

clusters2 <- clusters %>% rownames_to_column("cells") %>%
group_by(Idents(panc_cells)) %>% summarise(n())

colnames(clusters2) <- c("Clusters", "Number of Cells")

genes_paper <- c("GCG", "INS", "PPY", "SST", "GHRL", "PRSS1", "KRT19", "SPARC",
"VWF", "RGS5", "PDGFRA", "SOX10", "SDS", "TPSAB1", "TRAC")

filtered_out <- subset(genes_paper, !(genes_paper %in%
panc_cells@assays$RNA@counts@Dimnames[[1]]))

#gene in var feature
filtered_out_var <- subset(genes_paper, genes_paper %in%
panc_cells@assays$RNA@var.features)
```

Analyst Code

```
# loading RDS file containing saved Seurat object with processed, clustered counts
ds <- readRDS("/projectnb/bf528/users/group_5/project_4/evie/programmer_output.rda")

# how many cells in each clusters (15 clusters)
#table(ds@active.ident)

# markers in the paper
paper_markers<- c("GCG", "INS", "SST", "PPY", "GHRL", "CPA1", "KRT19", "RGS5",
"PDGFRA", 'VWF', 'SDS', 'TPSAB1', 'TRAC', 'SOX10')

# are the marker genes indicated in the paper found in our seurat object?
#paper_markers %in% rownames(ds)
###INS and PYY were not found in the gene list, but we found INS-IGF2 so substittuted
that for INS and removed PYY from out markers list (so no Gamma cells in our dataset)

updated_markers <- c("GCG", "INS-IGF2", "SST", "GHRL", "CPA1", "KRT19", "RGS5",
"PDGFRA", 'VWF', 'SDS', 'TPSAB1', 'TRAC', 'SOX10')

##############################################
#1) identify marker genes for each cluster
##############################################
```

```
# finding differentially expressed features (cluster biomarkers) in all clusters
(general analysis)
all_markers <- FindAllMarkers(ds, only.pos = TRUE, min.pct = 0.25, logfc.threshold =
0.25)

# checking if our gene list were cluster-specific
updated_markers %in% rownames(all_markers)
# INS-IGF2 and CPA1 were the only ones that were both identified to a certain cell
type in the paper and were cell specific markers/specific to one cluster in our
dataset
# According to the paper = INS - INS-IGF2 = beta and CPA1 = Acinar

# Find cell types for clusters based on the marker genes found in each
# find out marker genes for the rest of the clusters
# heatmap of expressison values of gene markers (given in the paper) in each cluster
to see which ones are differentially expressed
ds <- SetIdent(ds, value = "seurat_clusters")
heatdata <- AverageExpression(ds, features = updated_markers)$RNA
heatmap(as.matrix(heatdata))
# KRT19 and SOX10 = 6, SDS = 10, SST and TBSAB1 = 14, INS-IGF2 = 12, CPA1 and TRAC
and VWF and RGS5 = 11, GCG = 7, PDGFRA = 5, GHRL = 2 (might be a mixture of different
cells)

# could not find celltypes for clusters 0, 1, 3, 4, 8, 9, 13. so searched the other
differentially expressed marker genes on (https://panglaodb.se/) then got the cell
type from there; searched all of the marker genes differentially expressed in each
cluster together and also the one with the highest ave_logFC by itself
#rownames(head(FindMarkers(ds,ident.1 = 0, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)))
#rownames(head(FindMarkers(ds,ident.1 = 1, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)))
#rownames(head(FindMarkers(ds,ident.1 = 3, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)))
#rownames(head(FindMarkers(ds,ident.1 = 4, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)))
#rownames(head(FindMarkers(ds,ident.1 = 8, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)))
#rownames(head(FindMarkers(ds,ident.1 = 9, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)))
#rownames(head(FindMarkers(ds,ident.1 = 13, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)))
# 0 = Germ cell, 1 = Germ cells, 3 = Germ cells, 4 = T-cell, 8 = Germ cell, 9 =
Hepatocytes, 13 = Beta cell

# more info - top 10 marker genes for each cluster
#top1markers=all_markers %>% group_by(cluster) %>% top_n(n = 1, wt = avg_logFC)

# explaining clusters 2, 6, 8, 11
#head(ds@meta.data)
VlnPlot(ds, "nFeature_RNA")

#########################################################
#2) Label clusters with cell type based on marker genes
#########################################################
```

```
# used the paper and PanglaoDB
celltypes <- c('Germ cells', 'Germ cells', 'Epsilon', 'Germ cells', 'T-cells',
'Activated Stellate', 'Ductal, Schwann', 'Alpha cells', 'Germ cells', 'Hepatocytes',
'Macrophages', 'Acinar, Cytotoxic T, Endothelial, Quiescent Stellate', 'Beta cells',
'Beta cells', 'Delta cells, Mast')
all_markers$cell_types <- celltypes[all_markers$cluster] # add as a column

#######################################
#3) visualize clustered cells using tSNE
#######################################

lab <- c('Germ cells', 'Germ cells', 'Epsilon', 'Germ cells', 'T-cells', 'Activated
Stellate', 'Ductal, Schwann', 'Alpha cells', 'Germ cells', 'Hepatocytes',
'Macrophages', 'Ac, Cy, ET, QS', 'Beta cells', 'Beta cells', 'Delta cells, Mast')
ds_tsne <- RunTSNE(ds, dims = 1:10)
names(lab) <- levels(ds_tsne)
ds_tsne <- RenameIdents(ds_tsne, lab)
DimPlot(ds_tsne, reduction = "tsne",label = TRUE, pt.size = 0.5, label.size = 4) +
NoLegend()

################################################
#4) visualize the top 2 marker genes per cluster
################################################

top2markers=all_markers %>% group_by(cluster) %>% top_n(n = 2, wt = avg_logFC)
DoHeatmap(ds, features = top2markers$gene)

####################################
#5) Find novel marker genes (padj<0.05, detected at <50% frequency in the clusters
being compared)
####################################

feature0.markers <- FindMarkers(ds, ident.1 = "0", ident.2 = NULL, only.pos = TRUE)
feature0=feature0.markers[which(feature0.markers$p_val_adj<0.05),]
#marker genes that already exist
marker0=all_markers[which(all_markers$cluster==0),]
marker0=marker0$gene
#drop marker genes that exist
f0=feature0[-which(row.names(feature0) %in% marker0),]

feature1.markers <- FindMarkers(ds, ident.1 = "1", ident.2 = NULL, only.pos = TRUE)
feature1=feature1.markers[which(feature1.markers$p_val_adj<0.05),]
marker1=all_markers[which(all_markers$cluster==1),]
marker1=marker1$gene
f1=feature1[-which(row.names(feature1) %in% marker1),]

feature2.markers <- FindMarkers(ds, ident.1 = "2", ident.2 = NULL, only.pos = TRUE)
feature2=feature2.markers[which(feature2.markers$p_val_adj<0.05),]
marker2=all_markers[which(all_markers$cluster==2),]
marker2=marker2$gene
f2=feature2[-which(row.names(feature2) %in% marker2),]

feature3.markers <- FindMarkers(ds, ident.1 = "3", ident.2 = NULL, only.pos = TRUE)
```

```
feature3=feature3.markers[which(feature3.markers$p_val_adj<0.05),]
marker3=all_markers[which(all_markers$cluster==3),]
f3=feature3[-which(row.names(feature3) %in% marker3$gene),]

feature4.markers <- FindMarkers(ds, ident.1 = "4", ident.2 = NULL, only.pos = TRUE)
feature4=feature4.markers[which(feature4.markers$p_val_adj<0.05),]
marker4=all_markers[which(all_markers$cluster==4),]
f4=feature4[-which(row.names(feature4) %in% marker4$gene),]

feature5.markers <- FindMarkers(ds, ident.1 = "5", ident.2 = NULL, only.pos = TRUE)
feature5=feature5.markers[which(feature5.markers$p_val_adj<0.05),]
marker5=all_markers[which(all_markers$cluster==5),]
f5=feature5[-which(row.names(feature5) %in% marker5$gene),]

feature6.markers <- FindMarkers(ds, ident.1 = "6", ident.2 = NULL, only.pos = TRUE)
feature6=feature6.markers[which(feature6.markers$p_val_adj<0.05),]
marker6=all_markers[which(all_markers$cluster==6),]
f6=feature6[-which(row.names(feature6) %in% marker6$gene),]

feature.markers <- FindMarkers(ds, ident.1 = "7", ident.2 = NULL, only.pos = TRUE)
feature=feature.markers[which(feature.markers$p_val_adj<0.05),]
marker=all_markers[which(all_markers$cluster==7),]
f7=feature[-which(row.names(feature) %in% marker$gene),]

feature.markers <- FindMarkers(ds, ident.1 = "8", ident.2 = NULL, only.pos = TRUE)
feature=feature.markers[which(feature.markers$p_val_adj<0.05),]
marker=all_markers[which(all_markers$cluster==8),]
f8=feature[-which(row.names(feature) %in% marker$gene),]

feature.markers <- FindMarkers(ds, ident.1 = "9", ident.2 = NULL, only.pos = TRUE)
feature=feature.markers[which(feature.markers$p_val_adj<0.05),]
marker=all_markers[which(all_markers$cluster==9),]
f9=feature[-which(row.names(feature) %in% marker$gene),]

feature.markers <- FindMarkers(ds, ident.1 = "10", ident.2 = NULL, only.pos = TRUE)
feature=feature.markers[which(feature.markers$p_val_adj<0.05),]
marker=all_markers[which(all_markers$cluster==10),]
f10=feature[-which(row.names(feature) %in% marker$gene),]

feature.markers <- FindMarkers(ds, ident.1 = "11", ident.2 = NULL, only.pos = TRUE)
feature=feature.markers[which(feature.markers$p_val_adj<0.05),]
marker=all_markers[which(all_markers$cluster==11),]
f11=feature[-which(row.names(feature) %in% marker$gene),]

feature.markers <- FindMarkers(ds, ident.1 = "12", ident.2 = NULL, only.pos = TRUE)
feature=feature.markers[which(feature.markers$p_val_adj<0.05),]
marker=all_markers[which(all_markers$cluster==12),]
f12=feature[-which(row.names(feature) %in% marker$gene),]

feature.markers <- FindMarkers(ds, ident.1 = "13", ident.2 = NULL, only.pos = TRUE)
feature=feature.markers[which(feature.markers$p_val_adj<0.05),]
marker=all_markers[which(all_markers$cluster==13),]
f13=feature[-which(row.names(feature) %in% marker$gene),]
```

```
feature.markers <- FindMarkers(ds, ident.1 = "14", ident.2 = NULL, only.pos = TRUE)
feature=feature.markers[which(feature.markers$p_val_adj<0.05),]
marker=all_markers[which(all_markers$cluster==14),]
f14=feature[-which(row.names(feature) %in% marker$gene),]

################################################
#6) Output marker gene information
################################################
f0$cluster=rep(0,length(f0$p_val))
f1$cluster=rep(1,length(f1$p_val))
f2$cluster=rep(2,length(f2$p_val))
f3$cluster=rep(3,length(f3$p_val))
f4$cluster=rep(4,length(f4$p_val))
f5$cluster=rep(5,length(f5$p_val))
f6$cluster=rep(6,length(f6$p_val))
f7$cluster=rep(7,length(f7$p_val))
f8$cluster=rep(8,length(f8$p_val))
f9$cluster=rep(9,length(f9$p_val))
f10$cluster=rep(10,length(f10$p_val))
f11$cluster=rep(11,length(f11$p_val))
f12$cluster=rep(12,length(f12$p_val))
f13$cluster=rep(13,length(f13$p_val))
f14$cluster=rep(14,length(f14$p_val))

novel_markers=rbind(f0,f1,f2,f3,f4,f5,f6,f7,f8,f9,f10,f11,f12,f13,f14)
write.csv(novel_markers,'novel_markers.csv')
```