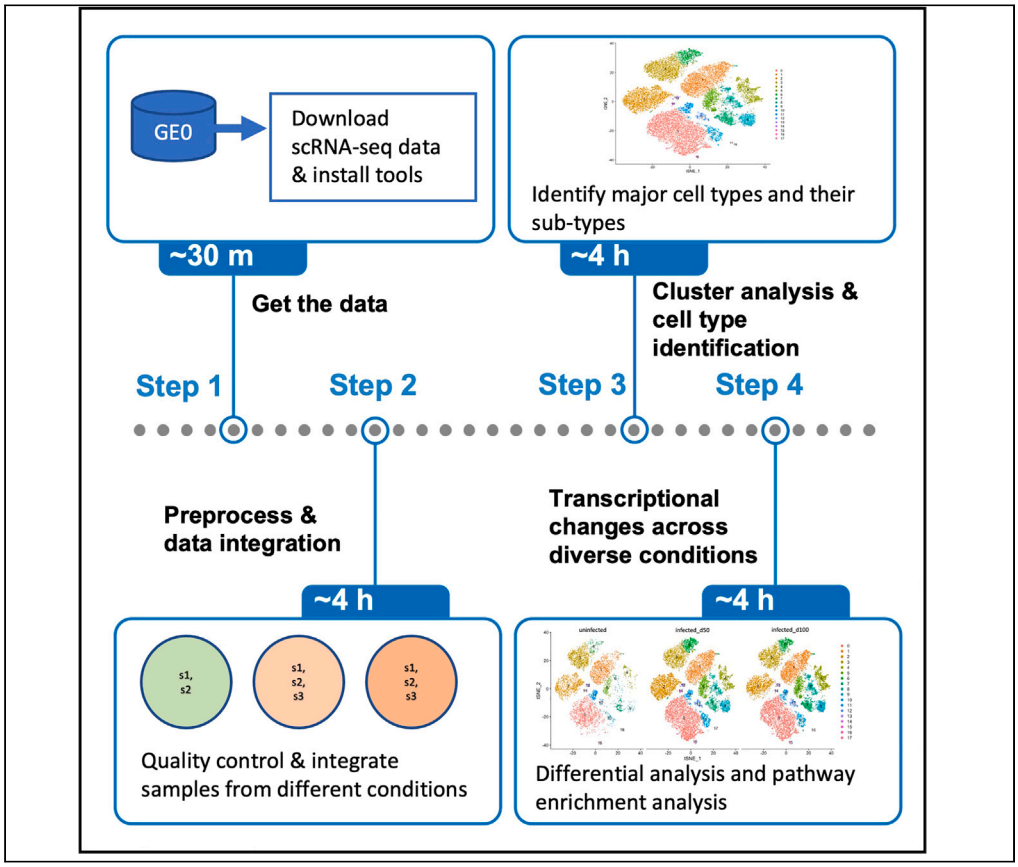
STAR Protocols

CellPress
OPEN ACCESS

Protocol

A protocol to analyze single-cell RNA-seq data from *Mycobacterium tuberculosis*-infected mice lung



Processing and analyzing single-cell RNA-seq (scRNA-seq) from lung cells are challenging due to the complexity of cell subtypes and biological variations within sample groups. Here, we present a protocol for performing an in-depth assessment on lung lymphocyte populations derived from healthy and *Mycobacterium tuberculosis*-infected mice. We describe steps for downloading processed scRNA-seq data, integrating samples across different conditions, and performing cluster analysis. We then detail procedures for identifying lymphoid cell subtypes, differential analysis, and pathway enrichment analysis.

Publisher's note: Undertaking any experimental protocol requires adherence to local institutional guidelines for laboratory safety and ethics.

Sadia Akter,
Shabaana A. Khader

khader@uchicago.edu

Highlights

scRNA-seq analysis of lymphocytes of healthy and *Mycobacterium tuberculosis*-infected mice lung

Pipeline will integrate samples across different conditions

Allows identification of cell types based on both known and unique markers

Explore transcriptional activities across diverse conditions

Akter & Khader, STAR Protocols 4, 102544 September 15, 2023
https://doi.org/10.1016/j.xpro.2023.102544

STAR Protocols

CellPress
OPEN ACCESS

Protocol

A protocol to analyze single-cell RNA-seq data from *Mycobacterium tuberculosis*-infected mice lung

Sadia Akter[1] and Shabaana A. Khader[1,2,3,*]

[1]Department of Microbiology, The University of Chicago, Chicago, IL 60637, USA

[2]Technical contact

[3]Lead contact

*Correspondence: khader@uchicago.edu
https://doi.org/10.1016/j.xpro.2023.102544

SUMMARY

Processing and analyzing single-cell RNA-seq (scRNA-seq) from lung cells are challenging due to the complexity of cell subtypes and biological variations within sample groups. Here, we present a protocol for performing an in-depth assessment on lung lymphocyte populations derived from healthy and *Mycobacterium tuberculosis*-infected mice. We describe steps for downloading processed scRNA-seq data, integrating samples across different conditions, and performing cluster analysis. We then detail procedures for identifying lymphoid cell subtypes, differential analysis, and pathway enrichment analysis.

For complete details on the use and execution of this protocol, please refer to Akter et al. (2022).[1]

BEFORE YOU BEGIN

Timing: 30 min

This manuscript details the steps necessary to identify lymphoid cell populations at the single cell level in the mouse lung. Comprehensive steps for single-cell RNA-seq data cluster analysis and cell type identification are discussed here. In different clusters, the process to perform differential analysis and enriched pathways while comparing among different groups is also described. In silico validations are also performed in Akter et al.[1]

1. Download the dataset.

   Note: The study includes uninfected (n = 2), *Mtb*-infected at 50 days post-infection (d50) (n = 3) and 100 days post infection (d100) (n = 3) scRNA-seq samples from mice lung.

   a. Download the dataset listed in the "Deposited data" section in the key resources table.
   b. Download the *cellranger* processed matrices where we have three files (barcodes.tsv.gz, features.tsv.gz, and matrix.mtx.gz) for each sample.
   c. Save these in folders named by each sample under a directory name "cellranger_out".
2. Install R on your machine.

   Note: Most of this protocol utilizes R language (R software v.3.5.3) for data processing and analysis.

KEY RESOURCES TABLE

REAGENT or RESOURCE	SOURCE	IDENTIFIER
Deposited data		
scRNA-seq, raw, and analyzed data	Akter et al.[1]	GEO: GSE200639
Software and algorithms		
Cellranger 3.1	10x Genomics	https://support.10xgenomics.com/
Seurat 3	Stuart et al.[2]	https://satijalab.org/seurat
ggplot2	Wickham[3]	https://cloud.r-project.org/package=ggplot2
biomaRt	Durinck et al.[4]	https://bioconductor.org/packages/biomaRt
PANTHER	Mi and Thomas[5]	http://www.pantherdb.org/

MATERIALS AND EQUIPMENT

Bioinformatics analysis

All bioinformatics analyses have been carried out on the University of Chicago's high-performance scientific compute platform (step 1-6 of the analysis) or on a local computer (step 7-8). We allocate a single CPU and 100 GB RAM in the institutional high-performance scientific computing environment. The local PC is a MacBook Pro with 2.3 GHz Quad-Core Intel Core i7 processor, 16 GB RAM).

STEP-BY-STEP METHOD DETAILS

Processing the single-cell RNA-seq data

Timing: ~ 4 h

These steps help researchers to preprocess the single-cell data and get it ready for downstream analysis once it comes out of the machine.

1. Quality Control.

Note: This section describes the process of quality control and decides on the filtering criteria of the scRNA-seq dataset including the installation of required R packages.

a. Installation and set up: Install all the R packages and set up the directories first and load the dataset in R.

Note: We use several R packages to generate figures and intermediate data preprocessing, such as Seurat[2] package (v.3.0.0), ggplot2,[3] and biomaRt.[4]

b. Explore the data: To decide on the filtering criteria, explore the data and plot the total genes per cell and total mitochondrial genes per cell on a log scale.
   i. First filter and count a gene only if it has non-zero reads mapped. Then plot genes per cell for each sample separately.
   ii. Second, count the percentage of mitochondrial genes and plot mitochondrial genes per cell for each sample separately.
   These steps help to decide on the filtering threshold.

```
# 1a. Installation and setting up the directories

install.packages('Seurat')

library(Seurat)

library(ggplot2)
```

```
#please replace the ''/PATH/TO/YOUR/DIRECTORY'' with your project directory

setwd(/PATH/TO/YOUR/DIRECTORY)

plots<- "./output/"

cellranger_out<-"./cellranger_output/"

# 1b: Explore the data #Change sample names as needed

samples<- c("TB1000A1","TB1000A2","TB1000A3","TB1000A4",

     "TB1000A5","TB1000A6","TB1000A7","TB1000A8")

for (i in 1:length(samples)){

  data <- Read10X(data.dir = paste0(cellranger_out, samples[i]))

genes_per_cell <- Matrix::colSums(data>0)

pdf(file=paste0(plots,samples[i],"_genes_per_cell_ordered.pdf"),paper="USr",width=11)

  par(oma=c(0,0,2,0))

  par(mfrow=c(1,1))

  plot(sort(genes_per_cell), xlab='cell', log='y', main='genes per cell (ordered)')

  title(main=paste0(samples[i]),outer=T)

  dev.off()

  mito.genes<-grep(pattern="^mt-",x = rownames(x=data), value=TRUE)

  mito_gene_read_counts = Matrix::colSums(data[mito.genes,])

# compute percentage of mitochondrial genes per cell

  pct_mito = mito_gene_read_counts / counts_per_cell * 100

pdf(file=paste0(plots,samples[i],"_mito_gene_per_cell.pdf"),paper="USr",width=11)

  par(oma=c(0,0,2,0))

  par(mfrow=c(1,1))

  plot(sort(pct_mito))

  title(main=paste0(samples[i]),outer=T)

  dev.off()

}
```

2. Create Seurat objects.

Read the scRNA-seq dataset and create Seurat objects. Add the percentage of mitochondrial genes in the Seurat metadata and the conditions of the samples: uninfected, infected at d50, and infected at d100.

```
## 2: create seurat objects

# Set up condition 1: uninfected object

for (file in c("TB1000A1","TB1000A2")){

  data <- Read10X(data.dir = paste0(cellranger_out, file))
```

```
  data_obj <- CreateSeuratObject(counts = data, min.features = 200, min.cells = 3, proj-
ect = file)

mito.genes <- grep(pattern = "^mt-", x = rownames(x = data_obj), value = TRUE)

percent.mito <- Matrix::colSums(x = GetAssayData(object = data_obj, slot = 'counts')[mito.-
genes, ]) / Matrix::colSums(x = GetAssayData(object = data_obj, slot = 'counts'))

  data_obj[['percent.mito']] <- percent.mito

  data_obj[["Dataset"]]<-file

  data_obj[["condition"]]<-"uninfected"

  assign(file, data_obj)

}

# Set up condition 2: infected d50 object

for (file in c("TB1000A3","TB1000A4","TB1000A5")){

  data <- Read10X(data.dir = paste0(cellranger_out, file))

  data_obj <- CreateSeuratObject(counts = data, min.features = 200, min.cells = 3, proj-
ect = file)

  mito.genes <- grep(pattern = "^mt-", x = rownames(x = data_obj), value = TRUE)

  percent.mito <- Matrix::colSums(x = GetAssayData(object = data_obj, slot = 'counts')[mi-
to.genes, ]) / Matrix::colSums(x = GetAssayData(object = data_obj, slot = 'counts'))

  data_obj[['percent.mito']] <- percent.mito

  data_obj[["Dataset"]]<-file

  data_obj[["condition"]]<-"infected_d50"

  assign(file, data_obj)

}

#Set up condition 3: infected d100 object

for (file in c("TB1000A6","TB1000A7","TB1000A8")){

  data <- Read10X(data.dir = paste0(cellranger_out, file))

  data_obj <- CreateSeuratObject(counts = data, min.features = 200, min.cells = 3, proj-
ect = file)

  mito.genes <- grep(pattern = "^mt-", x = rownames(x = data_obj), value = TRUE)

  percent.mito <- Matrix::colSums(x = GetAssayData(object = data_obj, slot = 'counts')[mi-
to.genes, ]) / Matrix::colSums(x = GetAssayData(object = data_obj, slot = 'counts'))

  data_obj[['percent.mito']] <- percent.mito

  data_obj[["Dataset"]]<-file

  data_obj[["condition"]]<-"infected_d100"

  assign(file, data_obj)

}

rm(data)

rm(data_obj)

gc()
```

3. Filter followed by normalization.
   a. Filter cells that have unique feature counts over certain threshold. Thresholds are decided based on the plots of step 1bi (genes_per_cell_ordered.pdf) where "sorted genes per cell" is more consistent within the thresholds and sharply falls otherwise.

   Note: We filter cells that have unique feature counts over 4,500 or less than 1,100.

   b. Set threshold for the percentage of mitochondrial genes based on the mitochondrial gene plots of step 1bii for each of the sample (mito_gene_per_cell.pdf). Beyond this threshold, there is a sharp increase in the plot.

   Note: We choose different thresholds for mitochondrial genes because of their distinct distribution in each sample.

   c. Merge samples with similar conditions.
   d. Normalize data with default parameters, and detect most variable genes using the *FindVariableFeatures* function.

   Note: All the samples of this study are sequenced in one batch. So, we merge samples with similar conditions as these are biological replicates. If samples are obtained from different batches, user should check and normalize for batch effects.

```
3. Filtering and normalization

TB1000A1 <- subset(TB1000A1, subset = nFeature_RNA >1100 & nFeature_RNA < 4500 & percent.
mito < 8)

TB1000A2 <- subset(TB1000A2, subset = nFeature_RNA >1100 & nFeature_RNA < 4500 & percent.
mito < 5)

# TB1000A1 and TB1000A2 are uninfected

TB1000A12 <- merge(x = TB1000A1, y = c(TB1000A2), add.cell.id = ("TB1000A1","TB1000A2"))

TB1000A12 <- NormalizeData(TB1000A12, verbose = FALSE)

TB1000A12 <- FindVariableFeatures(TB1000A12, selection.method = "vst", nfeatures = 2000)

TB1000A3 <- subset(TB1000A3, subset = nFeature_RNA >1100 & nFeature_RNA < 4500 & percent.
mito < 8)

TB1000A4 <- subset(TB1000A4, subset = nFeature_RNA >1100 & nFeature_RNA < 4500 & percent.
mito < 9)

TB1000A5 <- subset(TB1000A5, subset = nFeature_RNA >1100 & nFeature_RNA < 4500 & percent.
mito < 6)

# TB1000A3, TB1000A4 and TB1000A5 are infected at 50dpi

TB1000A345 <- merge(x = TB1000A3, y = c(TB1000A4,TB1000A5), add.cell.id = c("TB1000A3",
"TB1000A4","TB1000A5"))

TB1000A345 <- NormalizeData(TB1000A345, verbose = FALSE)

TB1000A345 <- FindVariableFeatures(TB1000A345, selection.method = "vst", nfeatures = 2000)

TB1000A6 <- subset(TB1000A6, subset = nFeature_RNA >1100 & nFeature_RNA < 4500 & percent.
mito < 6)

TB1000A7 <- subset(TB1000A7, subset = nFeature_RNA >1100 & nFeature_RNA < 4500 & percent.
mito < 7)
```

```
TB1000A8 <- subset(TB1000A8, subset = nFeature_RNA >1100 & nFeature_RNA < 4500 & percent.
mito < 8)

# TB1000A6, TB1000A7 and TB1000A8 are infected at 100dpi

TB1000A678 <- merge(x = TB1000A6, y = c(TB1000A7,TB1000A8), add.cell.id =c("TB1000A6",
"TB1000A7","TB1000A8"))

TB1000A678 <- NormalizeData(TB1000A678, verbose = FALSE)

TB1000A678 <- FindVariableFeatures(TB1000A678, selection.method = "vst", nfeatures = 2000)
```

4. Perform Integration and integrated analysis.
   a. Combine all samples using Seurat functions, *FindIntegrationAnchors* and *IntegrateData*.

   Note: This step integrates samples from different conditions.

   b. Use *ScaleData* to regress out a number of UMI's and mitochondrial content.
   c. Perform Principal component analysis (PCA) with *RunPCA*.
   d. Use *ElbowPlot* to visualize the standard deviation of each PCA.

   Note: *ElbowPlot* helps to determine how many PCAs are needed to capture most of the variation in the data. In this study, we found 17 PCAs can capture majority of the variations.

   e. Perform t-SNE dimensionality reduction on the scaled matrix.

   Note: We use the first 17 PCA for dimensionality reduction.

   f. The resolution parameter of the *FindClusters* function controls the granularity of the clustering, with higher values leading to more clusters.

   Note: There is no universally 'correct' resolution. However, user can try different values within this range and see which one best reflects data's biological variability.

   g. Next, visualize the clusters (Figure 1) and save the integrated Seurat object ("immune.combined").

```
# 4. Perform Integration and integrated analysis

immune.anchors <- FindIntegrationAnchors(object.list = list(TB1000A12, TB1000A345, TB1000A678),
dims = 1:20)

immune.combined <- IntegrateData(anchorset = immune.anchors, dims = 1:20)

# Run the standard workflow for visualization and clustering

all.genes <- rownames(immune.combined)

immune.combined <- ScaleData(immune.combined, features = all.genes)

#Perform linear dimensional reduction

immune.combined <- RunPCA(immune.combined, npcs = 30, features = VariableFeatures(object =
immune.combined))

DimPlot(immune.combined, reduction = "pca") + NoLegend()

#ElbowPlot which visualizes the standard deviation

ElbowPlot(immune.combined)
```

```
# Clustering
immune.combined <- FindNeighbors(immune.combined, reduction = "pca", dims = 1:17)
immune.combined <- FindClusters(immune.combined, resolution = c(0.4))
# tSNE and UMAP
immune.combined <- RunUMAP(immune.combined, reduction = "pca", dims = 1:17)
immune.combined <- RunTSNE(immune.combined, reduction = "pca", dims = 1:17)
#Figure 1
TSNEPlot(object = immune.combined, label=F)
saveRDS(immune.combined, file = "./immune.combined.rds")
```

Cluster analysis and cell type identification

Timing: ~ 4 h

These steps help researchers to perform cluster analysis and to know the process of identification of cell types.

5. Identify major cell types.
   Identify the non-immune and major immune cell types using the key genes and top marker genes.

   Note: Since in this study, most of the cells are lymphoid cells due to the prior purification for live cells,[1] we focus on characterizing the lymphoid cell subsets only.

   The composition of cell types varies across different tissues. For instance, brain tissue is characterized by the presence of neurons, astrocytes, oligodendrocytes, and microglia, whereas blood typically contains lymphocytes, neutrophils, monocytes, among others. Such cellular diversity or

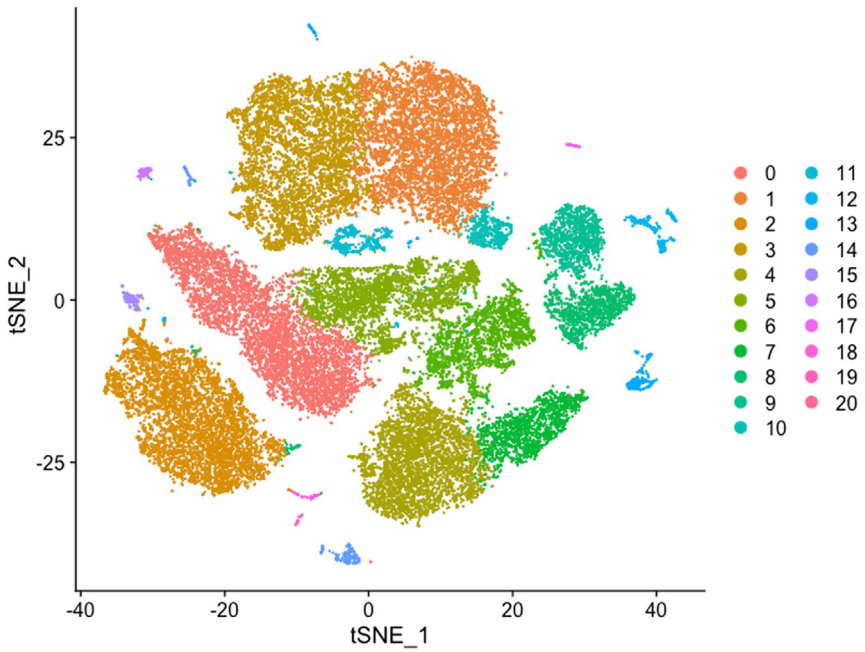


Figure 1. t-Distributed stochastic neighbor embedding (tSNE) visualization of different subsets of cells (all conditions together), colored according to different clusters

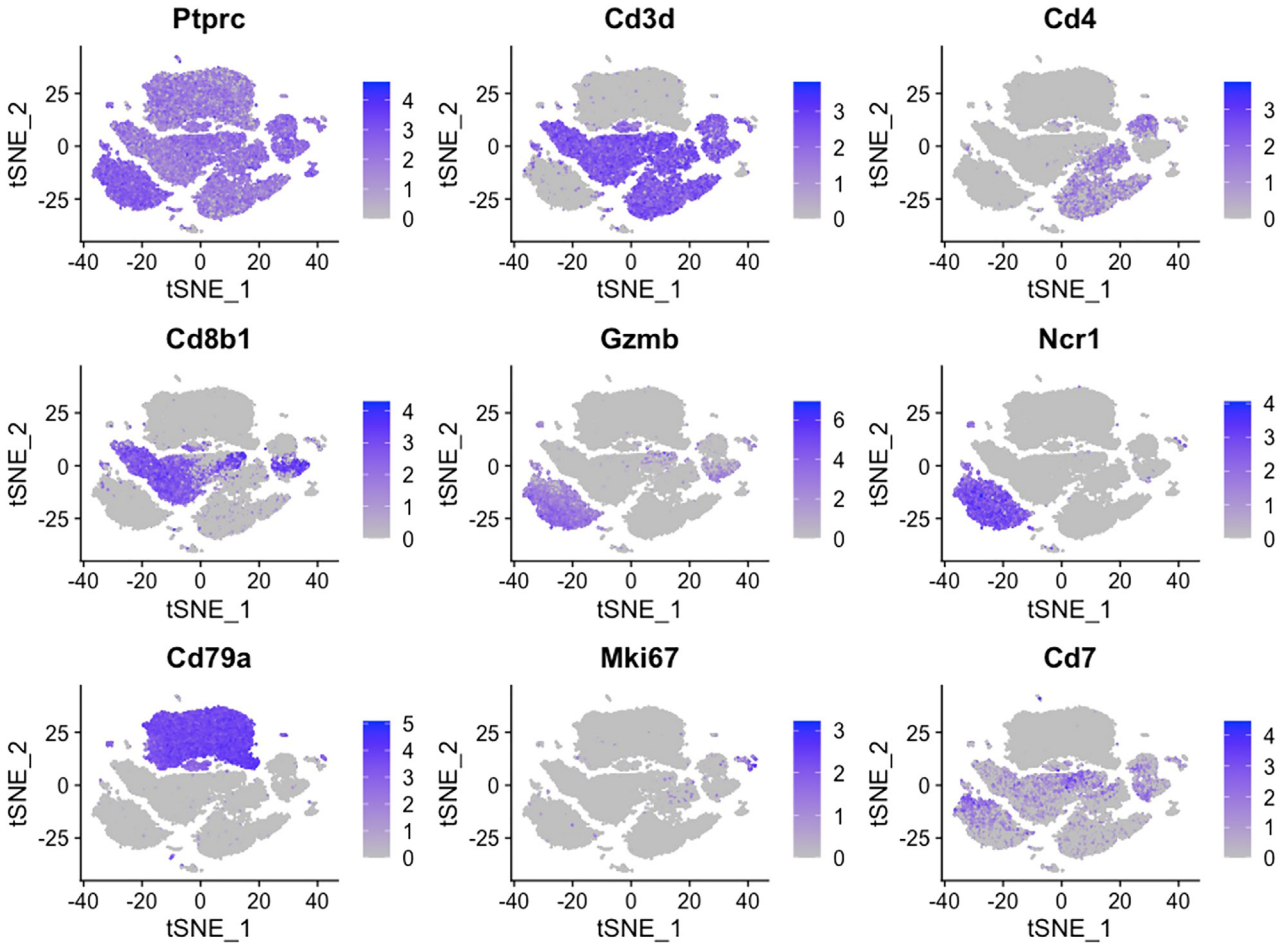


Figure 2. tSNE plot with the expression of different known cell markers

The expressions of these genes help to characterize distinct major cell types.

heterogeneity has significant implications for the interpretation of scRNA-seq data. Hence, it is crucial to account for these differences during clustering and differential expression analyses. Furthermore, tissue samples may be subject to batch effects. These effects could occur if the samples undergo processing at different time points or if they are handled by distinct individuals. Tools like Seurat and Harmony offer features for batch correction, including Integration and Harmony methods, that are invaluable in managing these potential discrepancies.

a. Identify the marker genes in each cluster and focused on the top 10 markers based on p-values.

Note: These are the key genes that help to understand the type of cells within the cluster. If the user uses a specific database to determine the marker genes of any particular cell type, the user should also consider determining if gene signatures in related tissue type as well, since gene expressions can vary based on the type of tissue.

b. Look into the expression of known marker genes.

Note: Immune cells can be identified as clusters expressing *Ptprc*, T cells expressing *CD3d*, *Cd4*, *CD8b1*, Natural killer (NK) cells expressing *Gzmb*, *Ncr1*, B cells expressing *CD79a*, etc

(Figure 2). Known maker genes helps to identify major cell types, and with that top markers of each cluster can enable further clustering of cell sub populations.

c. Name the clusters by their corresponding major cell types to visualize.[1]

Note: Identify the major cell types only as we will do re-clustering the lymphoid cells in the next step since majority of the cells in this dataset are lymphoid cells.

```
# 5a: Identify top 10 markers of each cluster

all.markers = FindAllMarkers(immune.combined, min.pct = 0.25, logfc.threshold = 0.25, only.pos = TRUE)

# sort all the markers by p-value

all.markers.sortedByPval = all.markers[order(all.markers$p_val),]

top10 <- all.markers.sortedByPval %>% group_by(cluster) %>% do(head(., n=10))

write.csv(top10,"top10_findallMarkers_onlyPos.csv")

# 5b: Look into the expression of known markers

DefaultAssay(immune.combined) <- "RNA"

#Figure 2

FeaturePlot(object = immune.combined, features = c( "Ptprc", "Cd3d",

"Cd4", "Cd8b1", "Gzmb","Ncr1","Cd79a", "Mki67", "Cd7"), cols = c("grey", "blue"), reduction = "tsne")

# 5c: tSNE visualization colored by major cell-types

new.cluster.ids <-

c("lymphoid","lymphoid","lymphoid","lymphoid","lymphoid","lymphoid","lymphoid","lymphoid","lymphoid",
"lymphoid","lymphoid",

        "lymphoid","myeloid","myeloid","non-immune","lymphoid","lymphoid","lymphoid","myeloid",
"myeloid","myeloid")

names(new.cluster.ids) <- levels(immune.combined)

immune.combined <- RenameIdents(immune.combined, new.cluster.ids)

TSNEPlot(object = immune.combined)
```

6. Re-cluster the lymphoid cells only and identify the cell subtypes. To identify the cell subtypes of each cluster, subset the lymphoid cells and perform cluster analysis.
   a. Re-clustering with lymphoid cells only.

   Note: The process is similar to the previous cluster analysis with all cells. To obtain a better resolution of cell subtypes, re-cluster analysis can be performed.

   b. Find top markers of each lymphoid cluster.

   Note: Top markers help to identify the cell subtypes of each cluster.

   c. Expression of top marker genes.

   Note: Investigate the expression of interested top marker genes.

   d. Expression of known marker genes.

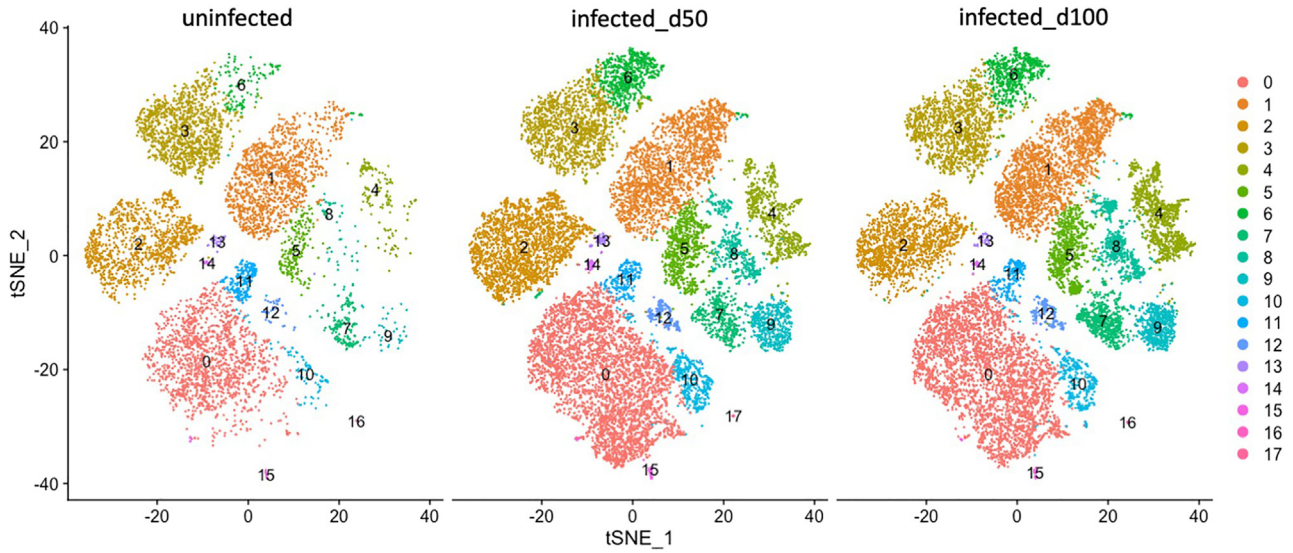


Figure 3. tSNE visualization of different subsets of lymphoid cells, colored according to cellular identity, split by conditions
Uninfected, n = 2; infected D50, n = 3; infected D100, n = 3.

Note: Investigate any known lymphoid marker genes along with top marker genes to know the cell type of a cluster. Using only one method is not enough to get a cell identity.[1]

e. Plot the clusters. Plot can be split for each condition (Figure 3).

Note: Clusters can also be renamed based on their respective cell identity and then visualized.

```
# 6a: Re-clustering with lymphoid cells only

DefaultAssay(immune.combined) <- "integrated"

lymphoid_cluster <- subset(immune.combined, idents =

c(0,1,2,3,4,5,6,7,8,9,10,11,15,16,17))

all.genes <- rownames(lymphoid_cluster)

lymphoid_cluster <- ScaleData(lymphoid_cluster, features = all.genes)

lymphoid_cluster <- RunPCA(lymphoid_cluster, npcs = 30, features = VariableFeatures(object =
lymphoid_cluster))

lymphoid_cluster <- RunTSNE(lymphoid_cluster, reduction = "pca", dims = 1:20)

lymphoid_cluster <- FindNeighbors(lymphoid_cluster, reduction = "pca", dims = 1:20)

lymphoid_cluster <- FindClusters(lymphoid_cluster, resolution = c(0.4))

saveRDS(lymphoid_cluster, file = "./lymphoid.combined.rds")

# 6b: Find markers of each lymphoid cluster

all.markers = FindAllMarkers(lymphoid_cluster, min.pct = 0.25, logfc.threshold = 0.25, on-
ly.pos = TRUE)

# sort all the markers by p-value

all.markers.sortedByPval = all.markers[order(all.markers$p_val),]

top10 <- all.markers.sortedByPval %>% group_by(cluster) %>% do(head(., n=10))
```

```
write.csv(top10,"top10_findallMarkers_lymphoid_onlyPos.csv")

# 6c: Expression of top genes in different clusters

markers.to.plot <- c("Ebf1","Ifi30","Ifit3","Ifi27l2a","Vpreb3", "Spib", Sp140","Rtp4","Fcmr",
"Ms4a1","Mzb1","Il7r","Tcf7","Lef1","Ctla4","Cxcr3",

"Cxcr6","Icos","Maf","Tnfrsf4","Cd7","Ctla2a","Il2rb","Ccl5","Cx3cr1","Ze-
b2","Lag3","Cd40lg","Stat1","Igtp","Ifng","Rora","S100a4","Gbp4","Gbp2","F-
cer1g","Gzmb", "Gzma","Gzmk","Tnfrsf17", "Foxp3")

DotPlot(lymphoid_cluster, features = rev(markers.to.plot), cols = c("blue", "red"), dot.s-
cale = 8) + RotatedAxis()

# 6d: Expression of known marker genes

DefaultAssay(lymphoid_cluster) <- "RNA"

FeaturePlot(object = lymphoid_cluster, features = c("Ptprc","Cd79a","Ms4a1","Cd3d","C-
d3e","Cd4","Cd8b1","Cd8a","Gzmb","Ncr1", "Sell","Ccr7","Cd44","Ifng","Rora","Foxp3"),
cols = c("grey", "blue"), reduction = "tsne")

#Figure 3

DimPlot(lymphoid_cluster, reduction = "tsne",label = TRUE, split.by = "condition")

# 6e: Rename the clusters with respective subtypes

new.cluster.ids <- c("B naive","CD8+T naive","NK_1","CD4+T naive","CD4+T

act_1","CD8+T eff","CD4+T IFN+","CD8+T act_1","CD8+T act_2",

          "CD4+T act_2","B act_1","B mem","B/T

doublets","NK_2","NK_3","B act_2","Plasma_1","Plasma_2")

lymphoid_cluster <- RenameIdents(lymphoid_cluster, new.cluster.ids)

#To visualize the clusters by labeling the cell sub-types

#all together

DimPlot(lymphoid_cluster, reduction = "tsne",label = TRUE)

#split by conditions

DimPlot(lymphoid_cluster, reduction = "tsne",label = TRUE, split.by = "condition")
```

Transcriptional differences among different conditions

Timing: ~ 4 h

These steps help researchers to perform differential analysis of samples from different conditions. This section also discusses the process pathway enrichment analysis using PANTHER.[5]

Note: In this study, we want to identify the transcriptional changes in the mouse lung due to the *Mtb* infection.

7. Differential analysis.

Use *Findmarkers* from Seurat for the differential analysis with default parameters. Change the assay to "RNA" before performing the analysis.

Note: In this study, we compare infected vs uninfected mice separately for d50 and d100 (*infected d50 vs uninfected* and *infected d50 vs uninfected*), and for each cluster at a time point.

```
# 7: Differential analysis to compare infected vs uninfected mice

lymphoid_cluster$condition.DE <- paste(Idents(lymphoid_cluster), lymphoid_cluster$condition, sep = "_")

lymphoid_cluster$condition <- Idents(lymphoid_cluster)

Idents(lymphoid_cluster) <- "condition.DE"

total_clusters<-18

for (i in 0:(total_clusters-1)){

  rm(infected_d50_vs_uninfected)

  rm(infected_d100_vs_uninfected)

  infected_d50_vs_uninfected<- FindMarkers(lymphoid_cluster, ident.1 = paste0(i,"_infec_d50"), ident.2 = paste0(i,"_uninfected"), verbose = FALSE)

  infected_d100_vs_uninfected<- FindMarkers(lymphoid_cluster, ident.1 = paste0(i,"_infec_d100"), ident.2 = paste0(i,"_uninfected"), verbose = FALSE)

  infected_d50_vs_uninfected<- subset(infected_d50_vs_uninfected, p_val_adj<0.05)

  infected_d100_vs_uninfected<- subset(infected_d100_vs_uninfected, p_val_adj<0.05)

write.csv(infected_d50_vs_uninfected,paste0("./DE_cluster_",i,"_infected_d50_vs_uninfected.csv"), row.names = FALSE)

write.csv(infected_d100_vs_uninfected,paste0("./DE_cluster_",i,"_infected_d100_vs_uninfected.csv"), row.names = FALSE)

}
```

8. Pathway enrichment analysis.

Perform over-representation test using the PANTHER website.[5] PANTHER is a web-based tool where user can provide the differential gene list, choose the parameters and the software enables the analysis. This need to be done separately for each gene list (differentially expressed genes) and up/down regulated.

Note: We use the *Reactome* pathway database with Fisher's exact test and corrected for multiple tests using a false discovery rate with default parameter settings as calculated by the Benjamini-Hochberg procedure.[6]

EXPECTED OUTCOMES

This protocol guides the process of analyzing scRNA-seq data from mouse lungs to perform cluster analysis, identify enriched genes, cell type identification of each cluster, and differential analysis among different conditions in each cluster followed by identification of enriched pathways.

Using this protocol, one can identify the major types of lymphoid cells, for example, $Cd4^+$/$Cd8^+$ T cells, B cells, NK cells and different sub-types of T/B (memory, naïve, activated) cells.

LIMITATIONS

This study has some limitations. For example, this protocol focuses mainly on the lymphoid cells. Most of the cells were lymphoid cells due to the prior purification for live cells, which resulted in

the enrichment of lymphoid populations to ~96% of the total cells; only 2.7% were myeloid and 0.82% were non-immune cells.[1] As such we have not discussed other type of cells and related markers. Further, we use *IntegrateData* from Seurat to integrate samples, which requires intensive memory for large genomes (e.g., human, monkey) and big sample size. In such cases, researchers need to use high-performance computing resources to perform the analysis.

TROUBLESHOOTING

Problem 1

Some steps of the data processing might require a higher processor and memory.

Potential solution

It is better to do the sample integration in high-performance computer if the data size is big and/or the genome is large. We have eight samples from three different conditions. Therefore, we use Seurat integration method, *IntegrateData,* which requires intensive memory. This method is suggested to use for batch effect correction and to perform comparative scRNA-seq analysis across different conditions. Alternatively, researchers can use Harmony.[7]

Problem 2

Assigning cell sub-type is critical. No tool or software can solely identify all cell subtypes.

Potential solution

It is better to investigate different databases as well as literature to identify cell sub-types. The latest peer reviewed articles on the same organism and same source of cells can be helpful to identify the cell identity. Also, it is better to determine the cell sub-types based on both the list of top marker genes and the expression of known markers. One can also apply existing tools to classify cell-types, such as Azimuth,[8] SC3,[9] CellAssign,[10] and SingleR.[11] These tools usually use machine learning or other statistical methods to differentiate between cell types based on unique gene expression patterns. The choice of tool depends on researcher's specific goals and available resources.

Problem 3

This manuscript and its reference article focus on lung lymphocytes only and so it discusses about the known marker genes of lymphocytes.

Potential solution

Since about 96% cells of the dataset used here are lymphoid cells due to method of isolation,[1] we focus on the markers of lymphoid cells only. If you have other type of cells, such as myeloid, non-immune cells, you can look into resources such as Human cell atlas, databases like CellMarker,[12] and PanglaoDB[13] and latest peer reviewed research articles.

Problem 4

A cluster might have cells that express mix type of cell markers, such as both lymphoid and myeloid.

Potential solution

If you have any cluster that has cells expressing both lymphoid and myeloid markers, those could potentially be doublets. It is crucial to carefully examine these instances and exclude them from subsequent analyses.

Problem 5

Sometimes user might face difficulty to load the data into R.

Potential solution

Please ensure that the files are present in the directory from which user is attempting to access the files. It's also important to verify the accuracy of the file and folder names.

RESOURCE AVAILABILITY

Lead contact

Further information and requests for resources should be directed to and will be fulfilled by the lead contact, Shabaana A. Khader (khader@uchicago.edu).

Materials availability

This study did not generate any new unique reagents.

Data and code availability

The accession number of the raw and processed data for single-cell RNA sequencing has been deposited in GEO. DOIs are listed in the key resources table. Any additional information required to reanalyze the data reported in this paper is available from the lead contact upon request.

ACKNOWLEDGMENTS

This work was supported by the University of Chicago and NIH grants HL105427, AI111914, AI134236, AI155024, and AI123780 to S.A.K.

AUTHOR CONTRIBUTIONS

Conceptualization, S.A.K.; investigation, S.A. and S.A.K.; formal analysis and data curation, S.A. and S.A.K.; visualization, S.A.; writing – original draft, S.A. and S.A.K.; writing – review and editing, S.A. and S.A.K.; funding acquisition, S.A.K.; supervision, S.A.K.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

1. Akter, S., Chauhan, K.S., Dunlap, M.D., Choreño-Parra, J.A., Lu, L., Esaulova, E., Zúñiga, J., Artyomov, M.N., Kaushal, D., and Khader, S.A. (2022). Mycobacterium tuberculosis infection drives a type I IFN signature in lung lymphocytes. Cell Rep. *39*, 110983. https://doi.org/10.1016/j.celrep.2022.110983.

2. Stuart, T., Butler, A., Hoffman, P., Hafemeister, C., Papalexi, E., Mauck, W.M., Hao, Y., Stoeckius, M., Smibert, P., and Satija, R. (2019). Comprehensive Integration of Single-Cell Data. Cell *177*, 1888–1902.e21. https://doi.org/10.1016/j.cell.2019.05.031.

3. Wickham, H. (2016). Data Analysis (Springer-Verlag New York), pp. 189–201. https://doi.org/10.1007/978-3-319-24277-4_9.

4. Durinck, S., Spellman, P.T., Birney, E., and Huber, W. (2009). Mapping identifiers for the integration of genomic datasets with the R/Bioconductor package biomaRt. Nat. Protoc. *4*, 1184–1191. https://doi.org/10.1038/nprot.2009.97.

5. Mi, H., and Thomas, P. (2009). PANTHER Pathway: An Ontology-Based Pathway Database Coupled with Data Analysis Tools. Methods Mol. Biol. *563*, 123–140. https://doi.org/10.1007/978-1-60761-175-2_7.

6. Benjamini, Y., and Hochberg, Y. (1995). Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. J. Roy. Stat. Soc. B *57*, 289–300. https://doi.org/10.2307/2346101.

7. Korsunsky, I., Millard, N., Fan, J., Slowikowski, K., Zhang, F., Wei, K., Baglaenko, Y., Brenner, M., Loh, P.R., and Raychaudhuri, S. (2019). Fast, sensitive and accurate integration of single-cell data with Harmony. Nat. Methods *16*, 1289–1296. https://doi.org/10.1038/s41592-019-0619-0.

8. Hao, Y., Hao, S., Andersen-Nissen, E., Mauck, W.M., Zheng, S., Butler, A., Lee, M.J., Wilk, A.J., Darby, C., Zager, M., et al. (2021). Integrated analysis of multimodal single-cell data. Cell *184*, 3573–3587.e29. https://doi.org/10.1016/j.cell.2021.04.048.

9. Kiselev, V.Y., Kirschner, K., Schaub, M.T., Andrews, T., Yiu, A., Chandra, T., Natarajan, K.N., Reik, W., Barahona, M., Green, A.R., and Hemberg, M. (2017). SC3: consensus clustering of single-cell RNA-seq data. Nat. Methods *14*, 483–486. https://doi.org/10.1038/nmeth.4236.

10. Zhang, A.W., O'Flanagan, C., Chavez, E.A., Lim, J.L.P., Ceglia, N., McPherson, A., Wiens, M., Walters, P., Chan, T., Hewitson, B., et al. (2019). Probabilistic cell-type assignment of single-cell RNA-seq for tumor microenvironment profiling. Nat. Methods *16*, 1007–1015. https://doi.org/10.1038/s41592-019-0529-1.

11. Aran, D., Looney, A.P., Liu, L., Wu, E., Fong, V., Hsu, A., Chak, S., Naikawadi, R.P., Wolters, P.J., Abate, A.R., et al. (2019). Reference-based analysis of lung single-cell sequencing reveals a transitional profibrotic macrophage. Nat. Immunol. *20*, 163–172. https://doi.org/10.1038/s41590-018-0276-y.

12. Zhang, X., Lan, Y., Xu, J., Quan, F., Zhao, E., Deng, C., Luo, T., Xu, L., Liao, G., Yan, M., et al. (2019). CellMarker: a manually curated resource of cell markers in human and mouse. Nucleic Acids Res. *47*, D721–D728. https://doi.org/10.1093/nar/gky900.

13. Franzén, O., Gan, L.-M., and Björkegren, J.L.M. (2019). PanglaoDB: a web server for exploration of mouse and human single-cell RNA sequencing data. Database *2019*, baz046. https://doi.org/10.1093/database/baz046.