'FIRM' Package for flexible integration of scRNA-seq data

September 20, 2022

1 Overview

This vignette provides an introduction to the 'FIRM' package. R package 'FIRM' implements FIRM, an algorithm for flexible integration of heterogeneous scRNA-seq datasets across multiple tissue types, platforms and experimental batches.

The FIRM package can be loaded with the command:

```
R> library("FIRM")
```

Also load required package:

```
R> library(Seurat)
R> library(RANN)
```

2 Workflow

In this vignette, we use the ExampleData in the package which is the aorta data from Tabula Muris generated using Smart-seq2 (SS2) and 10X. The 'ExampleData$SS2' and 'ExampleData$tenx' provide the gene expression matrices $\mathbf{X}$ for SS2 and 10X respectively, where X_{ij} is the number of reads (for SS2) and unique molecular identified (UMI, for 10X). We only considered the cells with annotations in the original study. The cell type annotations for SS2 and 10X datasets are provided in 'ExampleData$meta_SS2' and 'ExampleData$meta_tenx'.

```
R> data("ExampleData")

R> dim(SS2)

[1] 23341   408

R> length(meta_SS2)

[1] 408

R> dim(tenx)

[1] 23433   526

R> length(meta_tenx)

[1] 526
```

2.1 Data preprocessing

We performed the standard preprocessing workflow to prepare the scaled data for integration.

```
R> prep_SS2 <- prep_data(SS2)
R> Dataset1 <- prep_SS2$Dataset
R> hvg1 <- prep_SS2$hvg
R> prep_tenx <- prep_data(tenx)
R> Dataset2 <- prep_tenx$Dataset
R> hvg2 <- prep_tenx$hvg
```

2.2 Integration using FIRM

The integrated data is provided in '`Dataset`'.

```
R> dims <- 15
R> coreNum <- 4
R> Dataset <- FIRM(Dataset1, Dataset2, all_genes = FALSE,
+                  hvg1, hvg2, dims = dims, coreNum = coreNum)

R> dim(Dataset)

[1] 1453  934
```

For downstream analysis which focuses on the highly variable genes shared by the datasets, such as dimension reduction, clustering and visualization, we set '`all_genes = FALSE`' to improve the computational efficiency and memory usage, which is the default setting in '`FIRM`' function. For downstream analysis which requires full gene expression profiles, we set '`all_genes = TRUE`' to obtain the harmonized data for all genes.

```
R> Dataset <- FIRM(Dataset1, Dataset2, all_genes = TRUE,
+                  hvg1, hvg2, dims = dims, coreNum = coreNum)

R> dim(Dataset)

[1] 23433   934
```

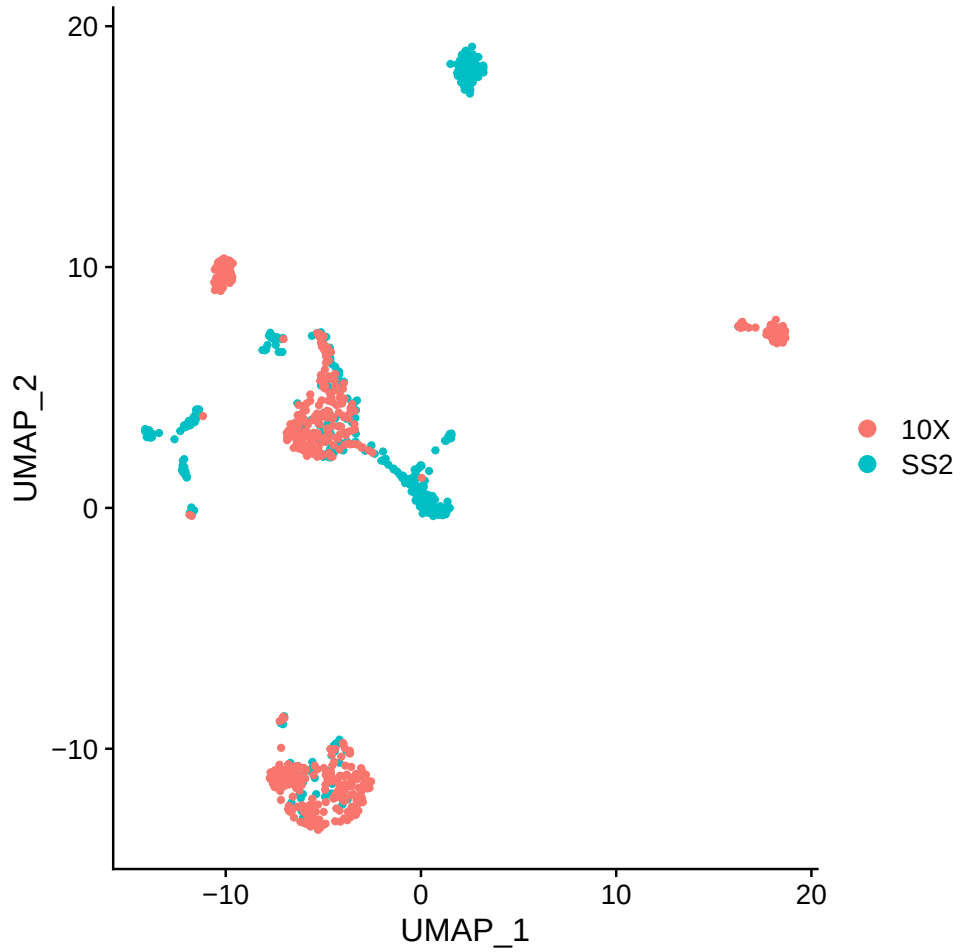
2.3 Downstream analysis

We can use the integrated data for downstream analysis, such as PCA and visualization. For example, we can create a Seurat object.

```
R> # counts
R> integrated_counts <- matrix(0, nrow(Dataset), ncol(Dataset))
R> rownames(integrated_counts) <- rownames(Dataset)
R> colnames(integrated_counts) <- colnames(Dataset)
R> integrated_counts[rownames(SS2), colnames(SS2)] <- as.matrix(SS2)
R> integrated_counts[rownames(tenx), colnames(tenx)] <- as.matrix(tenx)
R> # create Seurat object
R> integrated <- CreateSeuratObject(integrated_counts)
R> # normalization
```

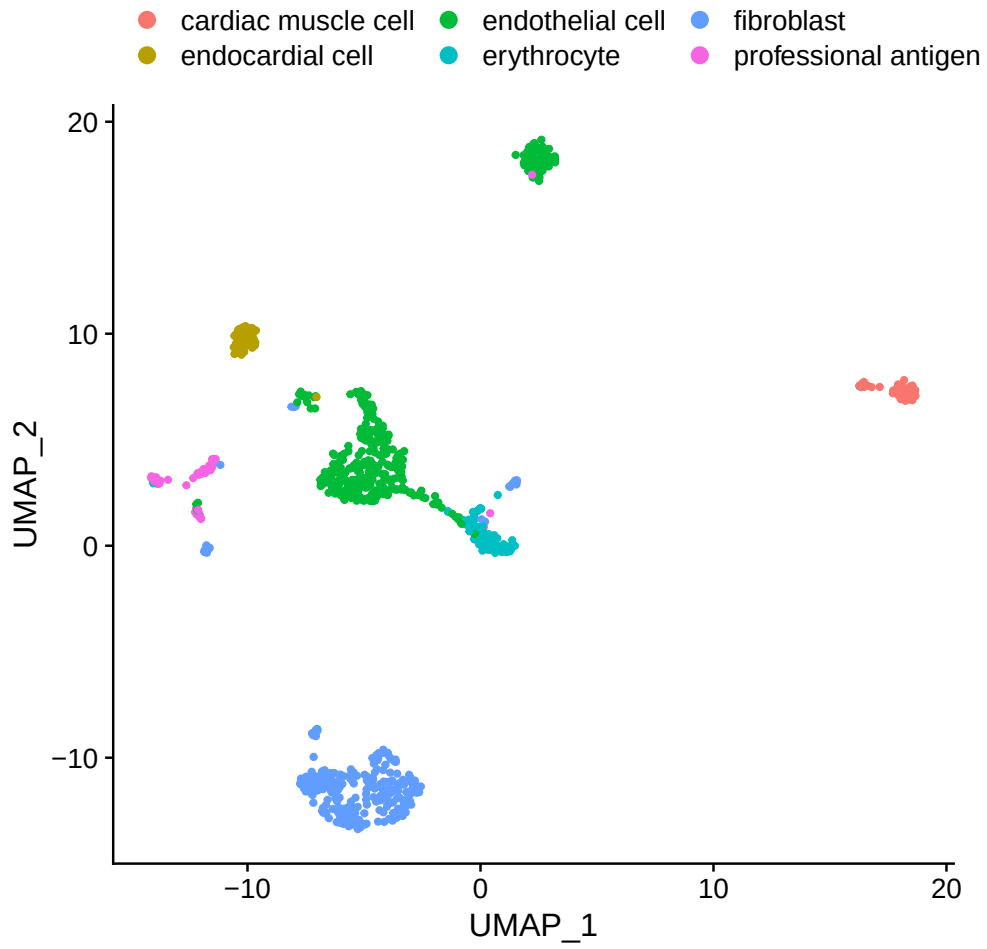
```
R> integrated <- NormalizeData(integrated)
R> # put the centered integrated data in scaled data
R> integrated[["RNA"]]@scale.data <- Dataset - rowMeans(Dataset)
R> # add meta data
R> integrated <- AddMetaData(integrated, metadata = c(meta_SS2, meta_tenx),
+                            col.name = "annotation")
R> integrated <- AddMetaData(integrated,
+                            metadata = c(rep("SS2", ncol(SS2)), rep("10X", ncol(tenx))),
+                            col.name = "dataset")
R> # hvg for PCA and visualization
R> hvg <- intersect(hvg1, hvg2)
R> # PCA
R> integrated <- RunPCA(integrated, features = hvg, npcs = dims)
R> # UMAP
R> integrated <- RunUMAP(integrated, reduction = "pca", dims = 1:dims,
+                         umap.method = 'umap-learn', metric = "correlation")

R> DimPlot(integrated, reduction = "umap", group.by = "dataset")
```




```
R> library(ggplot2)
```

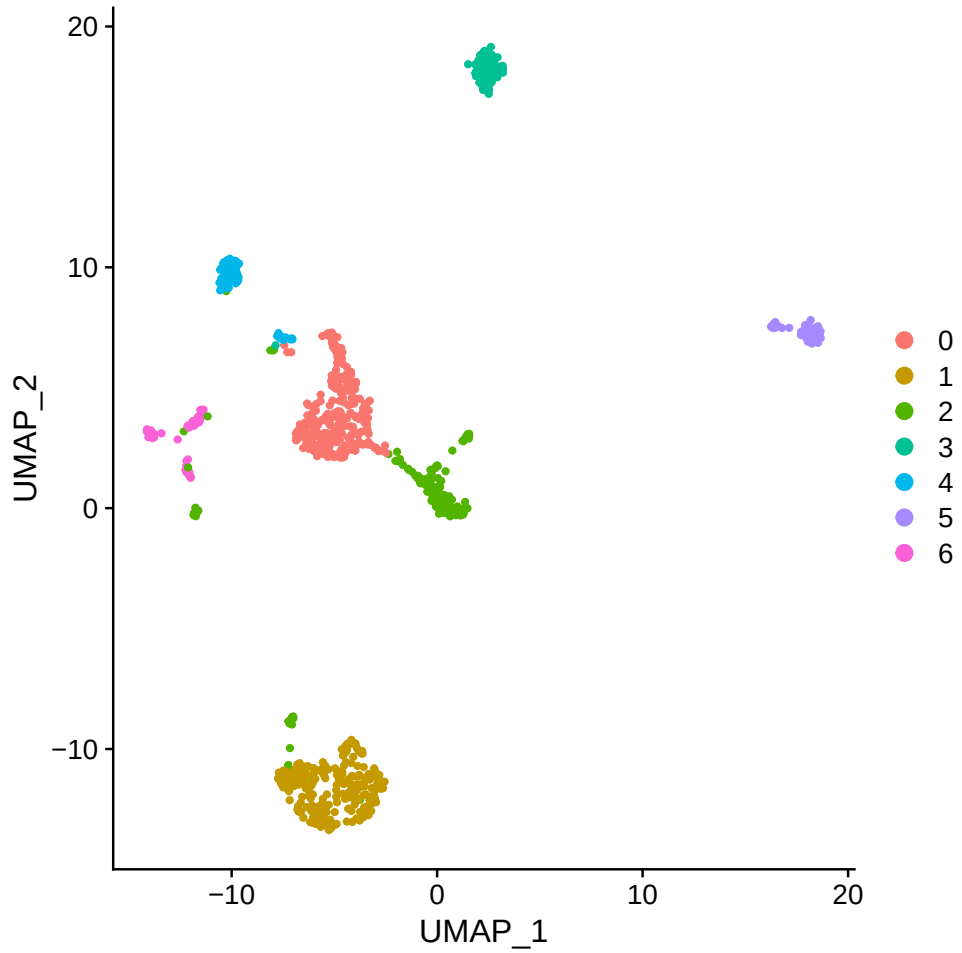
```
R> DimPlot(integrated, reduction = "umap", group.by = "annotation") +
+   theme(legend.position = "top")
```



Example codes for clustering

```
R> integrated <- FindNeighbors(integrated)
R> integrated <- FindClusters(integrated, resolution = 0.2, verbose = FALSE)

R> DimPlot(integrated, reduction = "umap", group.by = "seurat_clusters")
```



Example codes for differential gene expression analysis

```
R> markers <- FindMarkers(integrated, ident.1 = "0", ident.2 = "1",
+                         group.by = "seurat_clusters", slot = "scale.data")
R> head(markers)
```

```
              p_val   avg_diff pct.1 pct.2    p_val_adj
Col1a2 5.172882e-90 -1138.9138 0.119 0.992 1.212161e-85
Cdh5   1.775604e-89   122.5765 0.980 0.048 4.160774e-85
Col3a1 4.430898e-89 -1388.6182 0.146 0.992 1.038292e-84
Cfh    7.287236e-89  -271.1148 0.170 0.996 1.707618e-84
Dpt    8.087541e-89  -673.1153 0.178 0.996 1.895153e-84
Cd36   3.768867e-88  1790.9721 1.000 0.224 8.831586e-84
```

References

[1] Jingsi Ming, Zhixiang Lin, Jia Zhao, Xiang Wan, The Tabula Microcebus Consortium, Can Yang, Angela Ruohao Wu, FIRM: Flexible Integration of single-cell RNA-sequencing data for large-scale Multi-tissue cell atlas datasets. Briefings in Bioinformatics (2022). https://academic.oup.com/bib/advance-article/doi/10.1093/bib/bbac167/6585621

[2] Schaum, N. et al. Single-cell transcriptomics of 20 mouse organs creates a Tabula Muris. Nature 562, 367–372 (2018).

[3] Stuart, T. et al. Comprehensive Integration of Single-Cell Data. Cell 177, 1888-1902.e21 (2019).