Application of VAM to 10x PBMC 3k scRNA-seq data using Seurat log normalization for the MSigDB Hallmark collection.

H. Robert Frost

1 Load the VAM package

Loading VAM will also load the required packages MASS and Matrix. Seurat is referenced via suggests so must be directly loaded to enable access to Seurat functions.

```
> library(VAM)
> library(Seurat)
```

2 Load and process the 10x PBMC scRNA-seq data

This example uses the same 10x PBMC scRNA-seq data set that is used in the Seurat Guided Clustering vignette
(https://satijalab.org/seurat/v3.1/pbmc3k_tutorial.html). The Cell Ranger files for this data set can be downloaded from
https://s3-us-west-2.amazonaws.com/10x.files/samples/cell/pbmc3k/pbmc3k_filtered_gene_bc_matrices.tar.gz. This data is loaded and processed using the same Seurat logic found in the Guided Clustering vignette. In particular, the Seurat log normalization method implemented by NormalizeData() is used with variable genes determined by FindVariableFeatures(). This method for variable feature determination decomposes the measured variance for each gene into biological and technical components and provides the values of technical variance input to the VAM algorithm.

```
> # update the data.dir argument to reflect the local location of the PBMC data
> pbmc.data = Read10X(data.dir = "./filtered_gene_bc_matrices/hg19/")
> pbmc = CreateSeuratObject(counts = pbmc.data, project = "pbmc3k", min.cells = 3,
+         min.features = 200)
> pbmc[["percent.mt"]] = PercentageFeatureSet(pbmc, pattern = "^MT-")
> pbmc = subset(pbmc, subset = nFeature_RNA > 200 & nFeature_RNA < 2500 & percent.mt < 5)
> pbmc = NormalizeData(pbmc)
> pbmc = FindVariableFeatures(pbmc, selection.method = "vst", nfeatures = 2000)
> pbmc = ScaleData(pbmc, features = rownames(pbmc))
> pbmc = RunPCA(pbmc, features = VariableFeatures(object = pbmc))
> pbmc = RunUMAP(pbmc, dims = 1:10)
> pbmc = FindNeighbors(pbmc, dims = 1:10)
> pbmc = FindClusters(pbmc, resolution = 0.5)

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

Number of nodes: 2638
Number of edges: 96033

Running Louvain algorithm...
```

```
Maximum modularity in 10 random starts: 0.8720
Number of communities: 9
Elapsed time: 0 seconds

> # Assign the known cell type labels to the clusters (this follows the Seurat vingette)
> cell.types = c("Memory CD4 T",  "CD14+ Mono", "Naive CD4 T", "B", "CD8 T", "FCGR3A+ Mono",
+                         "NK", "DC", "Platelet")
> names(cell.types) = levels(pbmc)
> pbmc = RenameIdents(pbmc, cell.types)
> pbmc

An object of class Seurat
13714 features across 2638 samples within 1 assay
Active assay: RNA (13714 features, 2000 variable features)
 2 dimensional reductions calculated: pca, umap
```

3 Load Ensembl IDs

The Ensembl IDs and gene names must be read in from the genes.tsv file and filtered to match genes left after the quality control steps performed in the prior section.

```
> feature.data = read.delim("./filtered_gene_bc_matrices/hg19/genes.tsv",
+         header = FALSE, stringsAsFactors = FALSE)
> ensembl.ids = feature.data[,1]
> gene.names = feature.data[,2]
> genes.after.QC = rownames(pbmc@assays$RNA@counts)
> indices.to.keep = unlist(sapply(genes.after.QC, function(x) {which(gene.names == x)[1]}))
> ensembl.ids = ensembl.ids[indices.to.keep]
> gene.names = gene.names[indices.to.keep]
```

4 Load the MSigDB Hallmark collection

The following logic loads the MSigDB Hallmark collection using the msigdbr R package. Because the MSigDB gene sets are defined in terms of Entrez gene IDs, these must be mapped to Ensembl IDs using the org.HS.eg.db R package (from Bioconductor). The data frame returned by msigdbr is then converted into a list of gene ID vectors (each list element corresponds to a gene set and is a vector of Ensembl IDs). This list is transformed into the structure required by vamForSeurat() using the createGeneSetCollection() helper function. This helper function filters out genes not also contained in the PBMC scRNA-seq data and generates a list whose elements are vectors of gene indices in the scRNA-seq data.

```
> library(msigdbr)
> library(org.Hs.eg.db)
> # Load the MSigDB Hallmark collection using the msigdbr package
> H.collection = msigdbr(category="H")
> # Get the entrez gene IDs that are mapped to an Ensembl ID
> entrez2ensembl = mappedkeys(org.Hs.egENSEMBL)
> # Convert to a list
> entrez2ensembl = as.list(org.Hs.egENSEMBL[entrez2ensembl])
> # Convert Entrez IDs to Ensembl IDs using the org.Hs.eg.db package
> # Doing this a rather simplistic (and inefficient) way for clarity
```

```
> msigdb.entrez.ids = H.collection$entrez_gene
> num.ids = length(msigdb.entrez.ids)
> msigdb.ensembl.ids = rep(NA, num.ids)
> for (i in 1:num.ids) {
+         entrez.id = msigdb.entrez.ids[i]
+         id.index = which(names(entrez2ensembl) == entrez.id)
+         if (length(id.index > 0)) {
+           # only use the first mapped ensembl id
+           msigdb.ensembl.ids[i] = entrez2ensembl[[id.index]][1]
+         }
+ }
> # Save the ensembl IDs in the data frame
> H.collection$ensembl_gene = msigdb.ensembl.ids
> # Create a gene.set.collection list of Ensembl IDs
> gene.set.names = unique(H.collection$gs_name)
> num.sets = length(gene.set.names)
> gene.set.collection = list()
> for (i in 1:num.sets) {
+         gene.set.name = gene.set.names[i]
+         gene.set.rows = which(H.collection$gs_name == gene.set.name)
+         gene.set.ensembl.ids = H.collection$ensembl_gene[gene.set.rows]
+         gene.set.collection[[i]] = gene.set.ensembl.ids
+ }
> names(gene.set.collection) = gene.set.names
> # Create the collection list required by vamForSeurat()
> gene.set.collection = createGeneSetCollection(gene.ids=ensembl.ids,
+         gene.set.collection=gene.set.collection)
> length(gene.set.collection)

[1] 50
```

5 Execute VAM method

Since the scRNA-seq data has been processed using Seurat, we execute VAM using the vamForSeurat() function. We have set return.dist=T so that the squared adjusted Mahalanobis distances will be returned in a "VAMdist" Assay.

```
> pbmc = vamForSeurat(seurat.data=pbmc,
+     gene.set.collection=gene.set.collection,
+     center=F, gamma=T, sample.cov=F, return.dist=T)
```

Look at the first few entries in the "VAMdist" and "VAMcdf" Assays.

```
> pbmc@assays$VAMdist[1:5,1:5]

5 x 5 sparse Matrix of class "dgCMatrix"
                             AAACATACAACCAC-1 AAACATTGAGCTAC-1 AAACATTGATCAGC-1
HALLMARK-ADIPOGENESIS              98.166655       440.153976       449.653452
HALLMARK-ALLOGRAFT-REJECTION      461.961078       573.666648       479.882138
HALLMARK-ANDROGEN-RESPONSE        270.083708       181.536163       237.574914
HALLMARK-ANGIOGENESIS               5.985118         8.191464         7.099042
```

```
HALLMARK-APICAL-JUNCTION           187.012689       201.455359       256.959324
                           AAACCGTGCTTCCG-1 AAACCGTGTATGCG-1
HALLMARK-ADIPOGENESIS             254.45880         85.93626
HALLMARK-ALLOGRAFT-REJECTION      525.37851        852.28732
HALLMARK-ANDROGEN-RESPONSE        173.69509        368.83380
HALLMARK-ANGIOGENESIS              28.40593         52.57082
HALLMARK-APICAL-JUNCTION          191.29709        212.65132
```

```
> pbmc@assays$VAMcdf[1:5,1:5]

5 x 5 sparse Matrix of class "dgCMatrix"
                           AAACATACAACCAC-1 AAACATTGAGCTAC-1 AAACATTGATCAGC-1
HALLMARK-ADIPOGENESIS             0.1046634        0.8487740        0.8582855
HALLMARK-ALLOGRAFT-REJECTION      0.3677062        0.5720654        0.4015772
HALLMARK-ANDROGEN-RESPONSE        0.5721414        0.2977352        0.4767420
HALLMARK-ANGIOGENESIS             0.2478960        0.2979079        0.2740947
HALLMARK-APICAL-JUNCTION          0.4496806        0.4984440        0.6622745
                           AAACCGTGCTTCCG-1 AAACCGTGTATGCG-1
HALLMARK-ADIPOGENESIS             0.5264715        0.0788544
HALLMARK-ALLOGRAFT-REJECTION      0.4865253        0.8916543
HALLMARK-ANDROGEN-RESPONSE        0.2726758        0.7911543
HALLMARK-ANGIOGENESIS             0.5784735        0.7485725
HALLMARK-APICAL-JUNCTION          0.4643582        0.5347018
```

6 Compute DE pathways

Use the Seurat FindAllMarkers() function to identify Hallmark pathways whose VAM scores are enriched within each cell type cluster according to a Wilcoxon test.

```
> library(dplyr)
> pbmc.markers = FindAllMarkers(pbmc, assay="VAMcdf", only.pos = TRUE, logfc.threshold = 0.01)
> pbmc.markers %>% group_by(cluster) %>% top_n(n = 3, wt = avg_logFC)

# A tibble: 26 x 7
# Groups:   cluster [9]
       p_val avg_logFC pct.1 pct.2 p_val_adj cluster    gene
       <dbl>     <dbl> <dbl> <dbl>     <dbl> <fct>      <chr>
 1 1.46e- 14    0.0778 1     1     7.29e- 13 Memory CD~ HALLMARK-MYC-TARGETS-V1
 2 2.41e-  3    0.0318 0.989 0.968 1.21e-  1 Memory CD~ HALLMARK-MYC-TARGETS-V2
 3 1.65e- 31    0.140  1     1     8.23e- 30 CD14+ Mono HALLMARK-MYC-TARGETS-V1
 4 4.23e-  7    0.0660 1     1     2.11e-  5 CD14+ Mono HALLMARK-UNFOLDED-PROTE~
 5 2.15e-  5    0.0491 1     0.998 1.07e-  3 CD14+ Mono HALLMARK-E2F-TARGETS
 6 1.49e-184    0.452  1     1     7.44e-183 Naive CD4~ HALLMARK-REACTIVE-OXYGE~
 7 4.95e-127    0.363  1     1     2.48e-125 Naive CD4~ HALLMARK-COMPLEMENT
 8 1.75e-116    0.353  1     1     8.75e-115 Naive CD4~ HALLMARK-INFLAMMATORY-R~
 9 7.15e- 45    0.159  1     1     3.58e- 43 B          HALLMARK-ALLOGRAFT-REJE~
10 1.45e- 18    0.151  0.948 0.878 7.25e- 17 B          HALLMARK-KRAS-SIGNALING~
# ... with 16 more rows
```

7 Visualize VAM scores

Visualize VAM scores using Seurat DoHeatmap(). The default Assay must first be changed to "VAMcdf" and the slot parameter must be set to "data" in the call to DoHeatmap().

```
> library(ggplot2)
> DefaultAssay(object = pbmc) = "VAMcdf"
> top.pathways <- pbmc.markers %>% group_by(cluster) %>% top_n(n = 3, wt = avg_logFC)
> DoHeatmap(pbmc, slot="data", features = top.pathways$gene,  size=3, label=T) + NoLegend()
```