Supplementary Note 1 : Seurat alignment tutorial

Andrew Butler and Rahul Satija

7/26/2017

This tutorial walks through a basic alignment of two different datasets of peripheral blood mononuclear cells (PBMCs): one provided by 10X genomics and one from the recent Seq-well paper (Gierahn et. al, 2017). We first read in the two count matrices and set up the Seurat objects.

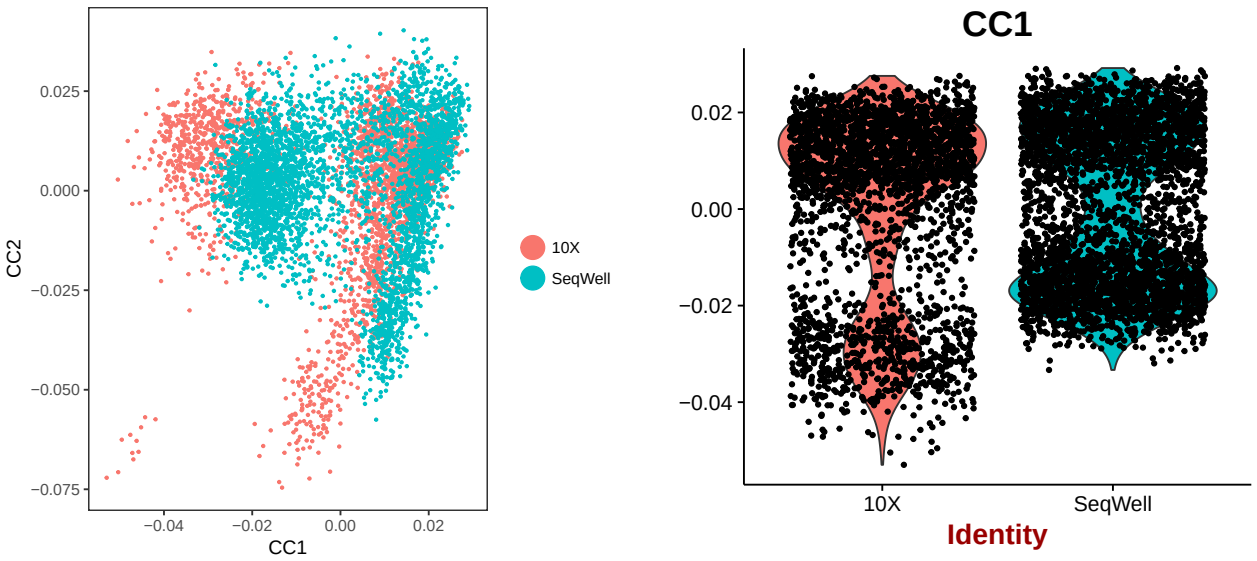
Next, we select the genes we want to use in the alignment procedure. Here we take the union of the top 2,000 genes with the highest dispersion (var/mean) from both datasets.

```
library(Seurat)
# cowplot enables side-by-side ggplots
library(cowplot)

# load data
seqwell.data <- read.table(paste0("~/Downloads/IntegratedAnalysis_ExpressionMatrices/",
                  "pbmc_SeqWell.expressionMatrix.txt"))
tenx.data <- read.table(paste0("~/Downloads/IntegratedAnalysis_ExpressionMatrices/",
                  "pbmc_10X.expressionMatrix.txt"))


# setup Seurat objects
# since both count matrices have already filtered cells, we do no additional filtering here

seqwell <- CreateSeuratObject(raw.data = seqwell.data)
seqwell <- NormalizeData(seqwell)
seqwell <- ScaleData(seqwell)
seqwell <- FindVariableGenes(seqwell, do.plot = F)

tenx <- CreateSeuratObject(raw.data = tenx.data)
tenx <- NormalizeData(tenx)
tenx <- ScaleData(tenx)
tenx <- FindVariableGenes(tenx, do.plot = F)

# we will take the union of the top 2k variable genes in each dataset for alignment
# note that we use 1k genes in the manuscript examples, you can try this here with
# negligible changes to the overall results
hvg.seqwell <- rownames(head(seqwell@hvg.info, 2000))
hvg.tenx <- rownames(head(tenx@hvg.info, 2000))
hvg.union <- union(hvg.seqwell, hvg.tenx)

# lastly, we set the "protocol" in each dataset for easy identification later
# it will be transferred to the merged object in RunCCA
tenx@meta.data[, "protocol"] <- "10X"
seqwell@meta.data[, "protocol"] <- "SeqWell"
```

We next run a canonical correlation analysis to identify common sources of variation between the two datasets. RunCCA will also combine the two objects into a single object and stores the canonical correlation vectors (the vectors that project each dataset into the maximally correlated subspaces). We also store the original dataset identity as a column in @meta.data

```
pbmc <- RunCCA(tenx, seqwell, genes.use = hvg.union)
```

```
# visualize results of CCA
# plot CC1 versus CC2 and look at a violin plot
p1 <- DimPlot(pbmc, reduction.use = "cca", group.by = "protocol", pt.size = 0.5,
              do.return = T)
p2 <- VlnPlot(pbmc, features.plot = "CC1", group.by = "protocol", do.return = T)
plot_grid(p1, p2)
```




```
PrintDim(pbmc, reduction.type = "cca", dims.print = 1:2, genes.print = 10)
```

```
## [1] "CC1"
##  [1] "FTL"    "TYROBP" "AIF1"   "FCER1G" "LYZ"    "FTH1"   "CTSS"
##  [8] "CST3"   "S100A9" "CD68"
## [1] ""
##  [1] "RPL31"  "RPS27A" "RPS29"  "RPS27"  "RPS12"  "RPS25"  "RPS6"
##  [8] "RPL30"  "RPS15A" "RPS14"
## [1] ""
## [1] ""
## [1] "CC2"
##  [1] "NKG7"   "CCL5"   "CST7"   "GZMB"   "PRF1"   "GZMA"   "GNLY"
##  [8] "FGFBP2" "KLRD1"  "CTSW"
## [1] ""
##  [1] "HLA-DRA"  "RPS16"    "HLA-DRB1" "RPS11"    "PABPC1"   "HLA-DRB5"
##  [7] "RPS9"     "HLA-DQA1" "RPS8"     "RPL8"
## [1] ""
## [1] ""
```

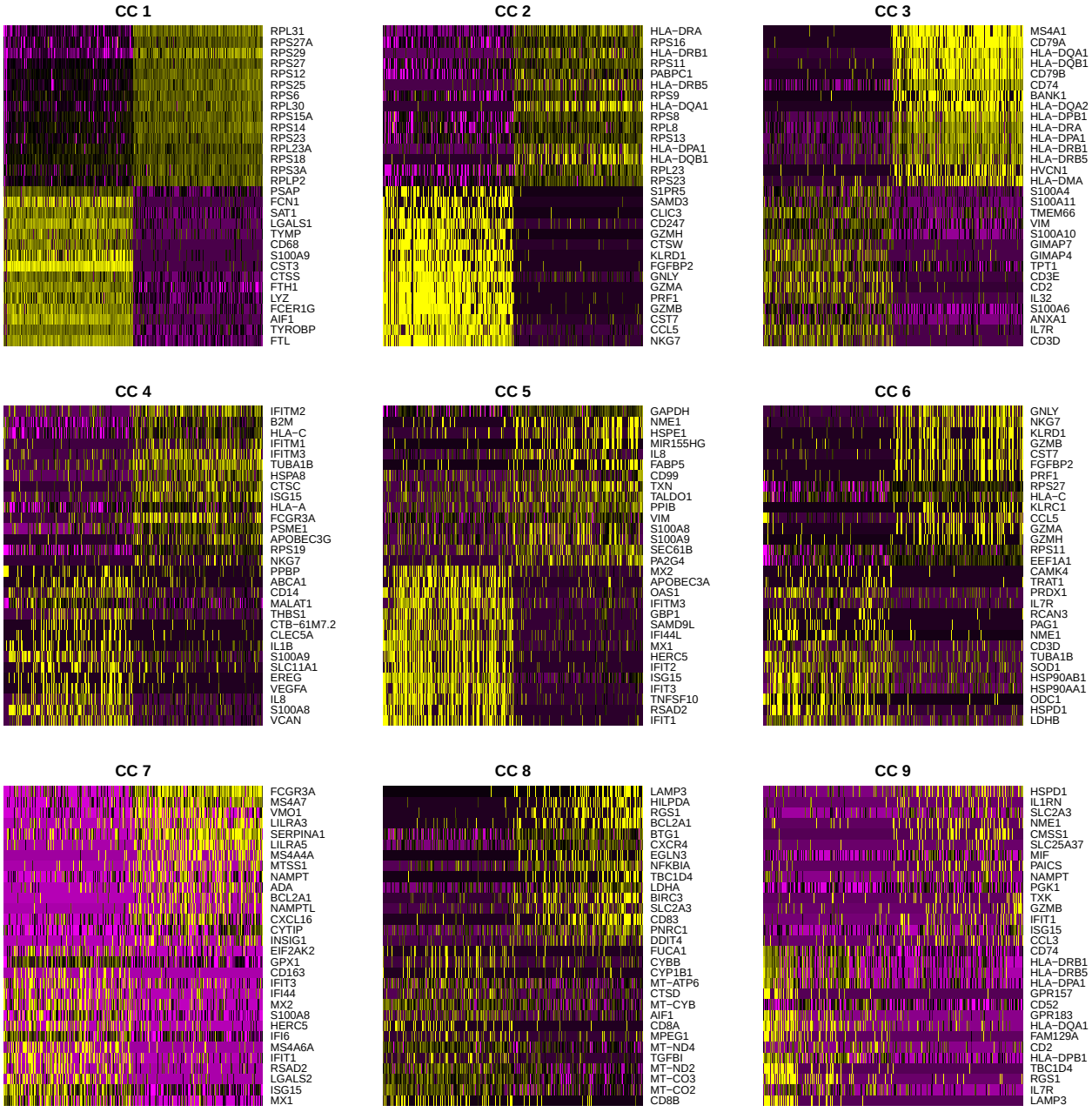
You can see, for example, that CC1 and CC2 separate myeloid from lymphoid cells in both datasets, but the values remain 'shifted' relative to each other. We need to choose CCs for downstream analysis and then 'align them'

Before this, we search for cells whose expression level cannot be well-explained by low-dimensional CCA, compared to low-dimensional PCA.

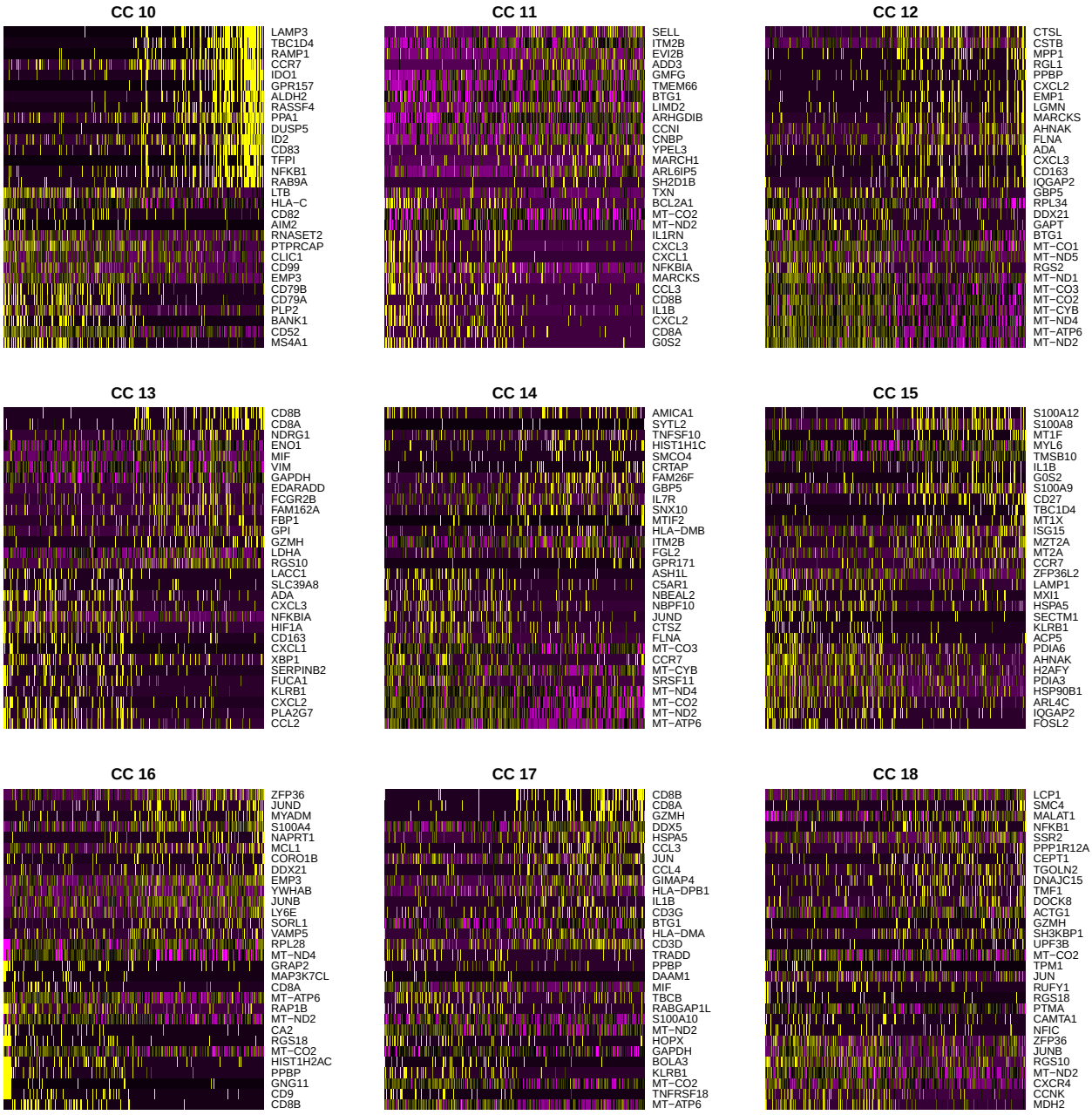
The problem of choosing CCs for downstream analysis is similar to choosing PCs for clustering. We are developing resampling based procedures for this, but here explore the CC dimensions as we have previously demonstrated for PCA. We begin to see drop-off in signal after CC13, so we chose

CC1...13 for analysis. You can try modifying this parameter (i.e. 1...15 or even 1...20) without significant changes in the results

```
DimHeatmap(pbmc, reduction.type = "cca", cells.use = 500, dim.use = 1:9, do.balanced = T)
```




```
DimHeatmap(pbmc, reduction.type = "cca", cells.use = 500, dim.use = 10:18, do.balanced = T)
```



Before we align the subspaces, we first search for cells whose expression profile cannot be well-explained by low-dimensional CCA, compared to low-dimensional PCA.

```
pbmc <- CalcVarExpRatio(pbmc, reduction.type = "pca", grouping.var = "protocol", dims.use = 1:13)

# We discard cells where the variance explained by CCA is <2-fold  (ratio < 0.5) compared to PCA
pbmc.all.save <- pbmc
pbmc <- SubsetData(pbmc, subset.name = "var.ratio.pca", accept.low = 0.5)
```

For illustrative purposes, we can also look at the discarded cells. Note that discarded cells tend to have lower gene counts, but a subset also express high levels of PF4. This is because megakaryocytes are only present in the 10X dataset, and thus are correctly discarded as 'dataset-specific']

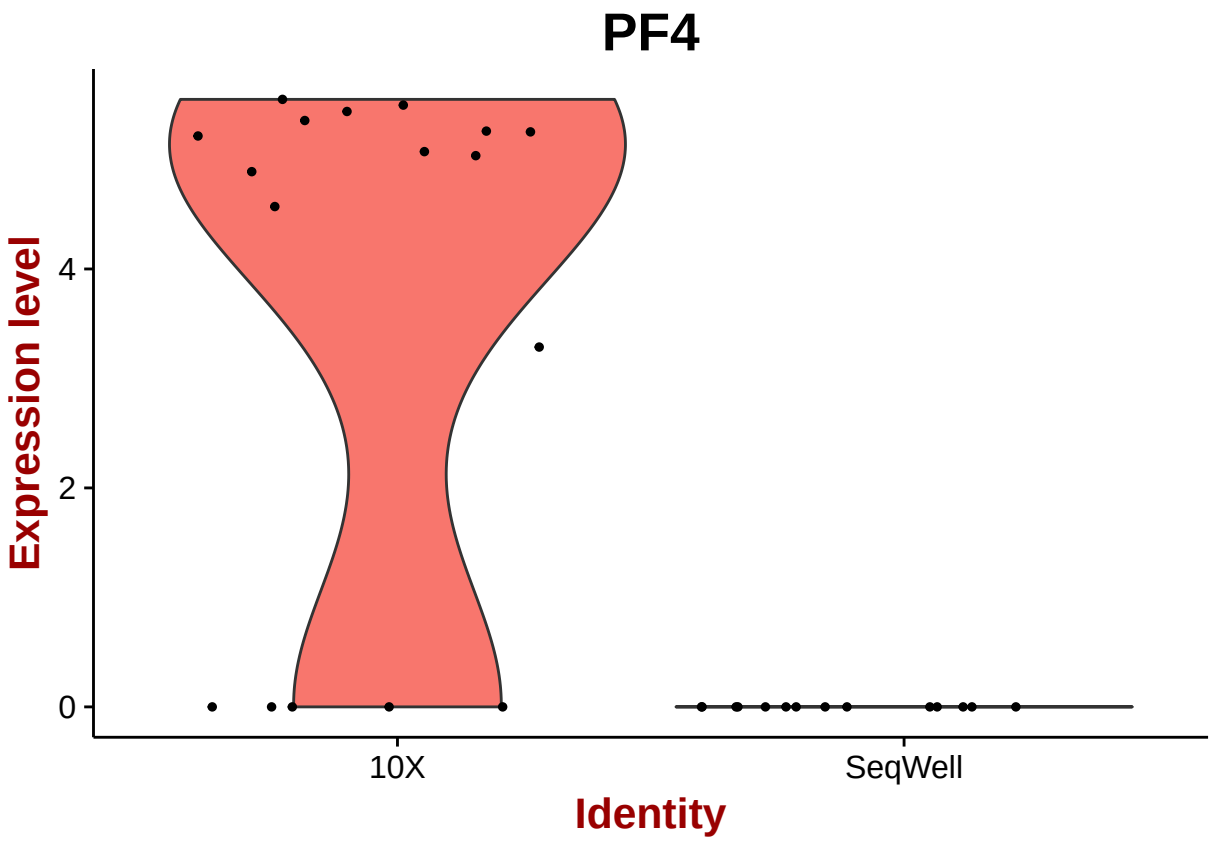
```
pbmc.discard <- SubsetData(pbmc.all.save, subset.name = "var.ratio.pca", accept.high = 0.5)
median(pbmc@meta.data[, "nGene"])
```

```
## [1] 843
```

```
median(pbmc.discard@meta.data[, "nGene"])
```

```
## [1] 624
```

```
VlnPlot(pbmc.discard, features.plot = "PF4", group.by = "protocol")
```

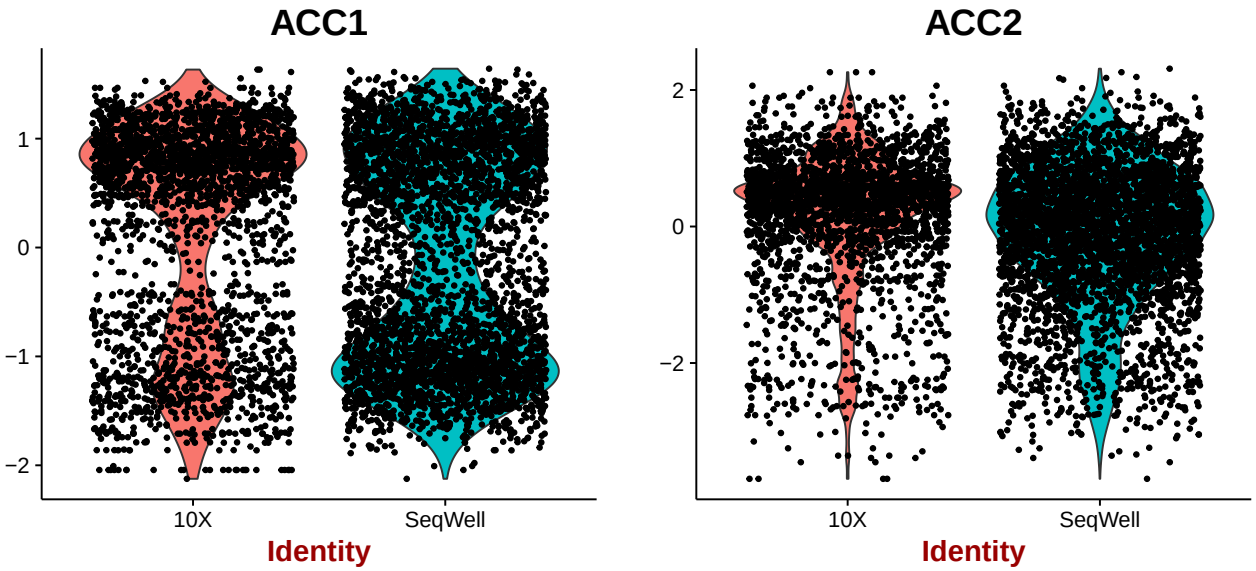


Now we align the CCA subspaces, which returns a new dimensional reduction called cca.aligned

```
pbmc <- AlignSubspace(pbmc, reduction.type = "cca", grouping.var = "protocol", dims.align = 1:13)
```

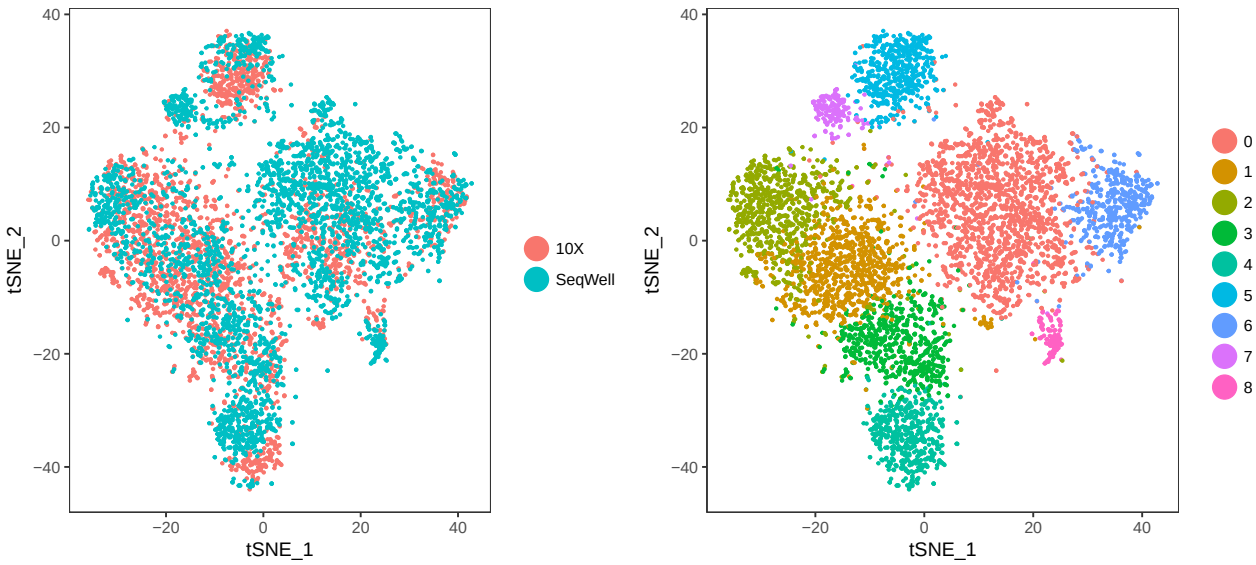
Visualize the aligned CCA and perform integrated analysis

```
p1 <- VlnPlot(pbmc, features.plot = "ACC1", group.by = "protocol", do.return = T)
p2 <- VlnPlot(pbmc, features.plot = "ACC2", group.by = "protocol", do.return = T)
plot_grid(p1, p2)
```



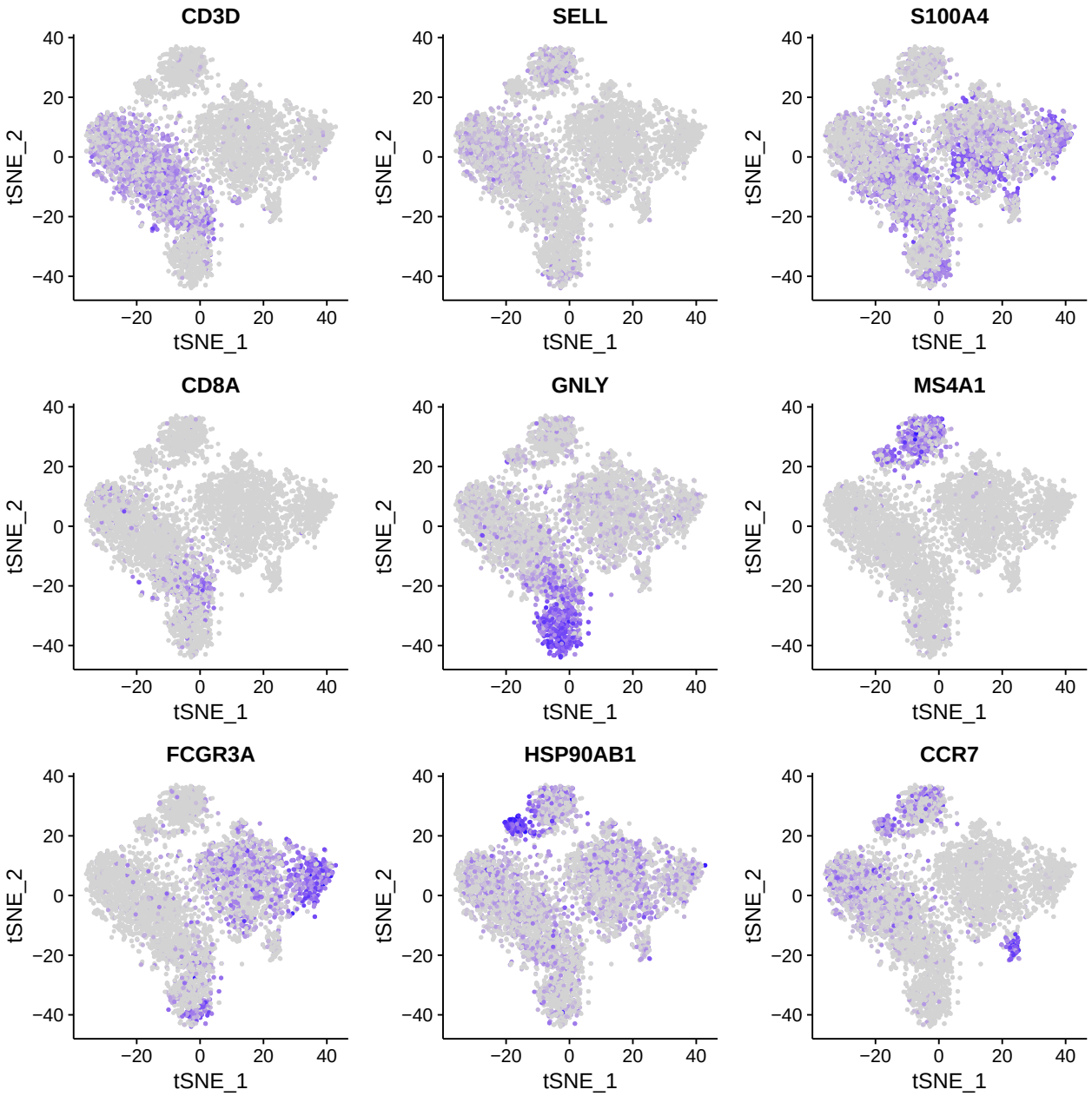
Now we can run a single integrated analysis on all cells!

```
pbmc <- RunTSNE(pbmc, reduction.use = "cca.aligned", dims.use = 1:13, do.fast = T)
pbmc <- FindClusters(pbmc, reduction.type = "cca.aligned", dims.use = 1:13, save.SNN = T)
p1 <- TSNEPlot(pbmc, group.by = "protocol", do.return = T, pt.size = 0.5)
p2 <- TSNEPlot(pbmc, do.return = T, pt.size = 0.5)
plot_grid(p1, p2)
```



Now, we annotate the clusters as before based on canonical markers.

```
FeaturePlot(pbmc, features.plot = c("CD3D", "SELL", "S100A4", "CD8A", "GNLY", "MS4A1",
                                    "FCGR3A", "HSP90AB1", "CCR7"),
            min.cutoff = "q9",
            cols.use = c('lightgrey', 'blue'))
```




```
new.ident <- c("CD14 Mono", "Memory CD4 T", "Naive CD4 T", "CD8 T", "B", "NK", "CD16 Mono",
               "HS_Stress", "DC")
for (i in 0:8) {
  pbmc <- RenameIdent(pbmc, old.ident.name = i, new.ident.name = new.ident[i + 1])
}
```

We now find all PBMC clusters we found in both datasets in a single integrated analysis, but since we double our cell number we can identify

1. A separation between memory/naive T cells
2. A subpopulation of cells dominated by heat-shock stress in both datasets
3. If you increase resolution higher (i.e. 1.2), you can also see distinct populations of monocytes (with strong type I IFN response), in both datasets

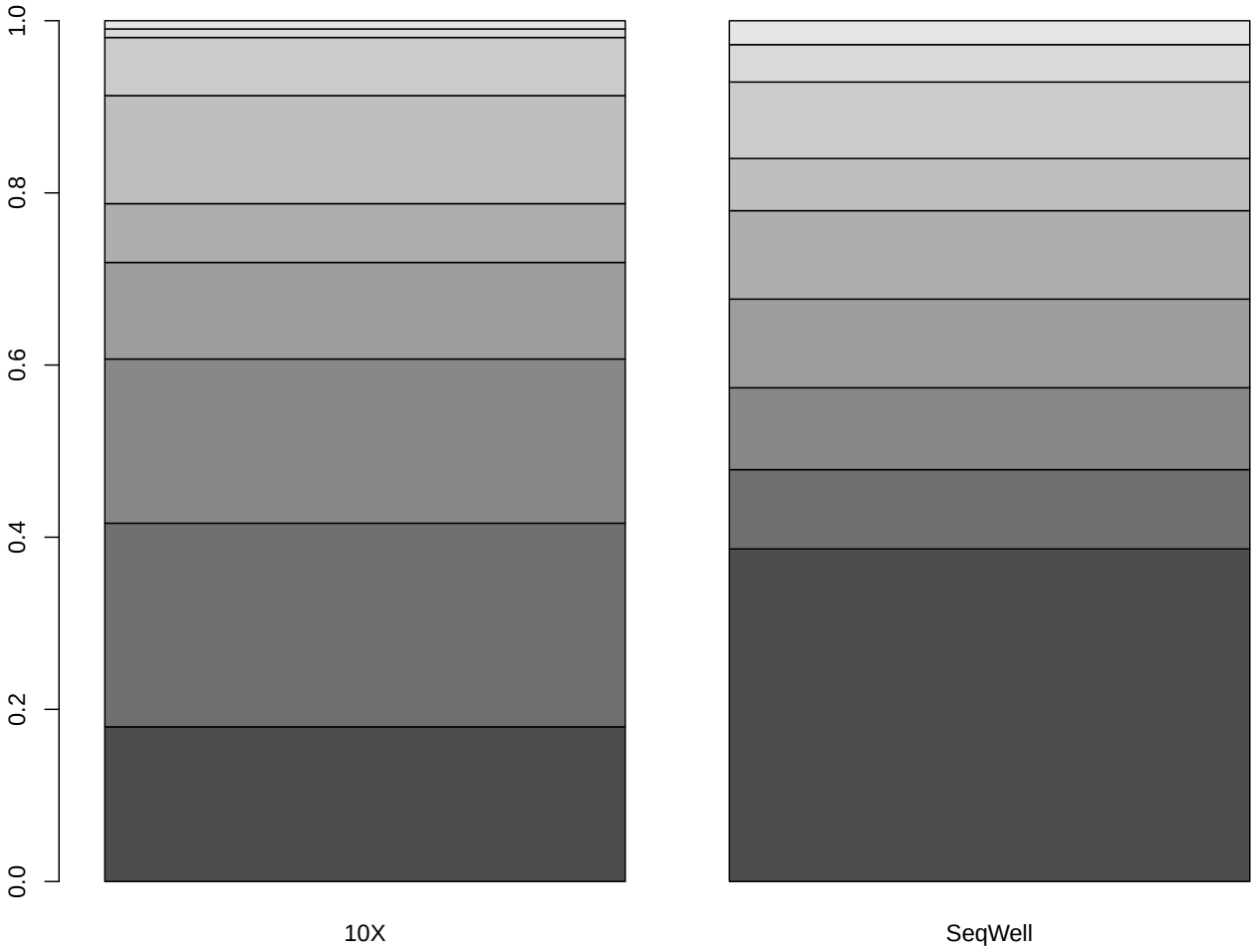
Find markers of memory vs naive cells in *both* datasets

```
mem_vs_naive <- FindConservedMarkers(pbmc, ident.1 = "Memory CD4 T", ident.2 = "Naive CD4 T",
                                     grouping.var = "protocol")
head(mem_vs_naive, 10)
```

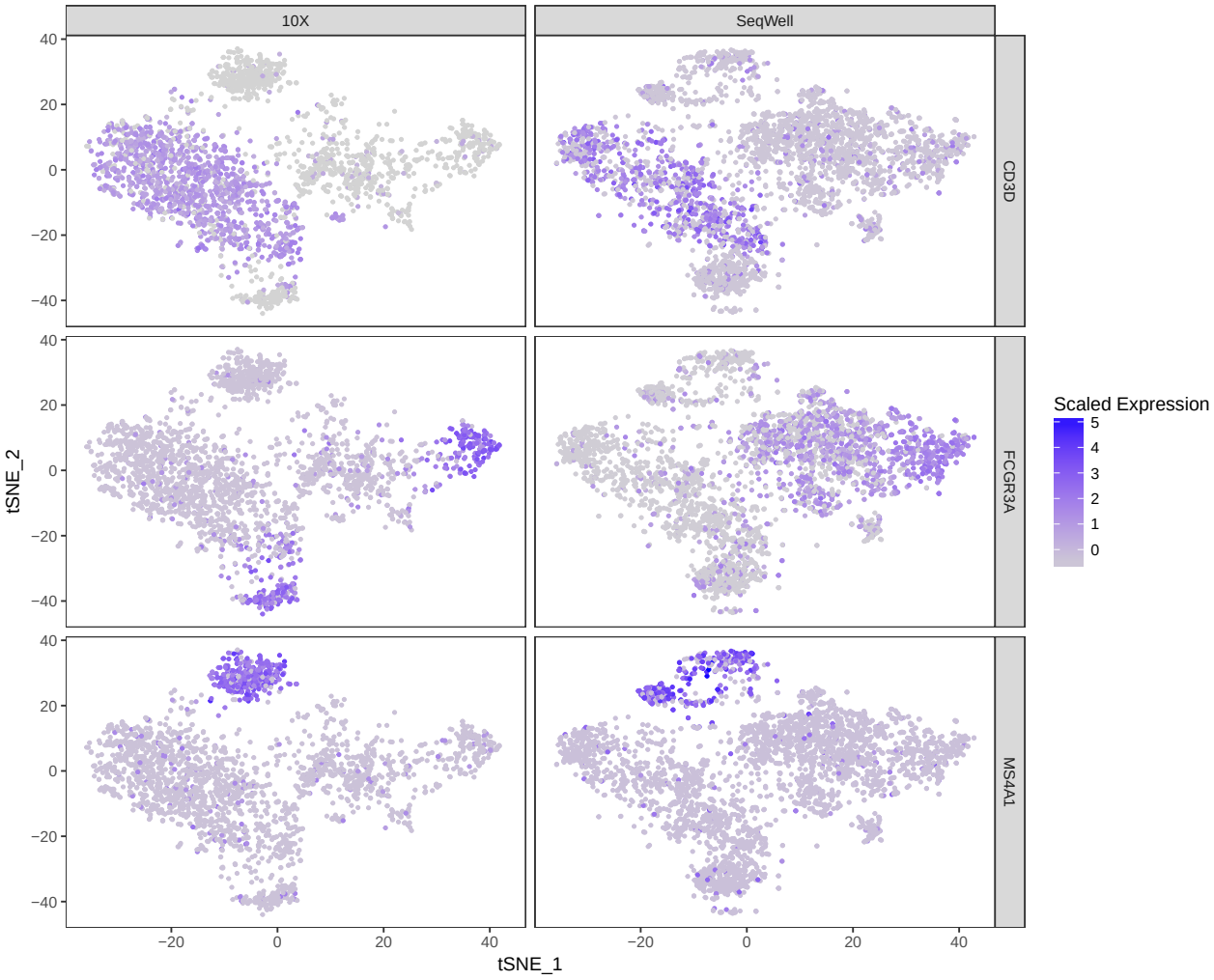
```
##            10X_p_val 10X_avg_diff 10X_pct.1 10X_pct.2 SeqWell_p_val
## S100A4  4.387843e-83    1.1447535     0.918     0.624  2.849128e-05
## B2M     1.208745e-60    0.3393787     1.000     1.000  8.281081e-18
## ANXA1   1.254520e-46    0.8676262     0.769     0.434  5.672705e-11
## S100A11 1.421123e-38    0.8718079     0.584     0.242  6.304153e-03
## ANXA2   5.905940e-36    0.8591081     0.439     0.120  1.695537e-06
## KLF6    5.932322e-31    0.6519894     0.765     0.438  3.910884e-06
## S100A10 5.827922e-30    0.6876870     0.750     0.492  1.970050e-08
## RPS3A   1.320666e-28   -0.2783626     1.000     1.000  1.108054e-09
## VIM     5.364723e-28    0.4897975     0.923     0.812  1.608739e-04
## LGALS1  6.119004e-26    0.8989090     0.377     0.148  1.151152e-05
##         SeqWell_avg_diff SeqWell_pct.1 SeqWell_pct.2     max_pval
## S100A4         0.4355190         0.401         0.257 2.849128e-05
## B2M            0.2642345         1.000         1.000 8.281081e-18
## ANXA1          0.5373080         0.708         0.480 5.672705e-11
## S100A11        0.2552688         0.575         0.503 6.304153e-03
## ANXA2          0.5001743         0.283         0.134 1.695537e-06
## KLF6           0.5004020         0.363         0.194 3.910884e-06
## S100A10        0.4246279         0.785         0.629 1.970050e-08
## RPS3A         -0.3264440         0.938         0.977 1.108054e-09
## VIM            0.2987739         0.826         0.729 1.608739e-04
## LGALS1         0.4793909         0.416         0.257 1.151152e-05
##         fisher_pval
## S100A4            0
## B2M               0
## ANXA1             0
## S100A11           0
## ANXA2             0
## KLF6              0
## S100A10           0
## RPS3A             0
## VIM               0
## LGALS1            0
```

We can also compare proportional shifts in the data. As can be seen in the barplot, the two patients profiled have very different composition

```
freq_table <- prop.table(table(pbmc@ident, pbmc@meta.data[, "protocol"]), 2)
barplot(freq_table)
```




```
freq_table

#A useful plotting functions ( valuable to confirm the alignment makes sense)
FeatureHeatmap(pbmc, c("CD3D", "FCGR3A", "MS4A1"), group.by = "protocol", sep.scale = T,
               pt.size = 0.7, cols.use = c('lightgrey', 'blue'))
```


10X
SeqWell
CD3D
FCGR3A
MS4A1
tSNE_1
tSNE_2
Scaled Expression
5
4
3
2
1
0