 # Poly-A transcriptome analysis of zebrafish embryos (example dataset)

```
bam_files <- read.delim("./bam_files.txt",header = T)
library(Rsubread)
Counts_fc <- featureCounts(bam_files$bam_file,
                        annot.ext = "./Danio_rerio.GRCz10.91.gtf",
                        isGTFAnnotationFile = TRUE, GTF.attrType ="gene_name",
isPairedEnd=TRUE, nthreads = 10)
write.table(Counts_fc$counts, file = "./Counts.txt", sep='\t', quote=F, row.names=T)
data <- read.table(file="./Counts.txt", sep = "\t", header = T, row.names = 1, com='')
col_ordering = c(1,2,4,5)
rnaseqMatrix = data[,col_ordering]
rnaseqMatrix = rnaseqMatrix[rowSums(rnaseqMatrix>2)>=2,]

# create a count matrix Data object from value object
CPMs <- cpm(rnaseqMatrix)
#Remove feature without at least 1 CPM in n samples in >=n, where n is the size of
the smallest group of replicates, 2 in this case
keep = rowSums(CPMs > 1) >= 2
#Keep good quality rows
rnaseqMatrix = rnaseqMatrix[keep,]
conditions = factor(c(rep("Dome_WT", 2), rep("Dome_momix", 2)))
exp_study= DGEList(counts=rnaseqMatrix, group=conditions)

exp_study = estimateCommonDisp(exp_study)
exp_study = estimateTagwiseDisp(exp_study)
exp_study = calcNormFactors(exp_study)
exp_study = estimateCommonDisp(exp_study)
exp_study = estimateTagwiseDisp(exp_study)



#sample correlation
svg("./Dome_momix_sample_correlation.svg")
plotMDS(exp_study, labels=c("Dome_Ctrl_1", "Dome_Ctrl_2",
"Dome_Momix_1","Dome_Momix_2"),
      col=as.numeric(exp_study$samples$group))
dev.off()

#get normalized cpm
norm_CPM= cpm(rnaseqMatrix,normalized.lib.sizes=TRUE)
write.table(norm_CPM, "./normCPM_momix.txt", sep='\t', quote=F, row.names=T)



#get normalized counts for barplots
library("RColorBrewer", lib.loc="/usr/lib/R/site-library")
scale <- exp_study$samples$lib.size*exp_study $samples$norm.factors
normCounts <- round(t(t(rnaseqMatrix)/scale)*mean(scale))
```

```
#boxplot after normalization
nsamples <- ncol(rnaseqMatrix)
col <- brewer.pal(nsamples, "Paired")
svg("./boxplot_momix.svg")
boxplot(log2(normCounts+1), las=3, col=col, cex.axis=0.5)
dev.off()


et_momix = exactTest(exp_study, pair=c("Dome_WT","Dome_momix"))
tTags_momix = topTags(et_momix,n=NULL)

write.table(tTags_momix, file='tTags_momix .edgeR.DE_results', sep='
        ', quote=F, row.names=T)
sig_DE_momix =tTags_momix[tTags_momix$table$PValue<0.05,]
write.table(sig_DE_momix, file='sig_DE_momix', sep='\t', quote=F, row.names=T)
up_sig_DE_momix=sig_DE_momix[sig_DE_momix$table$logFC>0.58,]
#1811
write.table(up_sig_DE_momix, file='up_sig_DE_momix', sep='\t', quote=F,
row.names=T)
down_sig_DE_momix=sig_DE_momix[sig_DE_momix$table$logFC<(-0.58),]
#1231
write.table(down_sig_DE_momix, file='down_sig_DE_momix', sep='\t', quote=F,
row.names=T)
```

 # analysis of 3’ quants data in zebrafish

```
bam_files <- read.delim("./quantseq_bam_files.txt",header = T)
library(Rsubread)
Counts_fc <- featureCounts(bam_files$bam_file,
                           annot.ext = "../genome/Danio_rerio.GRCz10.91.gtf",
                           isGTFAnnotationFile = TRUE, GTF.attrType = "gene_name",
isPairedEnd=TRUE, nthreads = 10)
write.table(Counts_fc$counts, file = "../Counts.txt", sep='\t', quote=F, row.names=T)
data <- read.table(file="../Counts.txt", sep = "\t", header = T, row.names = 1, com='')
col_ordering = 1:52
rnaseqMatrix = data[,col_ordering]
rnaseqMatrix = rnaseqMatrix[rowSums(rnaseqMatrix>2)>=2,]

# create a count matrix Data object from value object
CPMs <- cpm(rnaseqMatrix)
#Remove feature without at least 1 CPM in n samples in >=n, where n is the size of
the smallest group of replicates, 2 in this case
keep = rowSums(CPMs > 1) >= 2
#Keep good quality rows
rnaseqMatrix = rnaseqMatrix[keep,]
sample_labels_list <- read.table("./samples_labels_list.csv", quote="\"",
comment.char="")
conditions = factor(sample_labels_list$V1)
cds = DGEList(counts=rnaseqMatrix, group=conditions)
```

```
exp_study = estimateCommonDisp(cds)
exp_study = estimateTagwiseDisp(exp_study)
exp_study = calcNormFactors(exp_study)
exp_study = estimateCommonDisp(exp_study)
exp_study = estimateTagwiseDisp(exp_study)
et = exactTest(exp_study, pair=c("A80epi_WT","A80epi_MUT"))
tTags = topTags(et,n=NULL)
is.de = decideTestsDGE(et, p.value = 0.1)
summary(is.de)



design = model.matrix(~0+group, data=cds$samples)
colnames(design) = levels(cds$samples$group)

cds = calcNormFactors(cds)

cds_comm_disp = estimateGLMCommonDisp(cds, design, verbose=TRUE)
cds_tag_disp = estimateGLMTagwiseDisp(cds_comm_disp, design)
fit = glmFit(cds_tag_disp, design)

#sample correlation
svg("./sample_correlation.svg")
plotMDS(cds, labels=colnames(data),
      col=as.numeric(cds$samples$group))
dev.off()

#mean-variance
svg("./mean-variance.svg")
meanVarPlot <- plotMeanVar(cds_tag_disp, show.raw.vars=TRUE,
show.tagwise.vars=TRUE,
                  show.binned.common.disp.vars=FALSE,
show.ave.raw.vars=FALSE,
                  NBline = TRUE , nbins = 100 , pch = 16 ,
                  xlab ="Mean Expression (Log10 Scale)" , ylab = "Variance (Log10
Scale)" ,
                  main = "Mean-Variance Plot" )
dev.off()

#plot BCV
svg("./BCV.svg")
plotBCV(cds_tag_disp)
dev.off()

#get normalized counts
library("RColorBrewer", lib.loc="/usr/lib/R/site-library")
scale <- cds$samples$lib.size*cds$samples$norm.factors
normCounts <- round(t(t(rnaseqMatrix)/scale)*mean(scale))

#boxplot after normalization
```

```
nsamples <- ncol(rnaseqMatrix)
col <- brewer.pal(nsamples, "Paired")
svg("./boxplot.svg")
boxplot(log2(normCounts+1), las=3, col=col, cex.axis=0.5)
dev.off()

my.contrasts <- makeContrasts(A80epi_WTvsA80epi_MUT=A80epi_WT-
A80epi_MUT,
                        A80epi_WTvsA80epi_SATB2_MO=A80epi_WT-
A80epi_SATB2_MO,
                        A6som_WTvsA6som_SATB2_MUT=A6som_WT-
A6som_SATB2_MUT,
                        A14som_WTvsA14som_SATB2_MUT=A14som_WT-
A14som_SATB2_MUT,
                        levels=design)
lrt=lapply(seq(my.contrasts[1,]), function(i) glmLRT(fit, contrast=my.contrasts[,i]))
toptags=lapply(seq(length(lrt)), function(j) topTags(lrt[[j]],n=NULL))
DE_P0.01=lapply(seq(length(toptags)),function(k) subset(toptags[[k]]$table,FDR <=
0.01))
DE_P0.01_UP_DOWN=lapply(seq(length(DE_P0.01)),function(m)
subset(DE_P0.01[[m]], logFC >= 0.58|logFC <= -0.58))
rowname_lists=lapply(seq(length(DE_P0.01_UP_DOWN)),function(n)
rownames(DE_P0.01_UP_DOWN[[n]]))
sig_DE_rownames=unique(unlist(rowname_lists))

#get normalized cpm
norm_CPM= cpm(cds_tag_disp,normalized.lib.sizes=TRUE)
write.table(norm_CPM, "./normCPM.txt", sep='\t', quote=F, row.names=T)
sig_DE_normCPM=norm_CPM[row.names(norm_CPM) %in% sig_DE_rownames,]
write.table(sig_DE_normCPM, "./sig_DE_normCPM.txt", sep='\t', quote=F,
row.names=T)
```

 # ChIPseq data processing example dataset

```
open(R,"Trimmed_samples.txt") || die ("File not found");

# for obtaining aligned bam files

while ($line=<R>)
{
        chomp($line);

        print "$line\n";

        $fwd=$line."_R1.fastq.gz";
        $rev=$line."_R2.fastq.gz";
        $sam=$line.".sam";
        $bam=$line.".bam";
```

```
	$sor_bam=$line."_sorted.bam";
	$sor1=$sor_bam."";
	$bai=$line."_sorted.bam.bai";

	print "Running BWA for $line\n";

	`bwa mem -t 8 /media/iiser/SGHD9_saurabh/danRer10/danRer10.fa /media/
iiser/SGHD9_saurabh/zebrafish_Sox10_sorted_ChIPseq/fastq/Sox10_ChIPseq/
Trimmed/$fwd /media/iiser/SGHD9_saurabh/zebrafish_Sox10_sorted_ChIPseq/
fastq/Sox10_ChIPseq/Trimmed/$rev > /media/iiser/SGHD9_saurabh/
zebrafish_Sox10_sorted_ChIPseq/fastq/Sox10_ChIPseq/Trimmed/$sam`;

	print "Converting sam to bam for $line\n";

	`samtools view -S -b $sam > $bam`;

	print "removing sam file -- post sam to bam conversion\n";

	`rm $sam`;

	print "Converting bam to sorted bam for $line\n";

	`samtools sort $bam -o $sor_bam`;

	print "removing bam file -- post sorting of bam conversion\n";

	`rm $bam`;

	print "Converting sorted bam to indexed bam for $line\n";

	`samtools index $sor1 > $bai`;

}


# For subsampling using bbmap

open(R,"bam_details.txt") ll die ("File not found");

while ($line=<R>)
{
	chomp($line);

	print "$line\n";

	$bam=$line.".bam";  #E12.5_cortex_Input_Rep1.bam
	$mapped_bam=$line."_mapped.bam";
	$subsampled_10M=$line."subsample_10M.bam";

	print "Converting bam to mapped for $line\n";
```

```
	`reformat.sh in=$bam out=$mapped_bam mappedonly=t`;


	print "Converting bam to sorted bam for $line\n";

	`reformat.sh in=$mapped_bam out=$subsampled_10M mappedonly=t
samplereadstarget=10000000`;

	print "removing bam file -- post sorting of bam conversion\n";

	`rm $mapped_bam`;

}

# for MACS peak calling

macs2 callpeak -t 512-SATB2-BR3_clean_sortedsubsample_40M.bed -c 512-Input-
BR2_clean_sortedsubsample_40M.bed -f BED -q 0.05 --name 512-SATB2-
BR3_MACS_0.05;

# for generating bigWig files using Deeptools

bamCoverage --normalizeUsing RPKM -p 8 --binSize 10 --smoothLength 60 --
extendReads 200 -b Sox10_Negative_input_clean_sorted.bam -o
Sox10_Negative_input_clean_sorted.bw;


# for generating subtracted bigWig files

bigwigCompare --binSize=10 --numberOfProcessors 6 --
outFileName=Sox10_Negative_Satb2_Rep3_10M_subtract.bw --
outFileFormat=bigwig --bigwig1
Sox10_Negative_Satb2_Rep3_clean_sortedsubsample_10M.bam.bw --bigwig2
Sox10_Negative_input_clean_sortedsubsample_10M.bam.bw --pseudocount=1 --
operation subtract --skipNonCoveredRegions;
```

 # ATAC-seq data analysis (example dataset)

```
while ($line=<R>)
{
	chomp($line);

	print "$line\n";

	$fwd=$line."_R1.fastq.gz";
	$rev=$line."_R2.fastq.gz";
	$sam=$line.".sam";
	$bam=$line.".bam";   #E12.5_cortex_Input_Rep1.bam
	$sor_bam=$line."_sorted";
```

```
	$sor1=$sor_bam."";  #E12.5_cortex_Input_Rep1_sorted.bam
	$bai=$line."_sorted.bam.bai";

	print "Running BWA for $line\n";

	`bwa mem -t 8 /media/gastrula/SGHD9_saurabh/danRer10/danRer10.fa /
media/gastrula/SGHD9_saurabh/zebrafish_SATB2_ATAC/Trimmed/$fwd /media/
gastrula/SGHD9_saurabh/zebrafish_SATB2_ATAC/Trimmed/$rev > /media/gastrula/
SGHD9_saurabh/zebrafish_SATB2_ATAC/Trimmed/$sam`;

	print "Converting sam to bam for $line\n";

	`samtools view -S -b $sam > $bam`;

	print "removing sam file -- post sam to bam conversion\n";

	`rm $sam`;

	print "Converting bam to sorted bam for $line\n";

	`samtools sort $bam $sor_bam`;

	print "removing bam file -- post sorting of bam conversion\n";

	`rm $bam`;

	print "Converting sorted bam to indexed bam for $line\n";

	`samtools index $sor1 > $bai`;

}

# fileter mitochondrial reads

samtools idxstats SATB2mut_Som_14_rep2_clean_sorted.bam | cut -f 1 | grep -v
chrM | xargs samtools view -b SATB2mut_Som_14_rep2_clean_sorted.bam >
SATB2mut_Som_14_rep2_filtered.bam;
```

 # scRNAseq analysis of mosaically overexpressed FLAG-Satb2 at 16 cell stage of zebrafish embryos at dome

```
library(dplyr)
library(Seurat)
setwd("/home//FLAG_SATB2_scRNAseq_analysis/FLAG_SATB2_8k")
Satb2.data <- Read10X(data.dir = "/home/iiser/saurabh/
FLAG_SATB2_scRNAseq_analysis/FLAG_SATB2_8k/filtered_feature_bc_matrix/")
Satb2 <- CreateSeuratObject(counts = Satb2.data, project = "Satb2", min.cells = 3,
min.features = 200)
```

```
Satb2
Satb2[["percent.mt"]] <- PercentageFeatureSet(Satb2, pattern = "^MT-")
VlnPlot(Satb2, features = c("nFeature_RNA", "nCount_RNA", "percent.mt"), ncol = 3)
Satb2 <- subset(Satb2, subset = nFeature_RNA > 200 & nFeature_RNA < 2000 &
percent.mt < 5)
Satb2 <- NormalizeData(Satb2, normalization.method = "LogNormalize", scale.factor
= 10000)
Satb2 <- NormalizeData(Satb2)
Satb2 <- FindVariableFeatures(Satb2, selection.method = "vst", nfeatures = 2000)
top20 <- head(VariableFeatures(Satb2), 20)
plot1 <- VariableFeaturePlot(Satb2)
plot2 <- LabelPoints(plot = plot1, points = top20, repel = TRUE)
plot1 + plot2
all.genes <- rownames(Satb2)
Satb2 <- ScaleData(Satb2, features = all.genes)
Satb2 <- RunPCA(Satb2, features = VariableFeatures(object = Satb2))
print(Satb2[["pca"]], dims = 1:5, nfeatures = 20)
VizDimLoadings(Satb2, dims = 4:5, reduction = "pca")
DimPlot(Satb2, reduction = "pca")
DimHeatmap(Satb2, dims = 10, cells = 500, balanced = TRUE)
DimHeatmap(Satb2, dims = 1:15, cells = 500, balanced = TRUE)
Satb2 <- JackStraw(Satb2, num.replicate = 100)
Satb2 <- ScoreJackStraw(Satb2, dims = 1:20)
JackStrawPlot(Satb2, dims = 1:15)
ElbowPlot(Satb2)
Satb2 <- FindNeighbors(Satb2, dims = 1:20)
Satb2 <- FindClusters(Satb2, resolution = 0.5)
head(Idents(Satb2), 5)
Satb2 <- RunUMAP(Satb2, dims = 1:20)
DimPlot(Satb2, reduction = "umap")
saveRDS(Satb2, file = "Flag_Satb2_v2.rds")
cluster1.markers <- FindMarkers(Satb2, ident.1 = 1, min.pct = 0.25)
head(cluster1.markers, n = 20)
cluster2.markers <- FindMarkers(Satb2, ident.1 = 2, ident.2 = c(0, 3), min.pct = 0.25)
head(cluster2.markers, n = 50)
Satb2.markers <- FindAllMarkers(Satb2, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)
Satb2.markers %>% group_by(cluster) %>% top_n(n = 2, wt = avg_logFC)
cluster1.markers <- FindMarkers(Satb2, ident.1 = 0, logfc.threshold = 0.25, test.use
= "roc", only.pos = TRUE)
VlnPlot(Satb2, features = c("satb2", "bmp4", "pou5f3", "fgfr4", "foxd5"))
FeaturePlot(Satb2, features = c("satb2", "chd7"))

#to re read rds file
Satb2 <- readRDS(file = "Flag_Satb2_v2.rds")

next_set_cluster5
Satb2 <- FindNeighbors(Satb2, dims = 1:5)
Satb2 <- FindClusters(Satb2, resolution = 0.5)
head(Idents(Satb2), 5)
```

```
Satb2 <- RunUMAP(Satb2, dims = 1:5)
DimPlot(Satb2, reduction = "umap")
saveRDS(Satb2, file = "Flag_Satb2.rds")
cluster1.markers <- FindMarkers(Satb2, ident.1 = 1, min.pct = 0.25)
head(cluster1.markers, n = 10)
cluster5.markers <- FindMarkers(Satb2, ident.1 = 5, ident.2 = c(0, 3), min.pct = 0.25)
head(cluster5.markers, n = 5)
Satb2.markers <- FindAllMarkers(Satb2, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)
Satb2.markers %>% group_by(cluster) %>% top_n(n = 2, wt = avg_logFC)
cluster1.markers <- FindMarkers(Satb2, ident.1 = 0, logfc.threshold = 0.25, test.use
= "roc", only.pos = TRUE)
VlnPlot(Satb2, features = c("satb2", "sox3", "pou5f3", "sox19b", "nanog", "foxd5"))
FeaturePlot(Satb2, features = c("satb2", "sox3", "pou5f3", "gsc", "foxd5",
"noto","nanog", "sox19b"))

next_set_cluster3
Satb2 <- FindNeighbors(Satb2, dims = 1:3)
Satb2 <- FindClusters(Satb2, resolution = 0.5)
head(Idents(Satb2), 5)
Satb2 <- RunUMAP(Satb2, dims = 1:3)
DimPlot(Satb2, reduction = "umap")
saveRDS(Satb2, file = "Flag_Satb2.rds")
cluster1.markers <- FindMarkers(Satb2, ident.1 = 1, min.pct = 0.25)
head(cluster1.markers, n = 10)
cluster5.markers <- FindMarkers(Satb2, ident.1 = 5, ident.2 = c(0, 3), min.pct = 0.25)
head(cluster5.markers, n = 5)
Satb2.markers <- FindAllMarkers(Satb2, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)
Satb2.markers %>% group_by(cluster) %>% top_n(n = 2, wt = avg_logFC)
cluster1.markers <- FindMarkers(Satb2, ident.1 = 0, logfc.threshold = 0.25, test.use
= "roc", only.pos = TRUE)
VlnPlot(Satb2, features = c("satb2", "sox3", "pou5f3", "sox19b", "nanog", "foxd5"))
FeaturePlot(Satb2, features = c("satb2", "sox3", "pou5f3", "gsc", "foxd5",
"noto","nanog", "sox19b"))


next_cluster_20
Satb2 <- FindNeighbors(Satb2, dims = 1:20)
Satb2 <- FindClusters(Satb2, resolution = 0.5)
head(Idents(Satb2), 5)
Satb2 <- RunUMAP(Satb2, dims = 1:20)
DimPlot(Satb2, reduction = "umap")
saveRDS(Satb2, file = "Flag_Satb2.rds")
cluster1.markers <- FindMarkers(Satb2, ident.1 = 1, min.pct = 0.25)
head(cluster1.markers, n = 10)
cluster5.markers <- FindMarkers(Satb2, ident.1 = 5, ident.2 = c(0, 3), min.pct = 0.25)
head(cluster5.markers, n = 5)
Satb2.markers <- FindAllMarkers(Satb2, only.pos = TRUE, min.pct = 0.25,
logfc.threshold = 0.25)
```

```
Satb2.markers %>% group_by(cluster) %>% top_n(n = 2, wt = avg_logFC)
cluster1.markers <- FindMarkers(Satb2, ident.1 = 0, logfc.threshold = 0.25, test.use
= "roc", only.pos = TRUE)
VlnPlot(Satb2, features = c("satb2", "sox3", "pou5f3", "sox19b", "nanog", "foxd5"))
FeaturePlot(Satb2, features = c("satb2", "mixl1"))
library(patchwork)
FeaturePlot(Satb2, c("satb2", "mixl1")) &
  scale_colour_gradientn(colours = rev(brewer.pal(n = 11, name = "RdBu")))


FeaturePlot(object = Satb2, features = c("satb2", "dkk1b"), reduction = 'umap',
pt.size = 1, blend = T, cols = c("grey", "red", "blue"), min.cutoff = 0, max.cutoff = 5)
levels(Satb2)
```

 # scRNAseq analysis of will-type and Satb2 mutant embryos at 14 somites

```
library(dplyr)
library(Seurat)
library(patchwork)


MUT_14ss.data <- Read10X(data.dir = "/home/iiser/saurabh_lamark/
SATB2_zebrafish/Satb2_MUT_14ss_scRNAseq/Satb2_MUT_14som_Rep2/outs/
filtered_feature_bc_matrix/")

MUT_14ss <- CreateSeuratObject(counts = MUT_14ss.data, project = "MUT_14ss",
min.cells = 3, min.features = 200)

MUT_14ss


WT_14ss.data <- Read10X(data.dir = "/home/iiser/saurabh_lamark/
SATB2_zebrafish/Satb2_MUT_14ss_scRNAseq/WT_Satb2_14som_Rep1/outs/
filtered_feature_bc_matrix/")

WT_14ss <- CreateSeuratObject(counts = WT_14ss.data, project = "WT_14ss",
min.cells = 3, min.features = 200)

WT_14ss


MUT_14ss <- NormalizeData(MUT_14ss)
WT_14ss <- NormalizeData(WT_14ss)

WT_MUT_normalized <- merge(WT_14ss, y = MUT_14ss, add.cell.ids = c("WT",
"MUT"), project = "Satb2_MUT",
                           merge.data = TRUE)
```

```
GetAssayData(WT_MUT_normalized)[1:10, 1:15]

Satb2_MUT.list <- SplitObject(WT_MUT_normalized, split.by = "orig.ident")

Satb2_MUT.list <- lapply(X = Satb2_MUT.list, FUN = function(x) {
  x <- NormalizeData(x)
  x <- FindVariableFeatures(x, selection.method = "vst", nfeatures = 2000)
})
features <- SelectIntegrationFeatures(object.list = Satb2_MUT.list)

Satb2_MUT.anchors <- FindIntegrationAnchors(object.list = Satb2_MUT.list,
anchor.features = features)
Satb2_MUT.combined <- IntegrateData(anchorset = Satb2_MUT.anchors)

DefaultAssay(Satb2_MUT.combined) <- "integrated"

Satb2_MUT.combined <- ScaleData(Satb2_MUT.combined, verbose = FALSE)


Satb2_MUT.combined <- RunPCA(Satb2_MUT.combined, npcs = 15, verbose =
FALSE)


Satb2_MUT.combined <- RunUMAP(Satb2_MUT.combined, reduction = "pca", dims
= 1:15)


Satb2_MUT.combined <- FindNeighbors(Satb2_MUT.combined, reduction = "pca",
dims = 1:15)

Satb2_MUT.combined <- FindClusters(Satb2_MUT.combined, resolution = 0.50)

p1 <- DimPlot(Satb2_MUT.combined, reduction = "umap", group.by = "orig.ident")

p2 <- DimPlot(Satb2_MUT.combined, reduction = "umap", label = TRUE, repel =
TRUE)
p1 + p2
DimPlot(Satb2_MUT.combined, reduction = "umap", split.by = "orig.ident")

DefaultAssay(Satb2_MUT.combined) <- "RNA"

nk.markers <- FindConservedMarkers(Satb2_MUT.combined, ident.1 = 6,
grouping.var = "orig.ident", verbose = FALSE)

head(nk.markers)

FeaturePlot(Satb2_MUT.combined, features = c("crestin", "sox10", "sox19a", "tbx16",
"sox2", "sox32", "tyr",
                                             "etv2", "satb2", "myod1", "cebpb", "sox7", "epcam",
"foxd3", "myt1a", "zic1"), min.cutoff = "q9")
```

```
#sox9b, zic1, epcam, satb2, etv2, myt1a, sox9a, sox10, msgn1, krt8, id3

FeaturePlot(Satb2_MUT.combined, features = c("epcam"), split.by = "orig.ident",
max.cutoff = 0.5,
        cols = c("grey", "red"))

cluster0.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 0, min.pct = 0.25)
head(cluster0.markers, n = 20)

cluster1.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 1, min.pct = 0.25)
head(cluster1.markers, n = 20)

cluster2.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 2, min.pct = 0.25)
head(cluster2.markers, n = 20)

cluster3.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 3, min.pct = 0.25)
head(cluster3.markers, n = 20)

cluster4.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 4, min.pct = 0.25)
head(cluster4.markers, n = 20)

cluster5.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 5, min.pct = 0.25)
head(cluster5.markers, n = 20)

cluster6.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 6, min.pct = 0.25)
head(cluster6.markers, n = 20)

cluster7.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 7, min.pct = 0.25)
head(cluster7.markers, n = 20)

cluster8.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 8, min.pct = 0.25)
head(cluster8.markers, n = 20)

cluster9.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 9, min.pct = 0.25)
head(cluster9.markers, n = 20)

cluster10.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 10, min.pct =
0.25)
head(cluster10.markers, n = 20)

cluster11.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 11, min.pct =
0.25)
head(cluster11.markers, n = 20)

cluster12.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 12, min.pct =
0.25)
head(cluster12.markers, n = 20)

cluster13.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 13, min.pct =
```

```
0.25)
head(cluster13.markers, n = 20)

cluster14.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 14, min.pct =
0.25)
head(cluster14.markers, n = 20)

cluster15.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 15, min.pct =
0.25)
head(cluster15.markers, n = 20)

cluster16.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 16, min.pct =
0.25)
head(cluster16.markers, n = 20)

cluster17.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 17, min.pct =
0.25)
head(cluster17.markers, n = 20)

cluster18.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 18, min.pct =
0.25)
head(cluster18.markers, n = 20)

cluster19.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 19, min.pct =
0.25)
head(cluster19.markers, n = 20)

cluster20.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 20, min.pct =
0.25)
head(cluster20.markers, n = 20)

cluster21.markers <- FindMarkers(Satb2_MUT.combined, ident.1 = 21, min.pct =
0.25)
head(cluster21.markers, n = 20)

saveRDS(pbmc, file = "../output/pbmc_tutorial.rds")


saveRDS(Satb2_MUT.combined, file = "Satb2_mut.rds")
```