# SugenFvsM

2024-02-13

## Load the Required Library

```
library(Seurat)
```

```
## Loading required package: SeuratObject
```

```
## Loading required package: sp
```

```
##
## Attaching package: 'SeuratObject'
```

```
## The following object is masked from 'package:base':
##
##     intersect
```

```
library(ggplot2)
library(cowplot)
library(gridExtra)
```

## LIST ALL H5 FILTERED_FEATURE_BC_MATRIX

```
Sugen_Filtered <- list.files(path = "/Sugen_Ec_Filtered", pattern = "//.h5$", full.names = TRUE)
```

## LOAD EACH of THE Filtered_Bc_Matrixes

```
paste(getwd())

WTECF1_21_1 <- Read10X_h5(paste(getwd(), "Sugen_Ec_Filtered", "WTECF1_21_filtered_feature_bc_matrix.h5"
WTECF2_21_2 <- Read10X_h5(paste(getwd(), "Sugen_Ec_Filtered", "WTECF2_21_filtered_feature_bc_matrix.h5"
SUECF1_22_3 <- Read10X_h5(paste(getwd(), "Sugen_Ec_Filtered", "SUECF1_22_filtered_feature_bc_matrix.h5"
SUECF2_22_4 <- Read10X_h5(paste(getwd(), "Sugen_Ec_Filtered", "SUECF2_22_filtered_feature_bc_matrix.h5"
WTECM1_27_5 <- Read10X_h5(paste(getwd(), "Sugen_Ec_Filtered", "WTECM1_27_filtered_feature_bc_matrix.h5"
WTECM2_27_6 <- Read10X_h5(paste(getwd(), "Sugen_Ec_Filtered", "WTECM2_27_filtered_feature_bc_matrix.h5"
SUECM1_28_7 <- Read10X_h5(paste(getwd(), "Sugen_Ec_Filtered", "SUECM1_28_filtered_feature_bc_matrix.h5"
SUECM2_28_8 <- Read10X_h5(paste(getwd(), "Sugen_Ec_Filtered", "SUECM2_28_filtered_feature_bc_matrix.h5"
```

```
WTECF1_21_1 <- CreateSeuratObject(WTECF1_21_1, min.cells=3, min.features = 200)
WTECF2_21_2 <- CreateSeuratObject(WTECF2_21_2, min.cells=3, min.features = 200)
SUECF1_22_3 <- CreateSeuratObject(SUECF1_22_3, min.cells=3, min.features = 200)
SUECF2_22_4 <- CreateSeuratObject(SUECF2_22_4, min.cells=3, min.features = 200)
WTECM1_27_5 <- CreateSeuratObject(WTECM1_27_5, min.cells=3, min.features = 200)
WTECM2_27_6 <- CreateSeuratObject(WTECM2_27_6, min.cells=3, min.features = 200)
SUECM1_28_7 <- CreateSeuratObject(SUECM1_28_7, min.cells=3, min.features = 200)
SUECM2_28_8 <- CreateSeuratObject(SUECM2_28_8, min.cells=3, min.features = 200)
```

## MAKE A LIST FROM WHOLE SEURAT OBJECTS

```
sugrat_list <- list(
  WTECF1_21_1 = WTECF1_21_1,
  WTECF2_21_2 = WTECF2_21_2,
  SUECF1_22_3 = SUECF1_22_3,
  SUECF2_22_4 = SUECF2_22_4,
  WTECM1_27_5 = WTECM1_27_5,
  WTECM2_27_6 = WTECM2_27_6,
  SUECM1_28_7 = SUECM1_28_7,
  SUECM2_28_8 = SUECM2_28_8
)
```

## SAVE ALL 8 OBJECTS

```
saveRDS(sugrat_list, "sugrat_list.Rds")
```

## LOAD SUGRATLIST

```
sugrat_list <- readRDS("sugrat_list.Rds")
```

## LOADING SEURATS FROM SEURAT LIST

```
  WTECF1_21_1 <- sugrat_list[["WTECF1_21_1"]]
  WTECF2_21_2 <- sugrat_list[["WTECF2_21_2"]]
  SUECF1_22_3 <- sugrat_list[["SUECF1_22_3"]]
  SUECF2_22_4 <- sugrat_list[["SUECF2_22_4"]]
  WTECM1_27_5 <- sugrat_list[["WTECM1_27_5"]]
  WTECM2_27_6 <- sugrat_list[["WTECM2_27_6"]]
  SUECM1_28_7 <- sugrat_list[["SUECM1_28_7"]]
  SUECM2_28_8 <- sugrat_list[["SUECM2_28_8"]]

rm(sugrat_list)
```

## FIRST SET OF GROUPS

```
A_WT_F <- WTECF1_21_1
B_SU_F <- SUECF1_22_3
C_WT_M <- WTECM1_27_5
D_SU_M <- SUECM1_28_7

rm(WTECF1_21_1, SUECF1_22_3, WTECM1_27_5, SUECM1_28_7)
```

## SECOND SET OF GROUPS

```
A_WT_F_2 <- WTECF2_21_2
B_SU_F_2 <- SUECF2_22_4
C_WT_M_2 <- WTECM2_27_6
D_SU_M_2 <- SUECM2_28_8

rm(WTECF2_21_2, SUECF2_22_4, WTECM2_27_6, SUECM2_28_8)
```

## MAKING A SEURAT LIST FROM ALL THOSE SEURATS

```
seurat_list <- list(
  A_WT_F = A_WT_F,
  B_SU_F = B_SU_F,
  C_WT_M = C_WT_M,
  D_SU_M = D_SU_M,
  A_WT_F_2 = A_WT_F_2,
  B_SU_F_2 = B_SU_F_2,
  C_WT_M_2 = C_WT_M_2,
  D_SU_M_2 = D_SU_M_2
)
```

## Check if mitochondrial gene s are there

```
# Check if RNA_Seq assay is available

 mt_genes <- rownames(A_WT_F)[grep("Mt-", rownames(A_WT_F))]

print(mt_genes)
```

```
##  [1] "Mt-nd1"  "Mt-nd2"  "Mt-co1"  "Mt-co2"  "Mt-atp8" "Mt-atp6" "Mt-co3"
##  [8] "Mt-nd3"  "Mt-nd4l" "Mt-nd4"  "Mt-nd5"  "Mt-nd6"  "Mt-cyb"
```

### Make Percentage of MT Function

```
seurat_list <- lapply(seurat_list, function(seurat_obj) {
  seurat_obj[["percent.mt"]] <- PercentageFeatureSet(seurat_obj, pattern = "Mt-")
  return(seurat_obj)
})
```

## LOADING FUNCTIONS BACK TO SEPARATE SEURATS AFTER LAPPLY

```
  A_WT_F <- seurat_list[["A_WT_F"]]
  B_SU_F <- seurat_list[["B_SU_F"]]
  C_WT_M <- seurat_list[["C_WT_M"]]
  D_SU_M <- seurat_list[["D_SU_M"]]
  A_WT_F_2 <- seurat_list[["A_WT_F_2"]]
  B_SU_F_2 <- seurat_list[["B_SU_F_2"]]
  C_WT_M_2 <- seurat_list[["C_WT_M_2"]]
  D_SU_M_2 <- seurat_list[["D_SU_M_2"]]
```

### Separate each seurat objects from seurat list

```
  A_WT_F <- sugen_list1[["A_WT_F"]]
  B_SU_F <- sugen_list1[["B_SU_F"]]
  C_WT_M <- sugen_list1[["C_WT_M"]]
  D_SU_M <- sugen_list1[["D_SU_M"]]
  A_WT_F_2 <- sugen_list2[["A_WT_F_2"]]
  B_SU_F_2 <- sugen_list2[["B_SU_F_2"]]
  C_WT_M_2 <- sugen_list2[["C_WT_M_2"]]
  D_SU_M_2 <- sugen_list2[["D_SU_M_2"]]
```

### Make 2 separate seurat list

```
sugen_list1 <- list(
  A_WT_F = A_WT_F,
  B_SU_F = B_SU_F,
  C_WT_M = C_WT_M,
  D_SU_M = D_SU_M
  )


sugen_list2 <- list(
  A_WT_F_2 = A_WT_F_2,
  B_SU_F_2 = B_SU_F_2,
  C_WT_M_2 = C_WT_M_2,
  D_SU_M_2 = D_SU_M_2
)
```

## Plot one by one before cleaning -nCOUNTRNA

```
library(gridExtra)

p1 <- VlnPlot(A_WT_F, features = "nCount_RNA") + geom_hline(yintercept = c(2000, 25000), linetype = "da
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
## Warning: Using ‘size‘ aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use ‘linewidth‘ instead.
## This warning is displayed once every 8 hours.
## Call ‘lifecycle::last_lifecycle_warnings()‘ to see where this warning was
## generated.
```

```
p2 <- VlnPlot(B_SU_F, features = "nCount_RNA") + geom_hline(yintercept = c(2000, 25000), linetype = "da
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

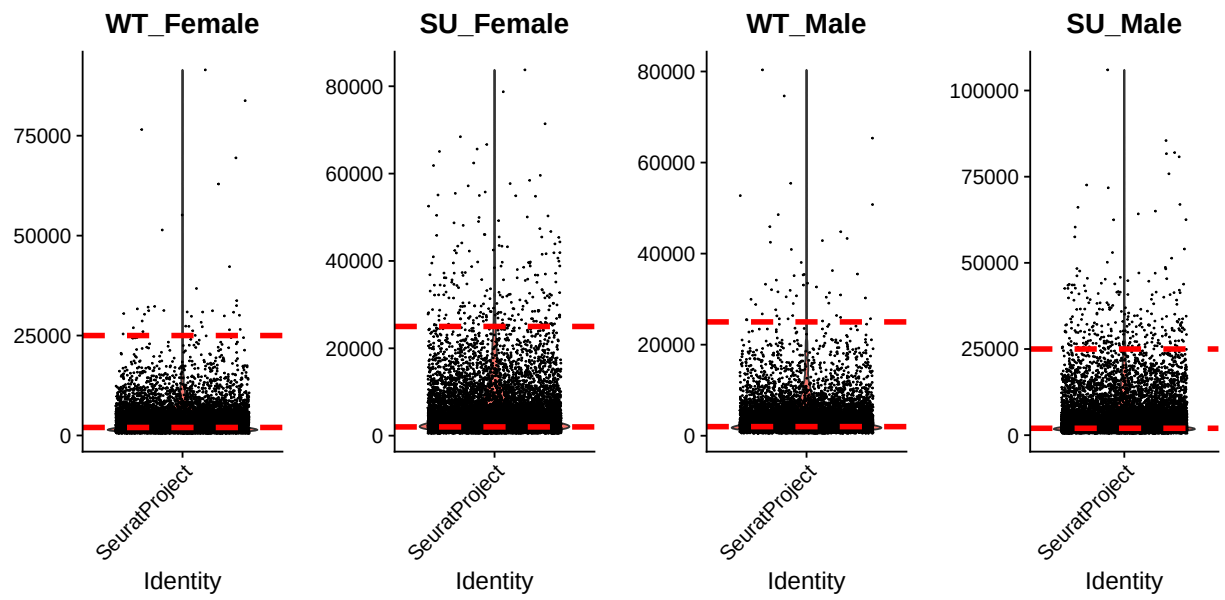
```
p3 <- VlnPlot(C_WT_M, features = "nCount_RNA") + geom_hline(yintercept = c(2000, 25000), linetype = "da
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
p4 <- VlnPlot(D_SU_M, features = "nCount_RNA") + geom_hline(yintercept = c(2000, 25000), linetype = "da
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
#grid.arrange(p1, p2, p3, p4, ncol = 2)
#p <- p1 + p2 + p3 + p4
plot_row <- plot_grid(p1, p2, p3, p4, ncol = 4, align = "h")


print(plot_row)
```




# NFEATURE

```
library(gridExtra)

p1 <- VlnPlot(A_WT_F, features = "nFeature_RNA") + geom_hline(yintercept = c(6000, 200), linetype = "da
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
p2 <- VlnPlot(B_SU_F, features = "nFeature_RNA") + geom_hline(yintercept = c(6000, 200), linetype = "da
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
p3 <- VlnPlot(C_WT_M, features = "nFeature_RNA") + geom_hline(yintercept = c(6000, 200), linetype = "da
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
p4 <- VlnPlot(D_SU_M, features = "nFeature_RNA") + geom_hline(yintercept = c(6000, 200), linetype = "da
```

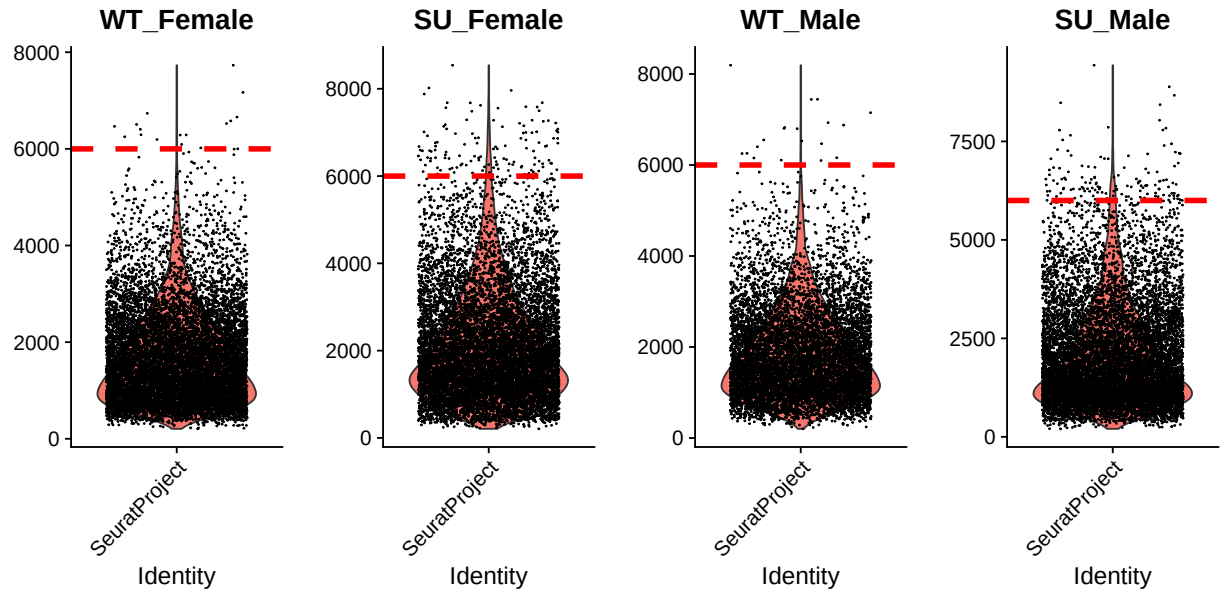
```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
#grid.arrange(p1, p2, p3, p4, ncol = 2)
#p <- p1 + p2 + p3 + p4
plot_row <- plot_grid(p1, p2, p3, p4, ncol = 4, align = "h")
```

```
## Warning: Removed 1 rows containing missing values (`geom_hline()`).
```

```
## Warning: Removed 1 rows containing missing values (`geom_hline()`).
## Removed 1 rows containing missing values (`geom_hline()`).
## Removed 1 rows containing missing values (`geom_hline()`).
```

```
print(plot_row)
```




# PERCENT.MT

```
library(gridExtra)

p1 <- VlnPlot(A_WT_F, features = "percent.mt") + geom_hline(yintercept = 15, linetype = "dashed", color
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
p2 <- VlnPlot(B_SU_F, features = "percent.mt") + geom_hline(yintercept = 15, linetype = "dashed", color
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

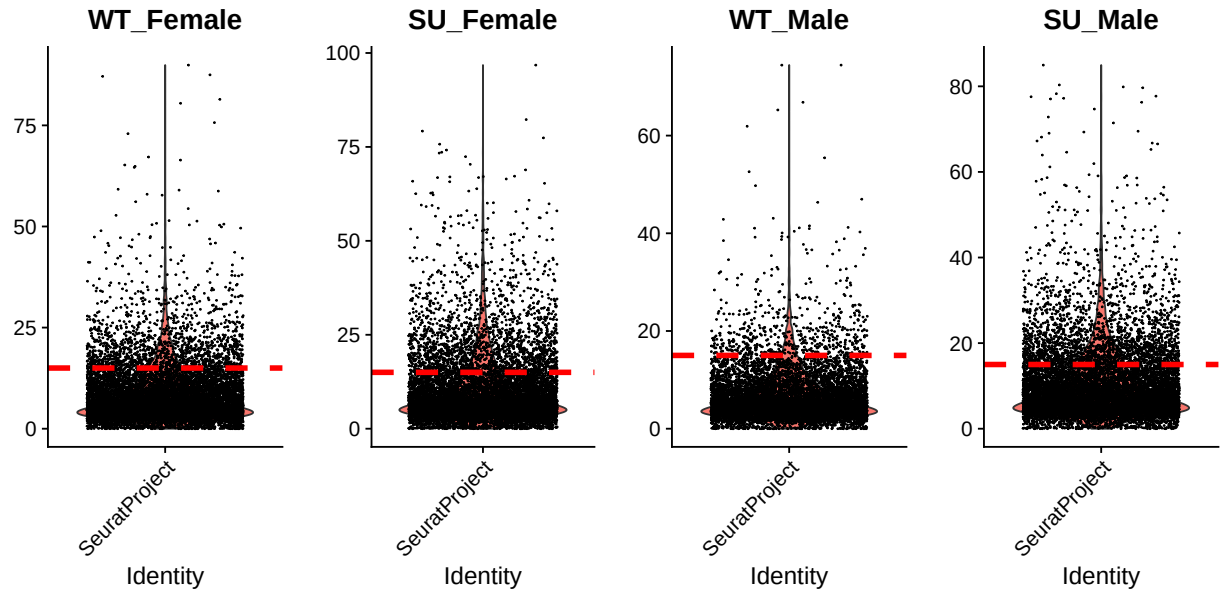
```
p3 <- VlnPlot(C_WT_M, features = "percent.mt") + geom_hline(yintercept = 15, linetype = "dashed", color
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

```
p4 <- VlnPlot(D_SU_M, features = "percent.mt") + geom_hline(yintercept = 15, linetype = "dashed", color
```

```
## Warning: Default search for "data" layer in "RNA" assay yielded no results;
## utilizing "counts" layer instead.
```

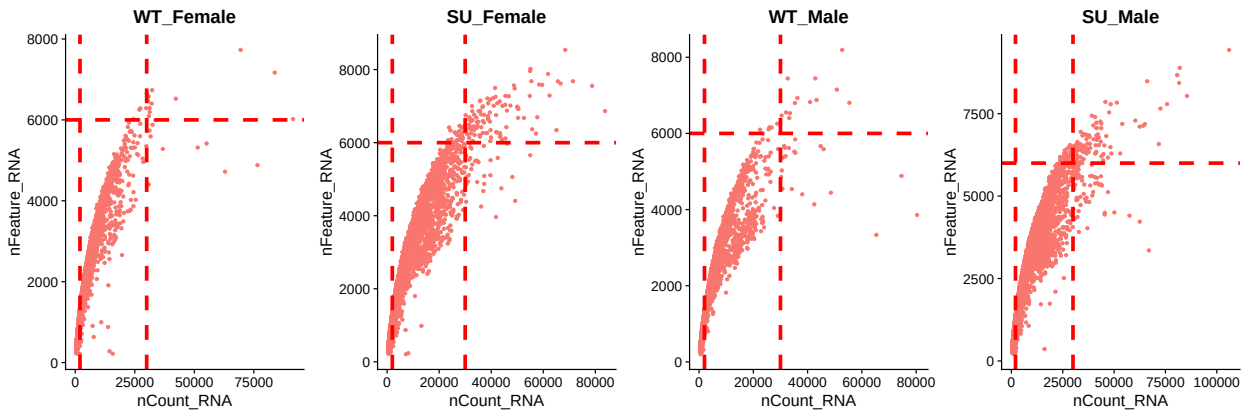
```
#grid.arrange(p1, p2, p3, p4, ncol = 2)
#p <- p1 + p2 + p3 + p4
plot_row <- plot_grid(p1, p2, p3, p4, ncol = 4, align = "h")


print(plot_row)
```




## nFeature vs NCOunt

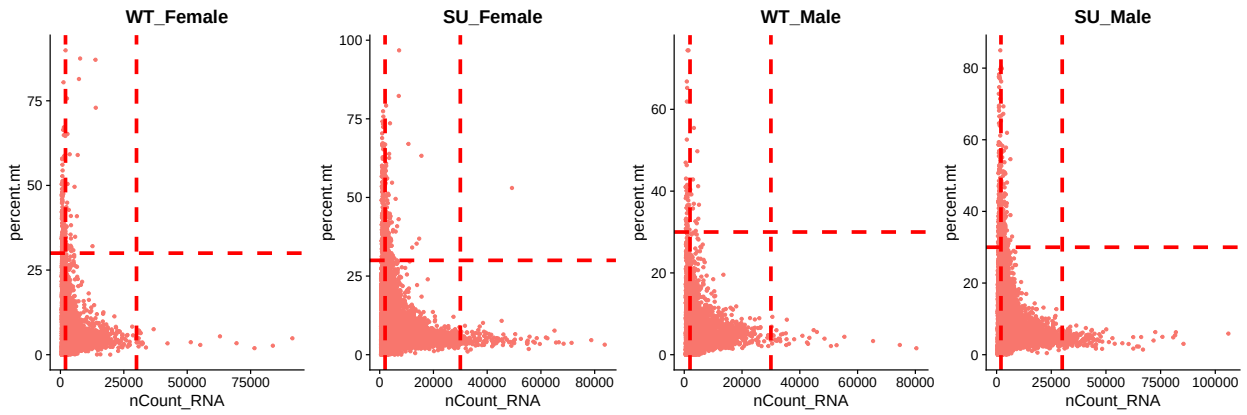
```
library(gridExtra)

p1 <- FeatureScatter(A_WT_F, feature1 = "nCount_RNA", feature2 = "nFeature_RNA") + geom_hline(yintercep

p2 <- FeatureScatter(B_SU_F,  feature1 = "nCount_RNA", feature2 = "nFeature_RNA") + geom_hline(yinterce

p3 <- FeatureScatter(C_WT_M,  feature1 = "nCount_RNA", feature2 = "nFeature_RNA") + geom_hline(yinterce

p4 <- FeatureScatter(D_SU_M,  feature1 = "nCount_RNA", feature2 = "nFeature_RNA") + geom_hline(yinterce


#grid.arrange(p1, p2, p3, p4, ncol = 2)
#p <- p1 + p2 + p3 + p4
plot_row <- plot_grid(p1, p2, p3, p4, ncol = 4, align = "h")
```

```
print(plot_row)
```




# nCount vs MTpercent

```
library(gridExtra)

p1 <- FeatureScatter(A_WT_F, feature1 = "nCount_RNA", feature2 = "percent.mt") + geom_hline(yintercept =

p2 <- FeatureScatter(B_SU_F,  feature1 = "nCount_RNA", feature2 = "percent.mt") + geom_hline(yintercept

p3 <- FeatureScatter(C_WT_M,  feature1 = "nCount_RNA", feature2 = "percent.mt") + geom_hline(yintercept

p4 <- FeatureScatter(D_SU_M,  feature1 = "nCount_RNA", feature2 = "percent.mt") + geom_hline(yintercept


#grid.arrange(p1, p2, p3, p4, ncol = 2)
#p <- p1 + p2 + p3 + p4
plot_row <- plot_grid(p1, p2, p3, p4, ncol = 4, align = "h")


print(plot_row)
```



## nfeatures vs MTpercent

```
library(gridExtra)

p1 <- FeatureScatter(A_WT_F, feature1 = "nFeature_RNA", feature2 = "percent.mt") + geom_hline(yintercep

p2 <- FeatureScatter(B_SU_F,  feature1 = "nFeature_RNA", feature2 = "percent.mt") + geom_hline(yinterce

p3 <- FeatureScatter(C_WT_M,  feature1 = "nFeature_RNA", feature2 = "percent.mt") + geom_hline(yinterce

p4 <- FeatureScatter(D_SU_M,  feature1 = "nFeature_RNA", feature2 = "percent.mt") + geom_hline(yinterce


#grid.arrange(p1, p2, p3, p4, ncol = 2)
#p <- p1 + p2 + p3 + p4
plot_row <- plot_grid(p1, p2, p3, p4, ncol = 4, align = "h")


print(plot_row)
```