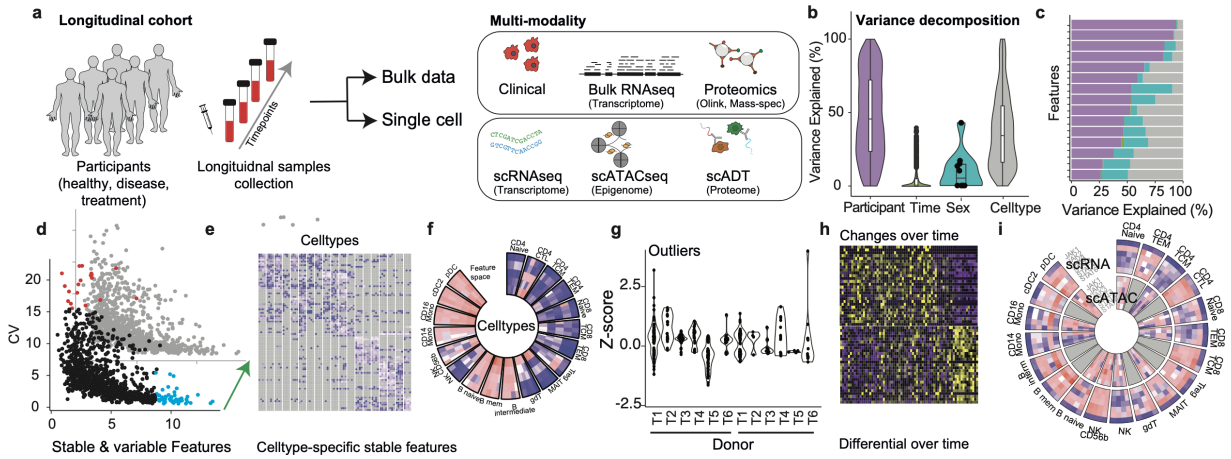
PALMO (Platform for Analyzing Longitudinal Multi-omics data) package

Last compiled on 20 May, 2022

Contents

1 Introduction

PALMO (`Platform for Analyzing Longitudinal Multi-omics data`) is a platform for analyzing longitudinal data from bulk as well as single cell platform. It allows to identify inter-, intra-donor variations in genes over longitudinal time points. The analysis can be done on bulk expression dataset without known celltype information or single cell with celltype/user-defined groups. PALMO can work with any longitudinal data (at least 3 timepoints), including clinical data, bulk omics data, and single-cell omics data. Variance decomposition module in PALMO allows to reveal contributions by factors of interest to the total variance. PALMO allows to infer stable and variable features in each celltype (or user defined group) within each donor. It also identifies stable and variable features across the donors (including within celltype). PALMO helps to detect abnormal temporal behavior in longitudinal data. Further, PALMO allows to visualize multimodal omics data together in Circos plot. Longitudinal differential analysis (time as a continuous variable) can be performed using PALMO to decipher time-wise changes in gene expression within a celltype. The installation of PALMO and its application with several tutorials are given below.



General workflow and analysis schema of **PALMO**. It can work with longitudinal data obtained from bulk such as clinical, bulk RNAseq, proteomic or single cell dataset from scRNAseq, and scATACseq.

2 Install package and load library

To install library, simply run

```
library("devtools")
install_github("aifimmunology/PALMO")
library("PALMO")
```

or from CRAN (https://cran.r-project.org/web/packages/PALMO/index.html)

```
install.packages("PALMO")
library("PALMO")
```

```
library(PALMO)
#> Loading required package: grid
```

3 Tutorials

3.1 Tutorial-1: Plasma proteome (Bulk dataset)

This tutorial allows users to explore bulk plasma proteome data (1156 proteins) measured from 6 healthy donors over 10 timepoints. Plasma proteomic data available at GitHub. 1. AIFI-Olink-NPX_log2_Protein.Rda (Normalized protein expression data) 2. AIFI-Metadata.Rda (clinical metadata). Longitudinal dataset includes 6 donors (3 male and 3 females). PBMC samples were collected from 6 donors over 10 weeks. To interrogate longitudinal data, please follow following steps.

Note: Users need to check the data quality, batch effect or normalization before running PALMO. Confounding effect of variables like batch on features can be deduced using variance decomposition analysis.

3.1.1 Load Library

```
#Load Library and other visualization packages
library("PALMO")
library("Hmisc")
```

3.1.2 Load data and assign parameters (Time ~ 10sec)

The annotation table `metadata` must consist of column `Sample` (Participant sample name), `PTID` (donor/Participant), `Time` (longitudinal time points) information, but not necessarily same column names. Users can assign respective columns in subsequent steps. The data is an Expression data frame, where rows represents gene/proteins and column represents participant samples (same as annotation table `Sample` column). We strongly suggest combining both `PTID` and `Time` column by separator to make `Sample` column. For example, PTID1 and Week1 can be represented as Sample PTID1W1 or PTID1_W1.

```
#Load Plasma proteome data (longitudinal)
load("data/AIFI-Olink-NPX_log2_Protein.Rda")
#Load metadata
load("data/AIFI-Metadata.Rda")
```

3.1.3 Create PALMO object and merge annotation data (Time ~20sec)

First create the PALMO S4 object consisting of expression dataframe and clinical annotations. The expression dataframe columns merged with input annotation dataframe. Only overlapping samples kept. Missing annotations with Sample, Donor/participant, or Time columns are removed from downstream analysis.

```
#Create PALMO object
palmo_obj <- createPALMOobject(anndata=ann, data=data)

#Assign Sample, PTID and Time parameters
palmo_obj<- annotateMetadata(data_object=palmo_obj,
            sample_column= "Sample", donor_column= "PTID",
            time_column= "Time")

#Sample overlap and final matrix
palmo_obj <- mergePALMOdata(data_object=palmo_obj, datatype="bulk")

#Check for replicates
palmo_obj <- checkReplicates(data_object=palmo_obj, mergeReplicates = T)
```

3.1.4 Remove genes with >40%NAs (optional)

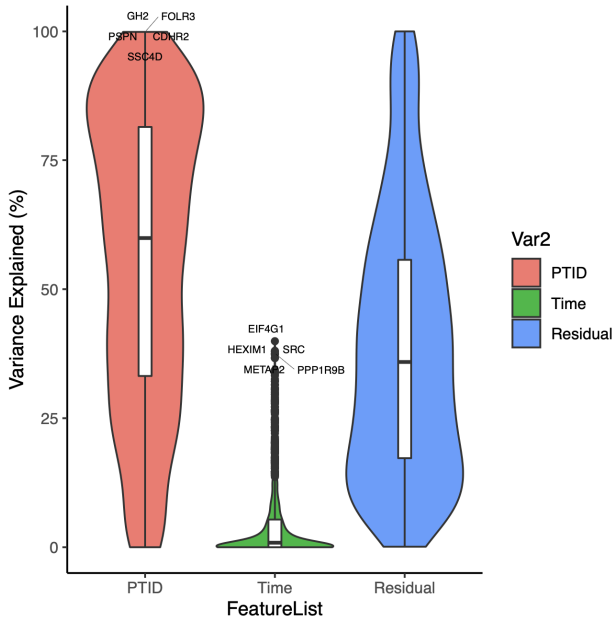
For downstream analysis select genes/proteins with less than 40% of missing values. Users can select cut-off for missing values as necessary.

```
palmo_obj <- naFilter(data_object=palmo_obj, na_cutoff=0.4)
```

3.1.5 Features contributing towards donor variations (Variance decomposition) (Time ~1min)

To perform variance decomposition across donors, apply `lmeVariance` function on PALMO object consisting of input metadata and expression data. The `featureSet` is a list of variables for which fraction of variance explained by each gene is calculated. `meanThreshold` defines the minimum average expression threshold to be used for longitudinal data. Here we used normalized protein expression 1 based on mean expression profile of each gene across longitudinal samples. Residuals suggest the variance cannot be explained by available feature set. The variance explained by each gene towards the featureSet of interest given in percentage. *(Note: To reduce the processing time use nodes cl=8 (or greater) and lmer_control=TRUE)*

```
featureSet <- c("PTID","Time")
palmo_obj <- lmeVariance(data_object=palmo_obj, featureSet=featureSet,
                         meanThreshold=1, fileName="olink")
var_decomp <- palmo_obj@result$variance_decomposition
```



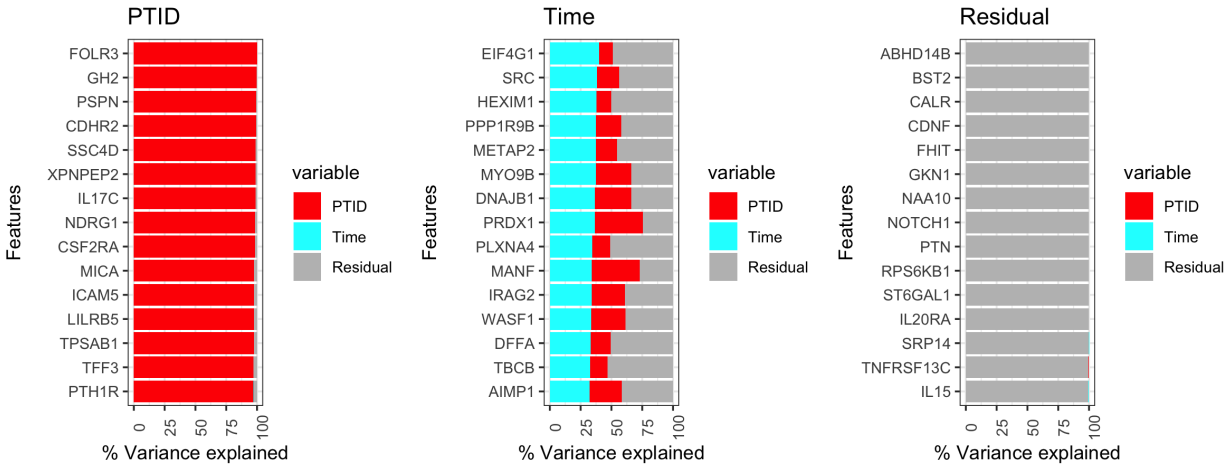

```
head(var_decomp[,c(featureSet, "Residual")])
#Features      donor      week  Residuals
#FOLR3   99.90070 0.0000000 0.09930098
#GH2     99.49856 0.0000000 0.50144042
#PSPN    99.26882 0.1021076 0.62906798
#CDHR2   99.07933 0.1157406 0.80493162
#SSC4D   98.82794 0.0000000 1.17206000
#XPNPEP2 98.67628 0.0000000 1.32372323
```

In this example FOLR3 protein shows high (99%) variations coming from donor.

3.1.6 Donor-specific variance contributing features

The top 15 features contributing to donor, and time attributes variance can be seen in barplot. Users can include other attributes to see variance decomposition on user-defined attributes.

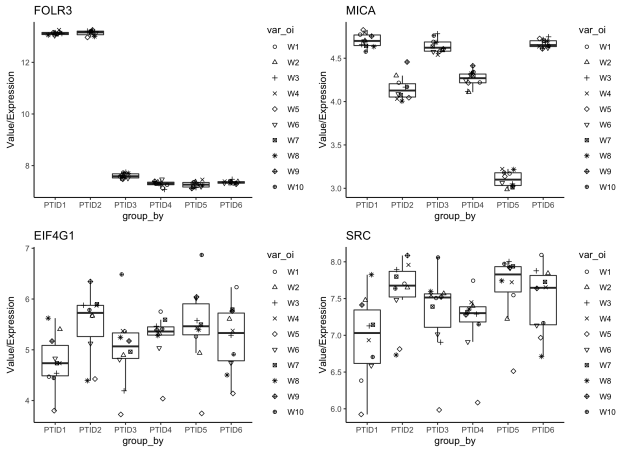
```
plots <- variancefeaturePlot(vardata=var_decomp,
featureSet=featureSet,
            Residual=T)
```



3.1.7 Plot top features

Protein expression plot for interested features can be visualized.

```
plots <- gene_featureplot(data_object=palmo_obj,
                          featureList=c("FOLR3", "MICA", "EIF4G1", "SRC"),
                          x_group_by="PTID", var_oi="Time")
```

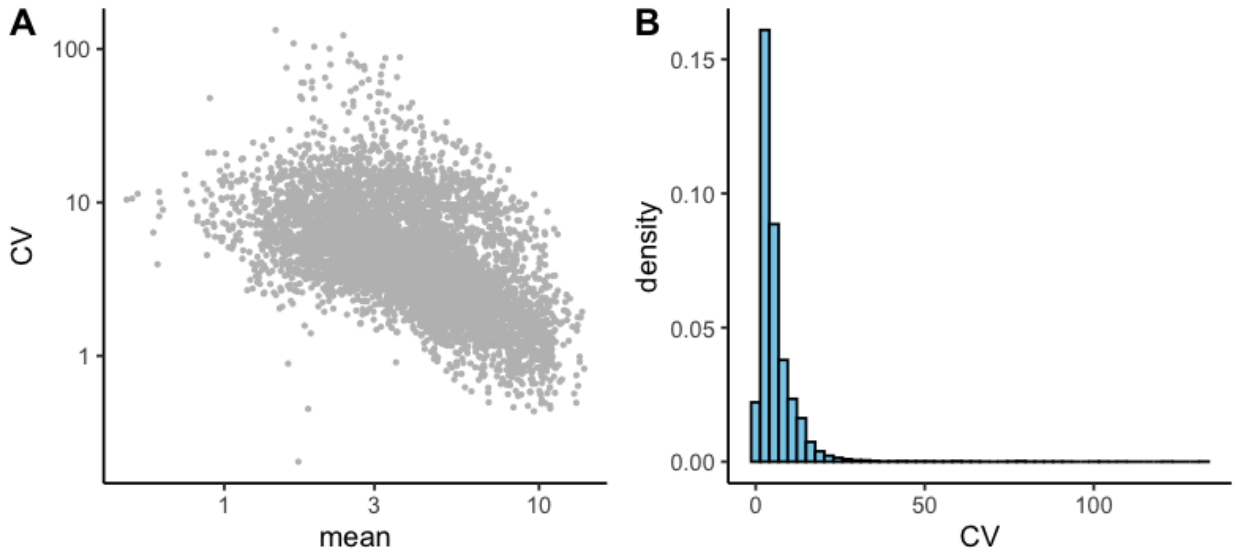


3.1.8 Intra-donor variations over time

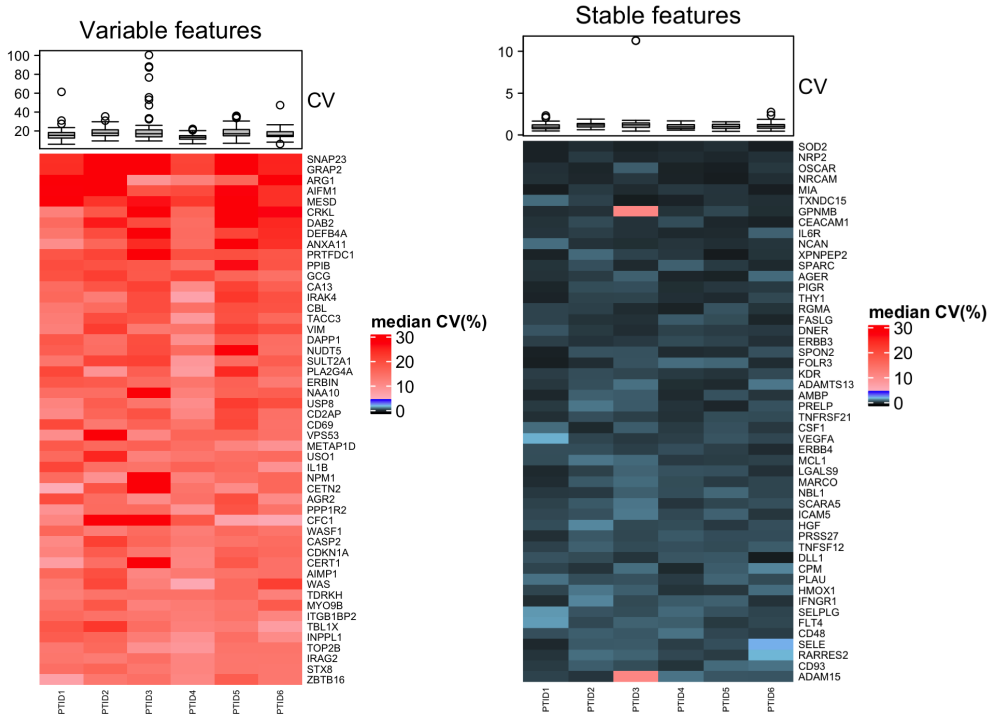
To calculate longitudinal stability within donor first visualize the CV (coefficient of variation) vs mean distribution for given data. The CV distribution plot allows users to visualize the CV vs mean relation and

helps to define threshold using the histogram or selecting few features of interest CV as cut-off. We used CV=5 as threshold here.

```
#CV dsitribution
cv <- cvCalcBulkProfile(data_object=palmo_obj)
```




```
#Find the stable and variable features from the user-defined cut-off
palmo_obj <- cvCalcBulk(data_object=palmo_obj, meanThreshold=1, cvThreshold=5)
```



The longitudinally stable and variable features within donors are identified using user-defined cut-off. The variable and stable heatmap shows top 50 features. Here each cell represents the CV value and color of value

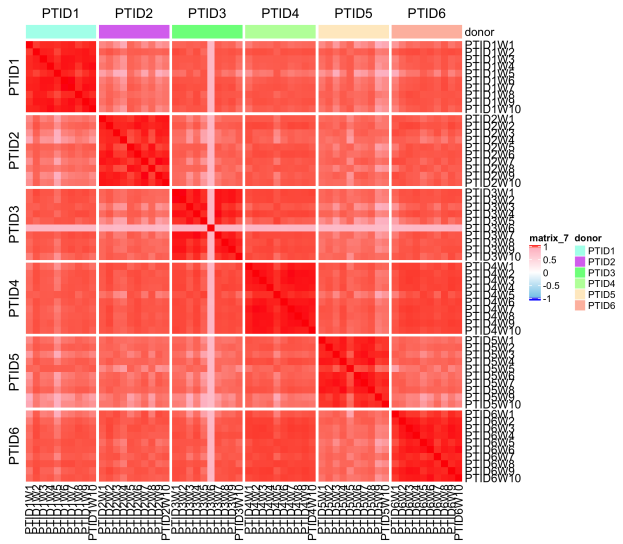
is black (CV=zero), blue (less than or equal to CV threshold) and red (greater than CV threshold). The data results can be retrieved from PLAMO S4 object as given below.

```
#Variable genes
head(palmo_obj@result[["variable_gene"]])
head(palmo_obj@result[["var_mat"]])
#Non-variable genes (stable)
head(palmo_obj@result[["non_variable_gene"]])
head(palmo_obj@result[["stable_mat"]])
```

3.1.9 Outlier analysis (Time ~ 30sec)

PALMO applies both graphic and statistical methods to examine the temporal behavior of longitudinal data. It calculates intra- and inter-donor correlations (across analytes) and displays the results in a heatmap. Timepoints showing obvious weaker correlations with other timepoints are potential outliers. Users can plot heatmap by donor (group_by = "PTID"), time (group_by = "Time") or group of interest.

```
#Sample variability (Correlation)
palmo_obj <- sample_correlation(data_object=palmo_obj, group_by="PTID")
```

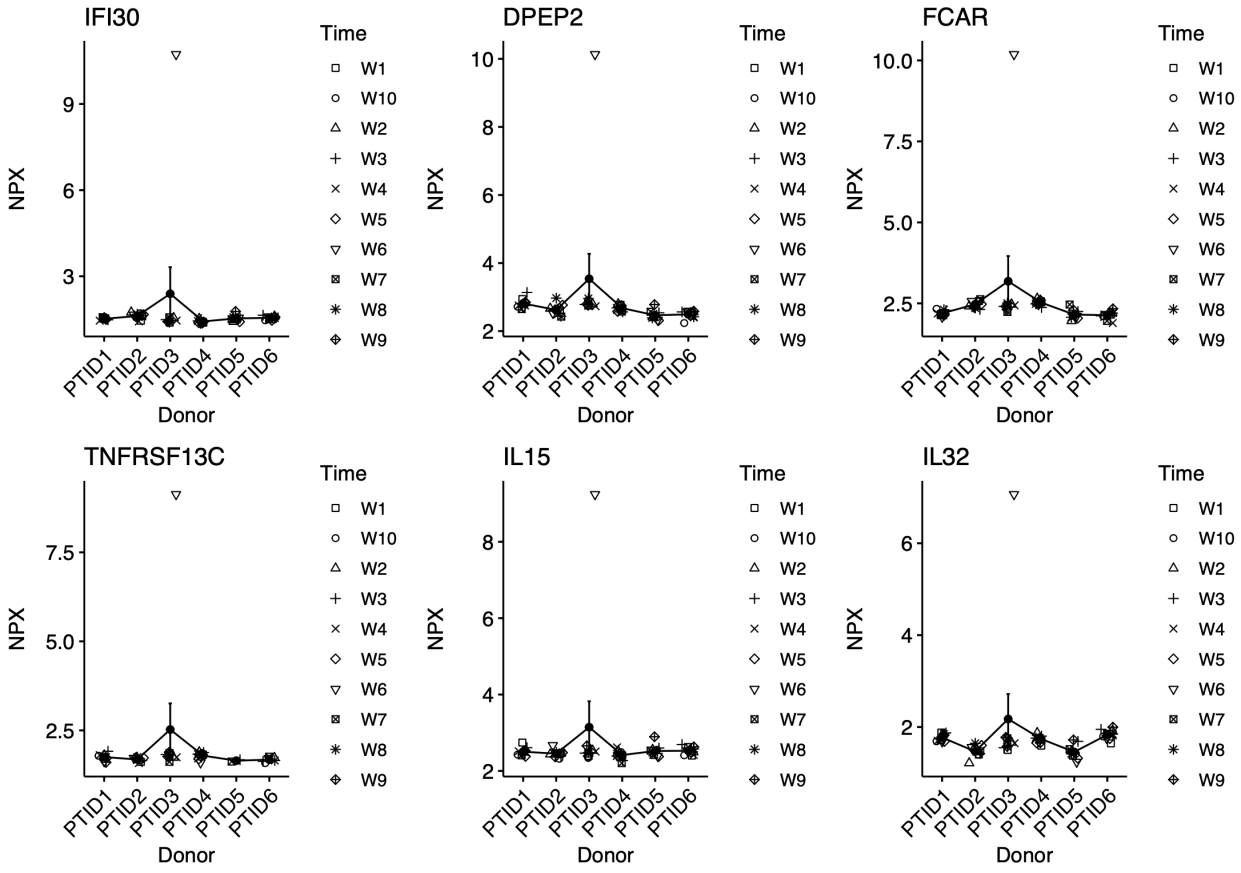


To detect abnormal timepoints, PALMO first calculates the mean and the standard deviation (SD) of each feature from samples of the same donor across all timepoints, then calculates `z=(value-mean)/SD` for the feature at individual timepoints. Using a user selected cutoff value of Z, the feature is identified as an outlier at the given time point if `|Z|>Z_cutoff`.

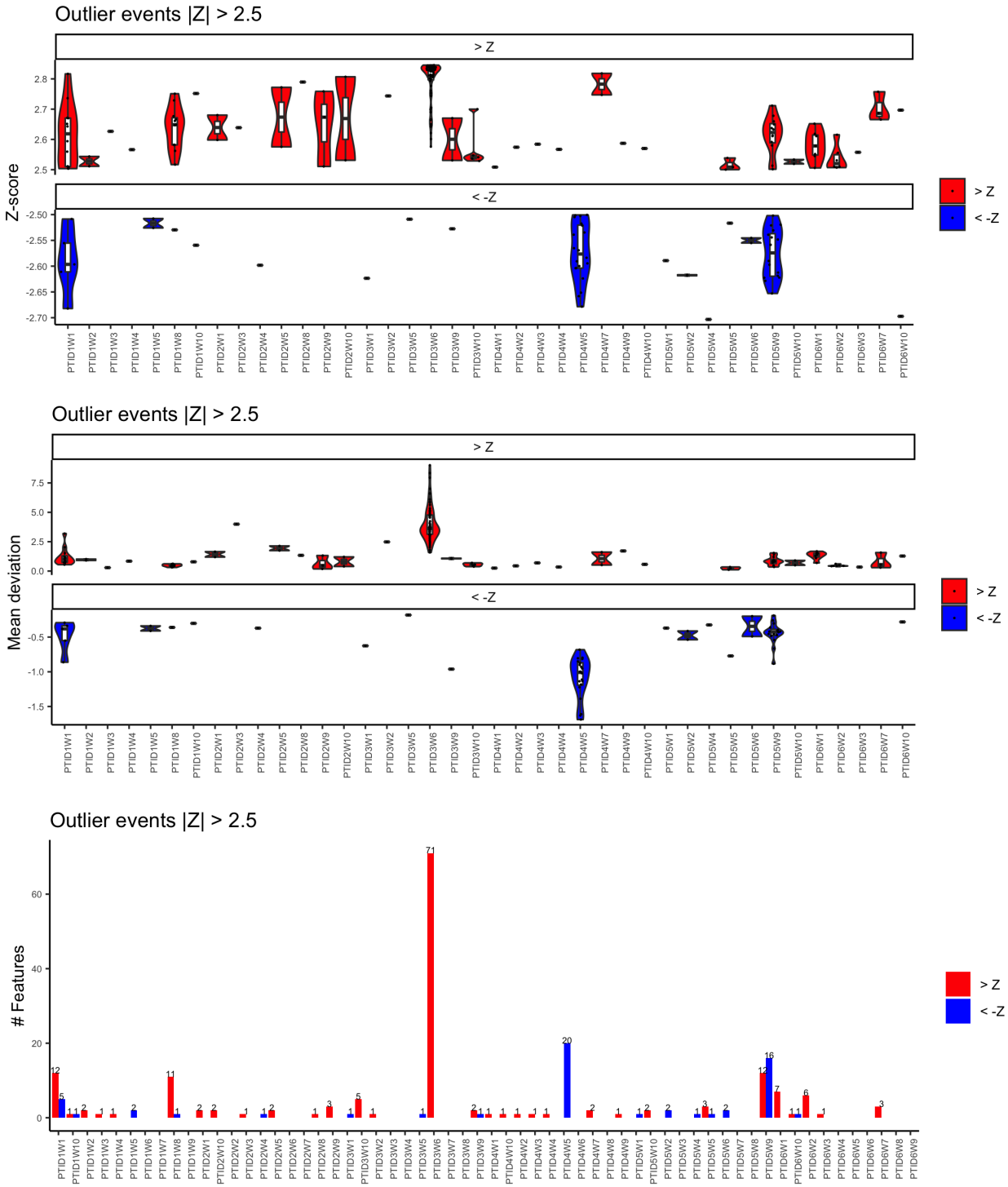
```
#Detect outliers (if any)
palmo_obj <- outlierDetect(data_object=palmo_obj, z_cutoff=2.5)
outlier_res <- palmo_obj@result[["outlier_res"]]
head(outlier_res)
#  exp  Sample  PTID Time    Sex      gene  meanDev        z
#PTID3W643 10.729880 PTID3W6 PTID3   W6 Female     IFI30 8.340474 2.845471
#PTID3W627 10.133645 PTID3W6 PTID3   W6 Female     DPEP2 6.595705 2.844629
#PTID3W631 10.188437 PTID3W6 PTID3   W6 Female      FCAR 7.004890 2.844607
#PTID3W626  7.656418 PTID3W6 PTID3   W6 Female     DPEP1 4.574449 2.844574
#PTID3W685  9.129149 PTID3W6 PTID3   W6 Female TNFRSF13C 6.605767 2.844410 2.844410
#PTID3W652  8.031215 PTID3W6 PTID3   W6 Female   KIR2DL3 5.156982 2.844018
```

Users can plot top features with `|Z|>Z_cutoff`.

```
#Gene plot (probable outliers)
plots <- gene_featureplot(data_object=palmo_obj,
                 featureList=c("IFI30", "DPEP2","FCAR",
                               "TNFRSF13C", "IL15", "IL32"),
                 x_group_by="PTID", var_oi="Time")
```

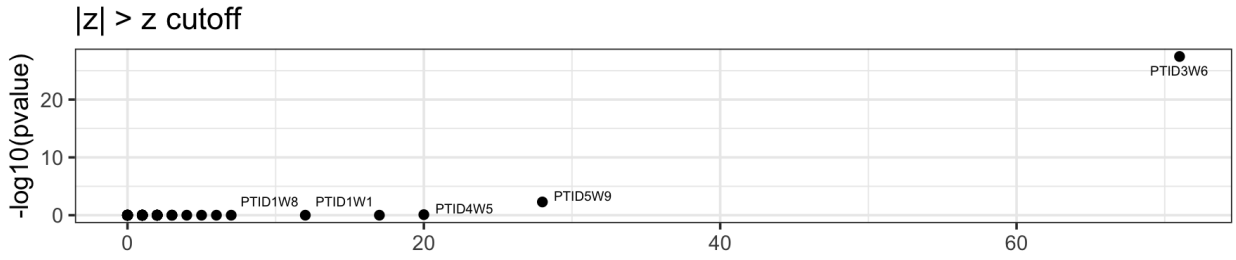


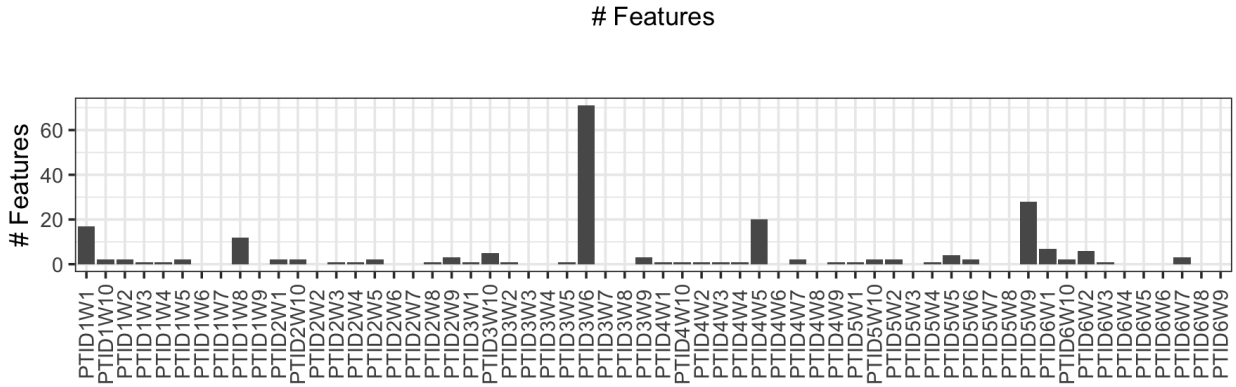

```
#Number of outlier features
num_outlier <- data.frame(table(outlier_res$Sample))
num_outlier <- num_outlier[order(num_outlier$Freq, decreasing = T),]
head(num_outlier)
#Var1 Freq
#PTID3W6   71
#PTID5W9   28
#PTID4W5   20
#PTID1W1   17
#PTID1W8   12
#PTID6W1    7
```

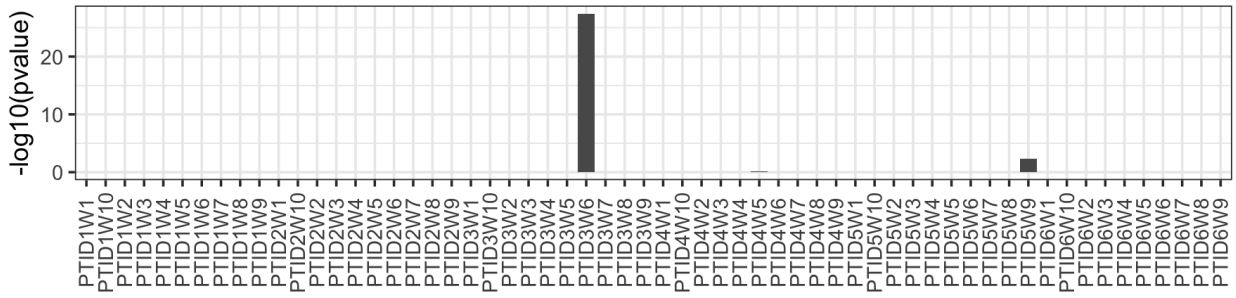


Assuming `Z` follows a normal distribution the expected rate `r` of features having `|Z|>Z_cutoff` (two-sided) or having `Z>Zcutoff` or `Z< -Zcutoff` (one-sided). Afterwards PALMO counts how many features are outliers at individual timepoints and uses binomial tests to evaluate the corresponding `p` values, and applies Benjamini & Hochberg procedure to adjust the `p` values since multiple timepoints are tested. A donor-specific abnormal timepoint is identified if the corresponding adjusted `p` value is less than 0.05.

```
#Calculate p value
outlierDetectP(outlier_events=outlier_res, z_cutoff=2.5, nGenes=1046)
```







3.2 Tutorial-2: scRNA longitudinal data (n=4 and 6 weeks follow-up)

This tutorial allows users to explore single cell RNAseq data measured from 4 healthy donors over 6 time points (week 2-7). Single cell data is available at GSE190992*. (1) AIFI-scRNA-PBMC-FinalData.RDS (Normalized scRNA seurat object) (2) AIFI-Metadata.Rda (clinical metadata). Longitudinal data set includes 4 donors and 6 donors (24 samples). To infer iner-donor, intra-donor variations, and stable features, please follow following steps.

(Note:If any issue with data access or needed RAW files please Contact Us).

3.2.1 Load Library

```
#Load Librarys
library("PALMO")
library("Hmisc")
```

Load single cell data and define the `Sample` column which is same as `Sample` column in metadata.

3.2.2 Load single cell data and metadata

```
#scRNA seurat object
pbmc <- readRDS("data/AIFI-scRNA-PBMC-FinalData.RDS")
metaData <- pbmc@meta.data
pbmc@meta.data$Sample <- pbmc@meta.data$orig.ident
pbmc@meta.data$celltype <- gsub(" ", "_", pbmc@meta.data$celltype)

#Load annotation datas
load("data/AIFI-Metadata.Rda")
```

We selected major 19 celltypes based on proportion. Users can select celltype of interest to investigate longitudinal stability.

```
#Group of ineterst
avgGroup <- "celltype"
#Celltypes observed in dataset
cell_type <- sort(unique(pbmc@meta.data$celltype))
#Celltypes selected for analysis consisting atleast >5% of cells in each celltype.
celltype_oi <- c("CD4_Naive","CD4_TEM","CD4_TCM","CD4_CTL","CD8_Naive",
                 "CD8_TEM","CD8_TCM","Treg","MAIT","gdT",
                 "NK", "NK_CD56bright",
                 "B_naive", "B_memory", "B_intermediate",
                 "CD14_Mono","CD16_Mono",
                 "cDC2","pDC")
```

3.2.3 Create PALMO object (Time ~ 2min)

First create the PALMO S4 object using input scRNA object and annotation dataframe. The expression dataframe columns merged with input annotation dataframe. Only overlapping samples kept. Missing annotations with Sample, Donor/participant, or Time columns are removed from downstream analysis.

```
#Create PALMO object
palmo_obj <- createPALMOobject(anndata=ann, data=pbmc)
```

Assign `sample_column`, `donor_column` and `time_column` variables in this step.

```
#Assign Sample, PTID and Time parameters
palmo_obj <- annotateMetadata(data_object=palmo_obj,
                          sample_column= "Sample",
                          donor_column= "PTID",
                          time_column= "Time")
```

For single cell data merge annotation and single cell metadata by mentioned `sample_column`.

```
#Sample overlap and final matrix
palmo_obj <- mergePALMOdata(data_object=palmo_obj, datatype="singlecell")
```

Single cell data is aggregated by average method at sample group level. Features with average expression greater than zero across all samples are kept.

```
#Aggregate data (Psuedo-bulk)
palmo_obj <- avgExpCalc(data_object=palmo_obj,
                    assay="RNA", group_column="celltype")
head(palmo_obj@curated[["anndata"]]) #merged annotation data
head(palmo_obj@curated[["data"]]) #scRNA average expression data
```

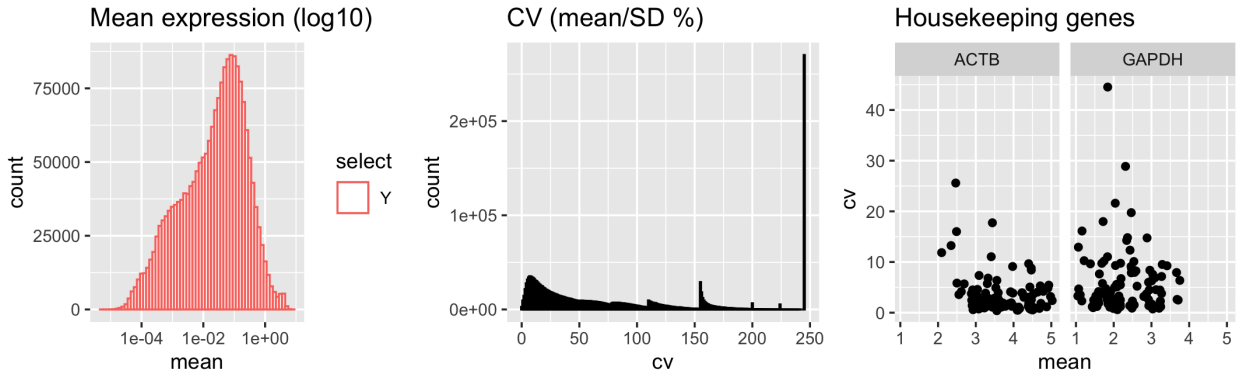
Optional step. If data consists of replicates then they can be merged by `mergeReplicates = TRUE`.

```
#Check for replicates
palmo_obj <- checkReplicates(data_object=palmo_obj, mergeReplicates = T)
```

3.2.4 CV profile (Time ~ 5min)

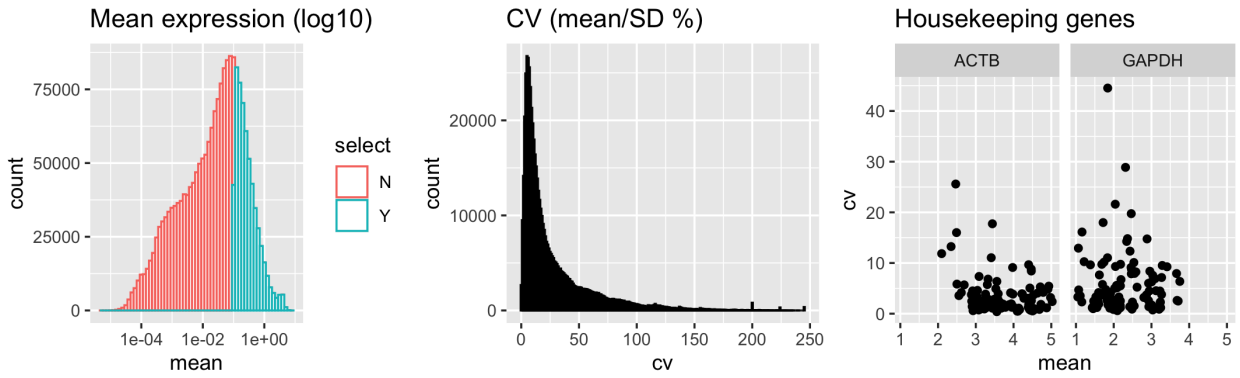
To calculate longitudinal stability within donor first visualize the CV vs mean distribution for given data. Define CV threshold using the histogram or selecting few features of interest CV as cut-off. Here we used housekeeping genes **GAPDH** and **ACTB** to define CV threshold.

```
#Check the mean expression and CV across groups (celltypes)
palmo_obj <- cvCalcSCProfile(data_object=palmo_obj,
                         housekeeping_genes=c("GAPDH", "ACTB"),
                         fileName="scrna")
```



Here, CV (coefficient of variation) vs mean plot shows spikes at end which are associated with lowly expressed genes. Thus, users can select appropriate `meanThreshold` to remove features with very low expression. Overall CV pattern for housekeeping genes can be explored on right panel to choose optimum CV threshold.

```
#Lowly expressed genes show abnormal CV
palmo_obj <- cvCalcSCProfile(data_object=palmo_obj, meanThreshold = 0.1,
                        housekeeping_genes=c("GAPDH", "ACTB"),
                        fileName="scrna")
```



3.2.5 Donorwise CV profile over longitudinal timepoints (Time ~ 4min)

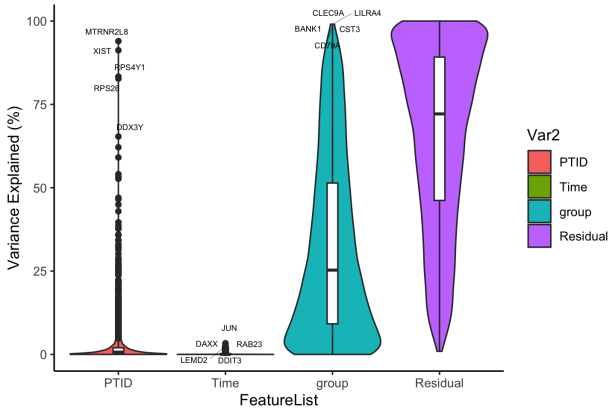
Optional step. To check the donor or participant wise CV profile run the CV sample profile analysis.

```
#Sample Celltype Mean-CV plot (check output folder)
cvSCsampleprofile(data_object=palmo_obj, meanThreshold = 0.1,
cvThreshold = 10)
```

3.2.6 Features contributing towards donor variations (Time ~ 10min)

Variance decomposition analysis was performed to identify the features associated with attributes of interest such as participants, sex, disease type, celltype, or batch. The `featureSet` is a list of variables for which fraction of variance explained by each gene is calculated. The variance explained by each gene towards the `featureSet` of interest given in percentage. *(Note: To reduce the processing time use nodes cl=8 (or greater) and lmer_control=TRUE)*

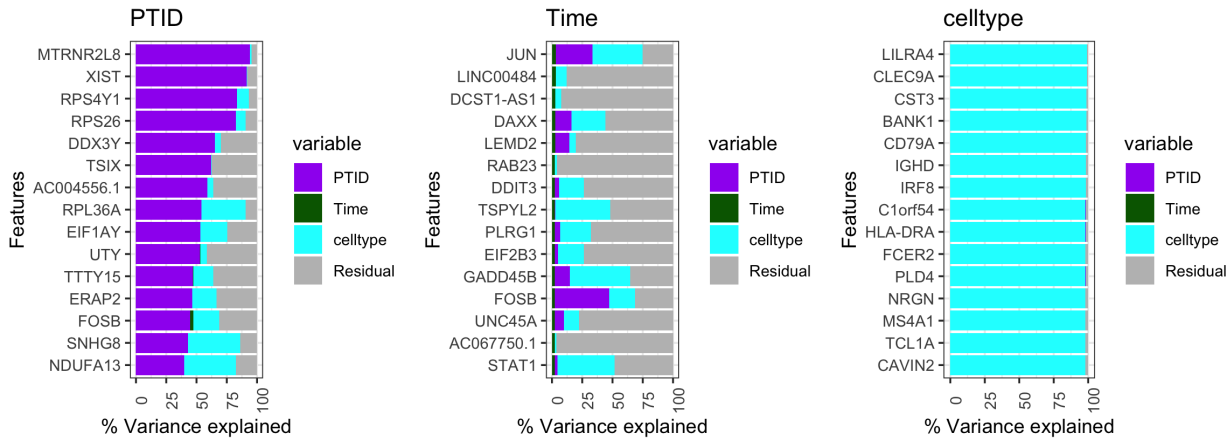
```
#Check the group of interest
head(palmo_obj@curated$anndata)

#Variance decomposition
featureSet <- c("PTID", "Time","celltype")
palmo_obj <- lmeVariance(data_object=palmo_obj,
                         featureSet=featureSet,
                         meanThreshold=0.1, cl=4,
                         fileName="scrna")

#Get the result
var_decomp <- palmo_obj@result$variance_decomposition
```



3.2.7 Variance plot

The top 15 features contributing to donor, time or celltype attributes variance can be seen in barplot.

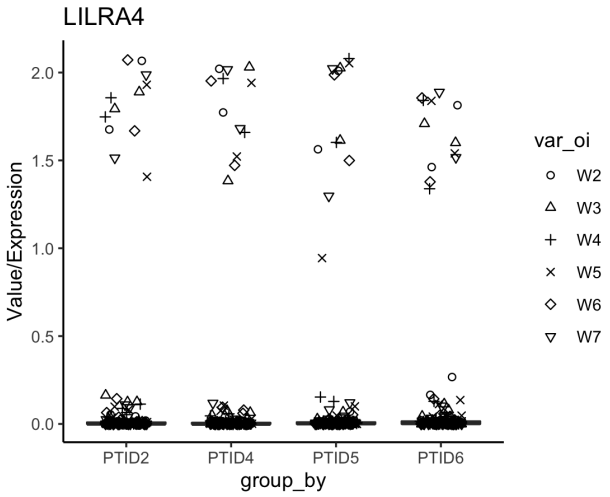
```
plots <- variancefeaturePlot(vardata=var_decomp, featureSet=featureSet,
                             cols=c("purple", "darkgreen", "cyan"))
```



3.2.8 Plot the top features

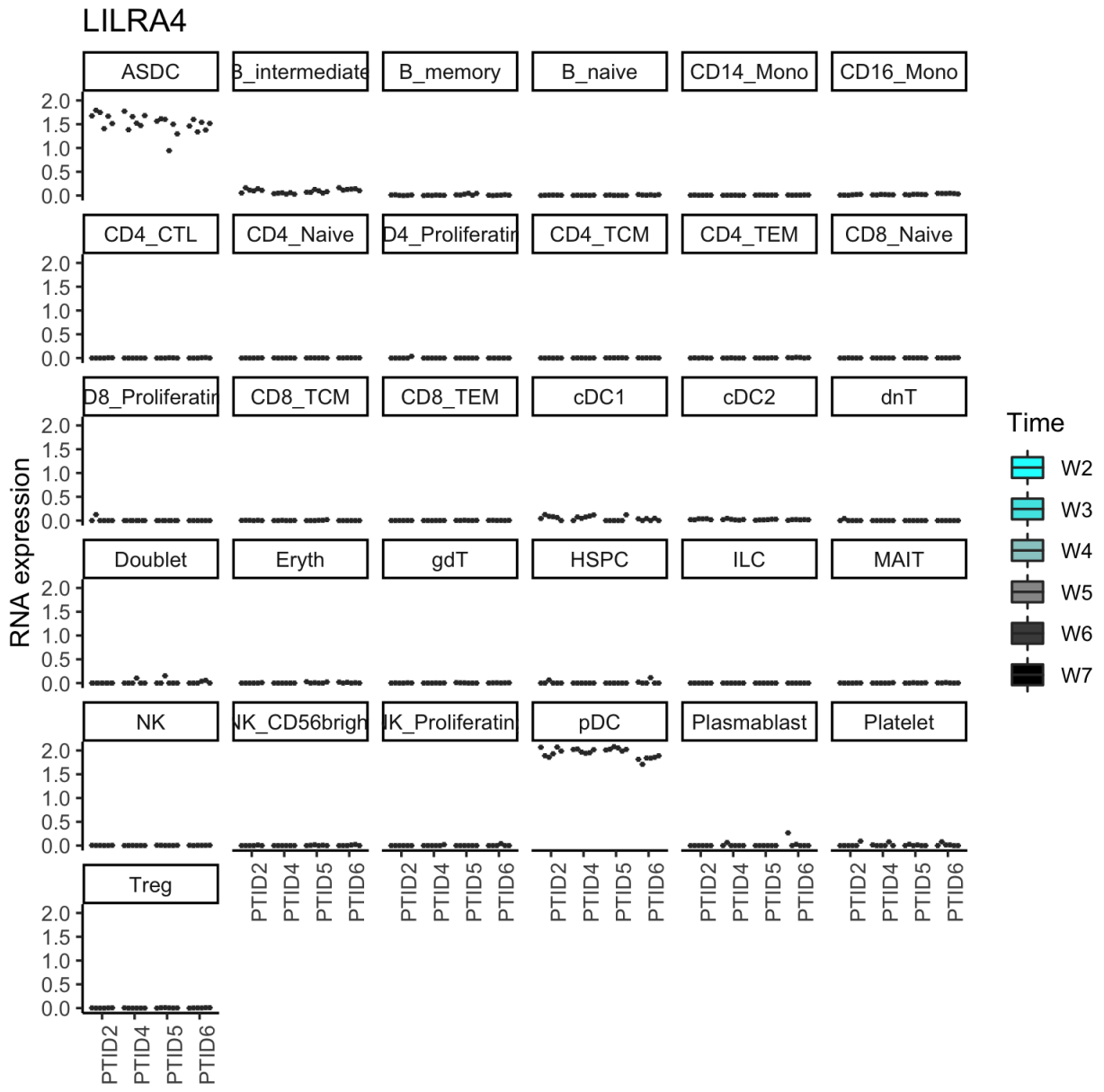
Gene expression plot for interested features can be visualized by donors or by time.

```
gene_featureplot(data_object=palmo_obj, featureList="LILRA4")
```



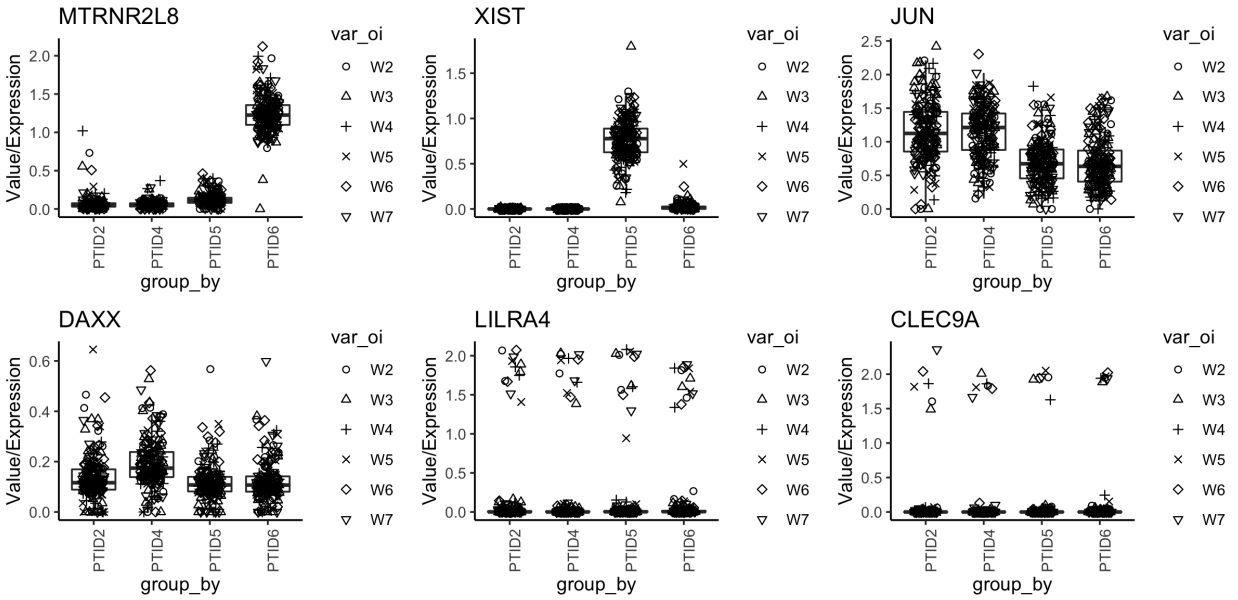
Gene expression plot for interested features can be visualized by celltypes.

```
gene_featureplot(data_object=palmo_obj, featureList="LILRA4", facet_by="group")
```



Further multiple features expression pattern can be visualized.

```
#Multiple-gene visulization
plots <- gene_featureplot(data_object=palmo_obj,
                          featureList=c("MTRNR2L8", "XIST",
                                        "JUN", "DAXX",
                                        "LILRA4", "CLEC9A"),
                          x_group_by="PTID", var_oi="Time",
                          x_text_angle=90)
```

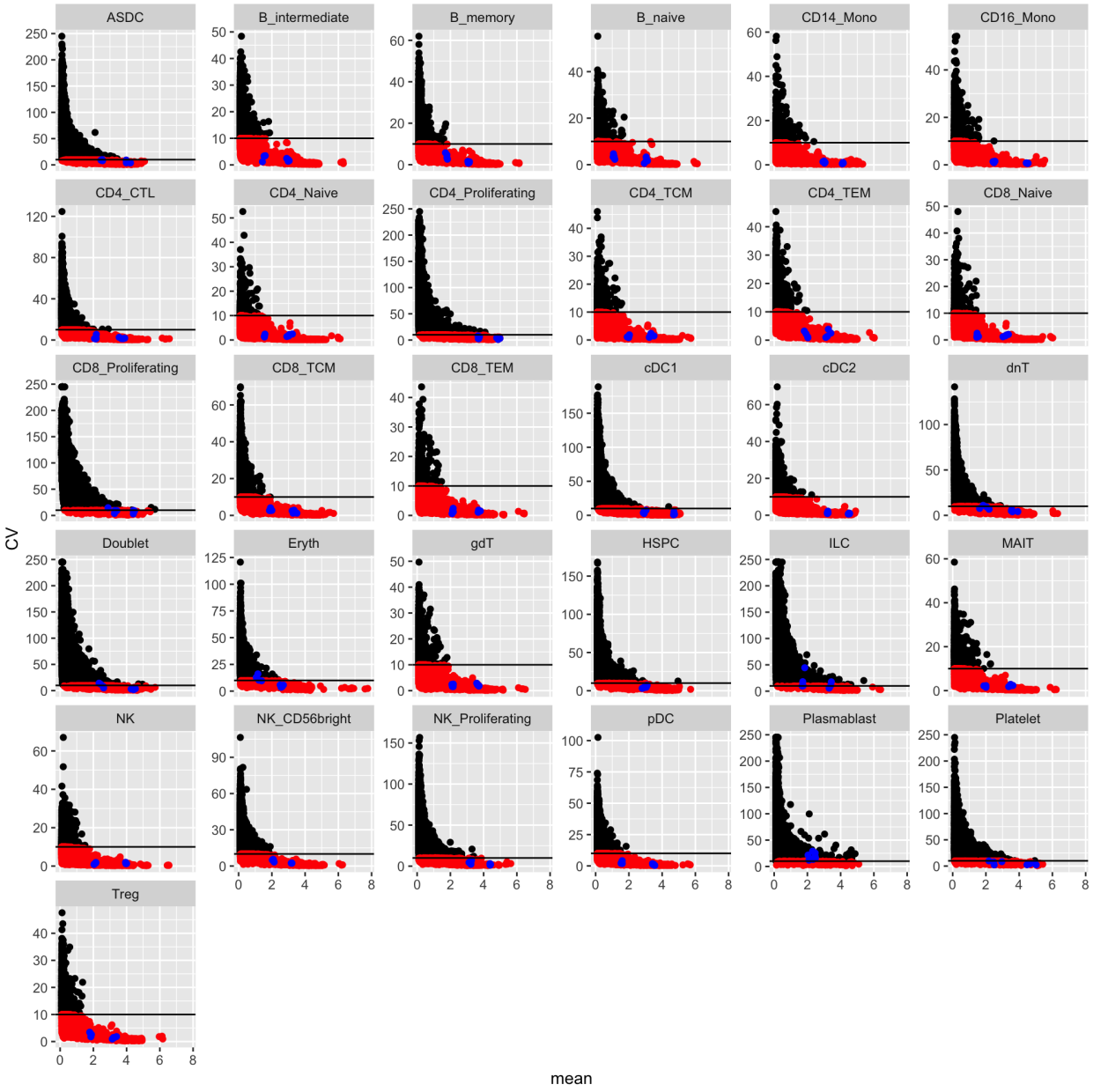


3.2.9 Intra-donor variations over time (Time ~10min)

To calculate longitudinal stability within donor define `meanThreshold` and `cvThreshold` parameters as discussed above. The analysis will calculate the CV across average group ("celltype") mentioned by user. It will create a CV-Mean plots for individual donor over longitudinal timepoints that shows the highly variable and stable features in each donor. The plots are stored in `output` directory if not mentioned. It also provides a housekeeping genes position in CV-Mean plot across each celltype/group of interest. *(Note: To reduce the processing time use nodes cl=8 (or greater))*

```
#Calculate CV
palmo_obj <- cvCalcSC(data_object=palmo_obj,
                      meanThreshold=0.1, cvThreshold=10,
                      housekeeping_genes=c("GAPDH", "ACTB"),
                      fileName="scrna")
```

Plots saved in user-defined output directory



(Note: Black dots, all genes; red dots, genes with given mean and CV threshold; blue dots, housekeeping genes)

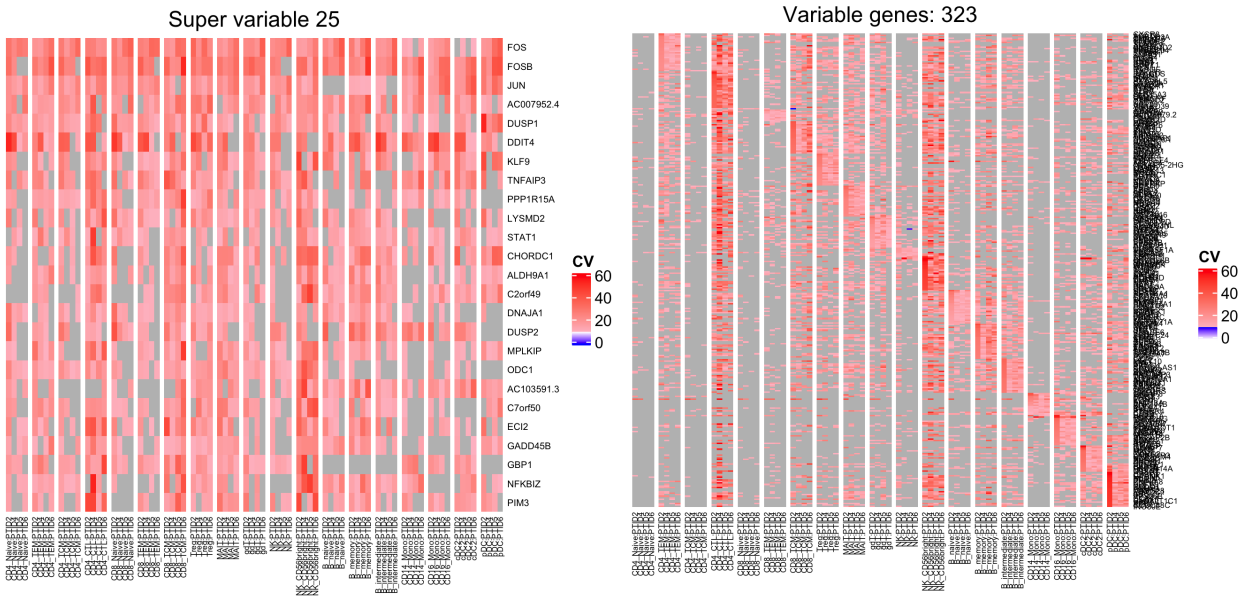
3.2.10 Find stable and variable features in longitudinal data (Time <1 min)

Above step resulted CV values used to identify Stable and variable features across celltype or group of interest. Define minimum number of donors (`donorThreshold`) for consideration and group threshold (`groupThreshold`) which means among group of interest (celltypes here) how many minimum groups should have consensus stable/variable profile. For example T-cell subtypes can share similar stable features. This value is arbitrary but suggest criteria to be more stringent or lenient in selecting features. For example, `donorThreshold=4` and `groupThreshold=40` suggests identifying stable/variable features in all 4 donors with atleast in 40 donor-group from 4 donors and 19 celltypes. These parameters are optional and PALMO can calculate it for users. The general formula to select number of donors is more than 50% of donors ($> N/2$) and group threshold $groupThreshold = numberofDonors * (numberofCelltypes/2)$. For stringency users

can select group threshold=number of donors. Here its (~4x19/2) where we considered about 40 samples (less stringent). Left panel shows super-variable (SUV) or super-stable (SUS) features that shows high CV over longitudinal timepoints across celltypes. Whereas right panel shows variable features or stable (here STATIC, longitudinally stable transcription across time in cell-types) across celltypes.

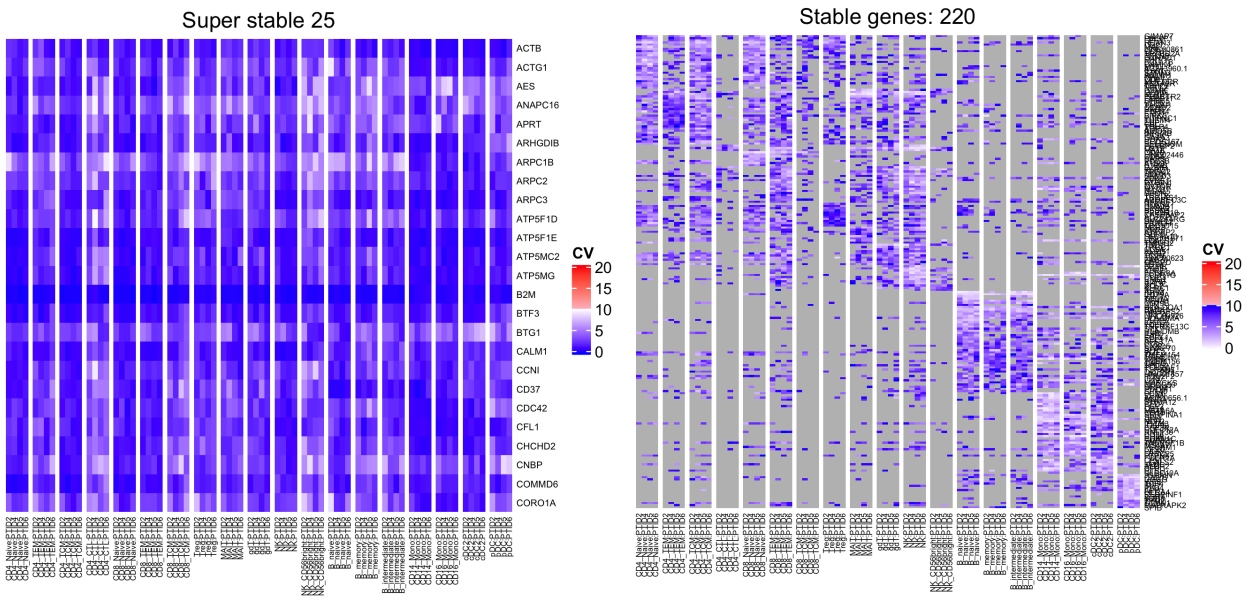
```
palmo_obj <- VarFeatures(data_object=palmo_obj, group_oi=celltype_oi,
                         cvThreshold=10,
                         donorThreshold=4,
                         groupThreshold=40,
                         topFeatures=25,
                         fileName="scrna")
var_genes <- palmo_obj@result$var_genes
```

Variable genes observed in longitudinal data (CV>10%)




```
palmo_obj <- StableFeatures(data_object=palmo_obj, group_oi=celltype_oi,
                            cvThreshold=10,
                            donorThreshold=4, groupThreshold=40,
                            topFeatures=25,
                            fileName="scrna")
stable_genes <- palmo_obj@result$stable_genes
```

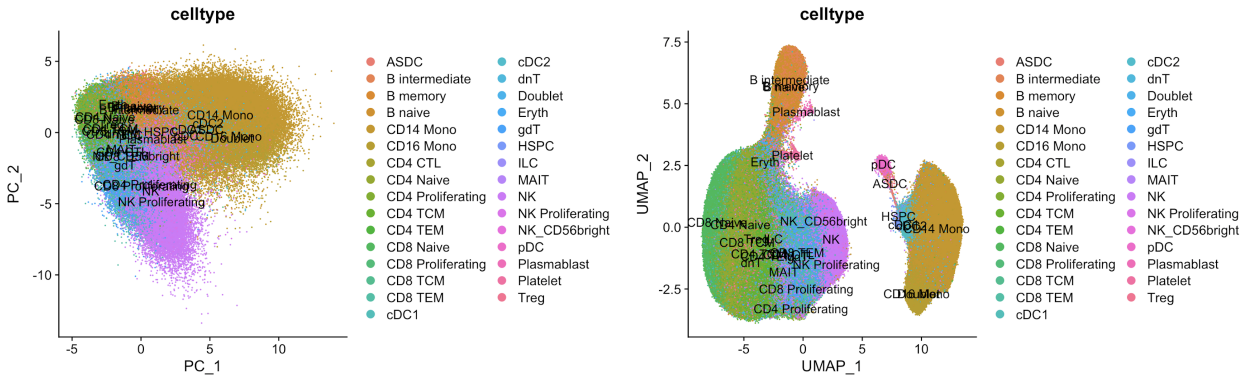
Stable genes observed in longitudinal data (CV<10%)



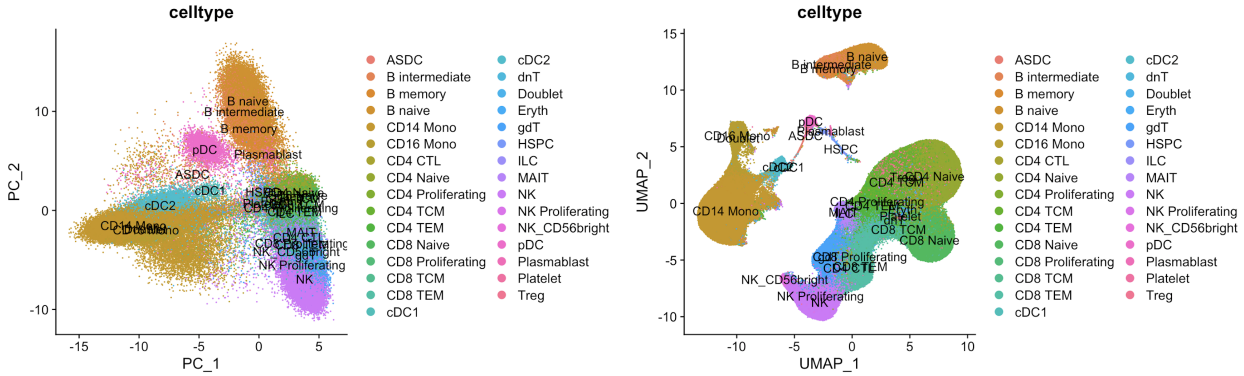
3.2.11 UMAP Plot (Stable/variable) (Time ~ 5min)

To visualize how these stable or variable features performing, users can reduce high dimensional single cell RNA (scRNA) data into lower dimensions using selected Top features from stable/variable analysis. The UMAP plot shows whether identified stable features can reproduce celltype/group identity based on few but stable features.

```
group_column <- "celltype"
#Top variable and stable features used for UMAP
dimUMAPPlot(data_object=palmo_obj, nPC=15,
        gene_oi=unique(var_genes$gene),
        group_column=group_column, plotname="variable",
        fileName="scrna")
```



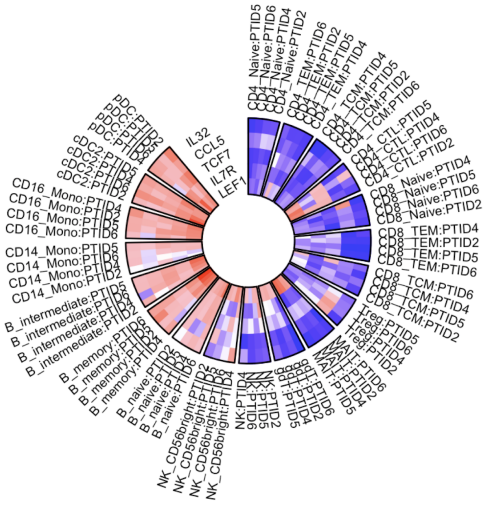

```
dimUMAPPlot(data_object=palmo_obj, nPC=15,
        gene_oi=unique(stable_genes$gene),
        group_column=group_column, plotname="stable",
        fileName="scrna")
```



3.2.12 Circular gene expression plot

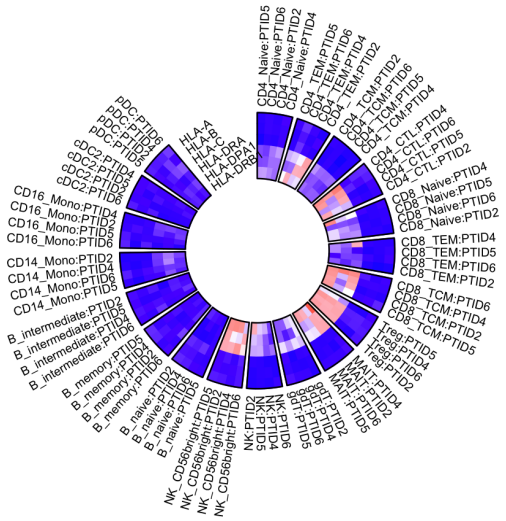
The top features or gene families can be visualized in circos plot for easy comparison.

```
geneList <- c("IL32","CCL5","TCF7","IL7R","LEF1") #T-cell
plotres <- genecircosPlot(data_object=palmo_obj, geneList=geneList,
                          group_oi=celltype_oi, colorThreshold=10)
```



Users can also load PALMO output result.

```
#Get CV result
cv_res <- palmo_obj@result[["cv_all"]]

#Cannonical and non-canonical HLA markers
geneList <- c("HLA-A","HLA-B","HLA-C","HLA-DRA","HLA-DPA1","HLA-DRB1")
plotres <- genecircosPlot(data=cv_res, geneList=geneList,
                          titleName="HLA", group_oi=celltype_oi,
                          colorThreshold=10)
```


CD4_Naive:PTID5
CD4_Naive:PTID6
CD4_Naive:PTID2
CD4_Naive:PTID4
CD4_TEM:PTID5
CD4_TEM:PTID6
CD4_TEM:PTID4
CD4_TEM:PTID2
CD4_TCM:PTID2
CD4_TCM:PTID6
CD4_TCM:PTID5
CD4_TCM:PTID4
CD4_CTL:PTID4
CD4_CTL:PTID6
CD4_CTL:PTID5
CD4_CTL:PTID2
CD8_Naive:PTID4
CD8_Naive:PTID5
CD8_Naive:PTID6
CD8_Naive:PTID2
CD8_TEM:PTID4
CD8_TEM:PTID5
CD8_TEM:PTID6
CD8_TEM:PTID2
CD8_TCM:PTID6
CD8_TCM:PTID4
CD8_TCM:PTID2
CD8_TCM:PTID5
Treg:PTID5
Treg:PTID4
Treg:PTID6
Treg:PTID2
MAIT:PTID4
MAIT:PTID2
MAIT:PTID6
MAIT:PTID5
NK:PTID2
NK:PTID5
NK:PTID4
NK:PTID6
NK_CD56bright:PTID2
NK_CD56bright:PTID5
NK_CD56bright:PTID4
NK_CD56bright:PTID6
B_naive:PTID2
B_naive:PTID4
B_naive:PTID5
B_naive:PTID6
B_memory:PTID5
B_memory:PTID4
B_memory:PTID2
B_memory:PTID6
B_intermediate:PTID2
B_intermediate:PTID5
B_intermediate:PTID4
B_intermediate:PTID6
CD14_Mono:PTID2
CD14_Mono:PTID4
CD14_Mono:PTID6
CD14_Mono:PTID5
CD16_Mono:PTID4
CD16_Mono:PTID2
CD16_Mono:PTID5
CD16_Mono:PTID6
cDC2:PTID4
cDC2:PTID5
cDC2:PTID2
cDC2:PTID6
pDC:PTID4
pDC:PTID2
pDC:PTID5
pDC:PTID6
HLA-A
HLA-B
HLA-C
HLA-DRA
HLA-DPA1
HLA-DRB1

3.3 Tutorial-3: scATAC Longitudinal data (n=4 and 6 weeks follow-up)

This tutorial allows to explore single cell ATACseq genscore data measured from 4 healthy donors over 6 time-points (week 2-7). Single cell ATAC data available at GSE190992*. (1) AIFI-scATAC-PBMC-FinalData.Rda (2) AIFI-Metadata.Rda (clinical metadata). Longitudinal dataset have 4 donors and 18 samples. To infer the variations at single cell ATAC please follow following steps. *scATAC genescore for longitudinal data was retrieved using ArchR (https://www.archrproject.com/) package. The genescore matrix was used as input for PALMO to explore longitudinal stability or variations across celltypes.*

(Note:If any issue with data access or needed RAW files please Contact Us).

3.3.1 Load Library

```
library("PALMO")
```

3.3.2 Load scATAC genescore data and assign paramaters (Time <30sec)

Load genescorematrix (pseudo-bulk) from ArchR or relevant tools (Aggregated peaks at gene level). We used ArchR to generate genescore matrix from scATAC peakmatrix of longitudinal data by each sample and celltype level.

```
#scATAC object
load("data/AIFI-scATAC-PBMC-FinalData.Rda")
#log-normalized
datamatrix <- log2(scatac_gm+1)

#Load annotation data
load("data/AIFI-Metadata.Rda")
```

3.3.3 Create PALMO object (Time <30sec)

First create the PALMO S4 object using input scATAC genescore data and annotation dataframe. The expression dataframe columns merged with input annotation dataframe. Only overlapping samples kept.

```
#Create PALMO object
palmo_obj <- createPALMOobject(anndata=ann, data=datamatrix)
```

Assign sample_column, donor_column and time_column variables in this step.

```
#Assign Sample, PTID and Time parameters
palmo_obj <- annotateMetadata(data_object=palmo_obj,
              sample_column= "Sample", donor_column= "PTID",
              time_column= "Time")
```

For aggregated genescore data merge data column with `Sample` annotation mentioned in annotation dataframe.

```
#Sample overlap and final matrix
palmo_obj <- mergePALMOdata(data_object=palmo_obj, datatype="singlecell")
```

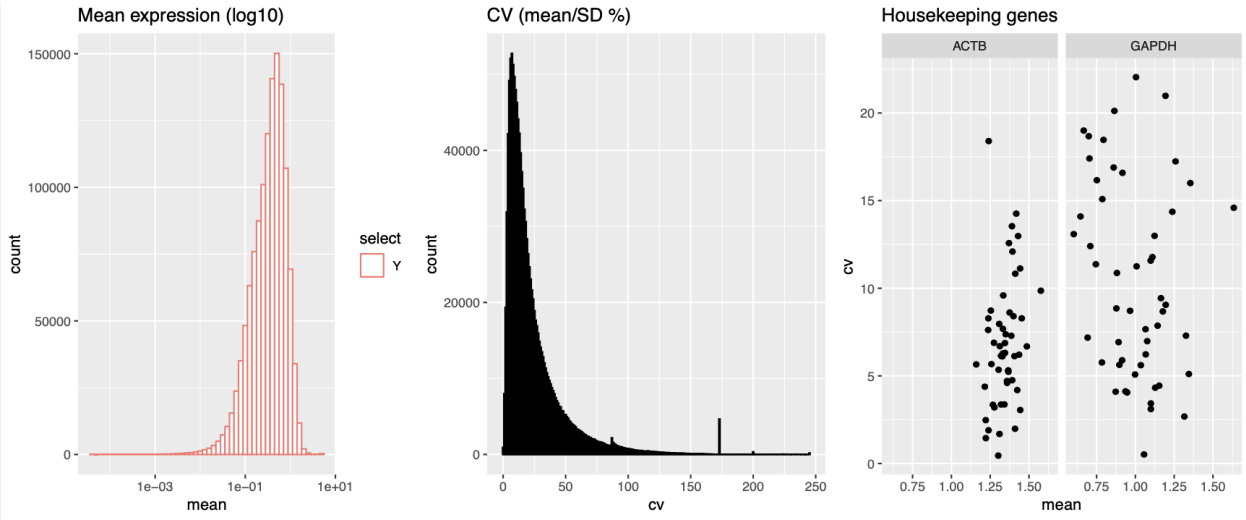
Optional step. If data consists of replicates then they can be merged by `mergeReplicates = TRUE`.

```
#Check for replicates
palmo_obj <- checkReplicates(data_object=palmo_obj, mergeReplicates = T)
```

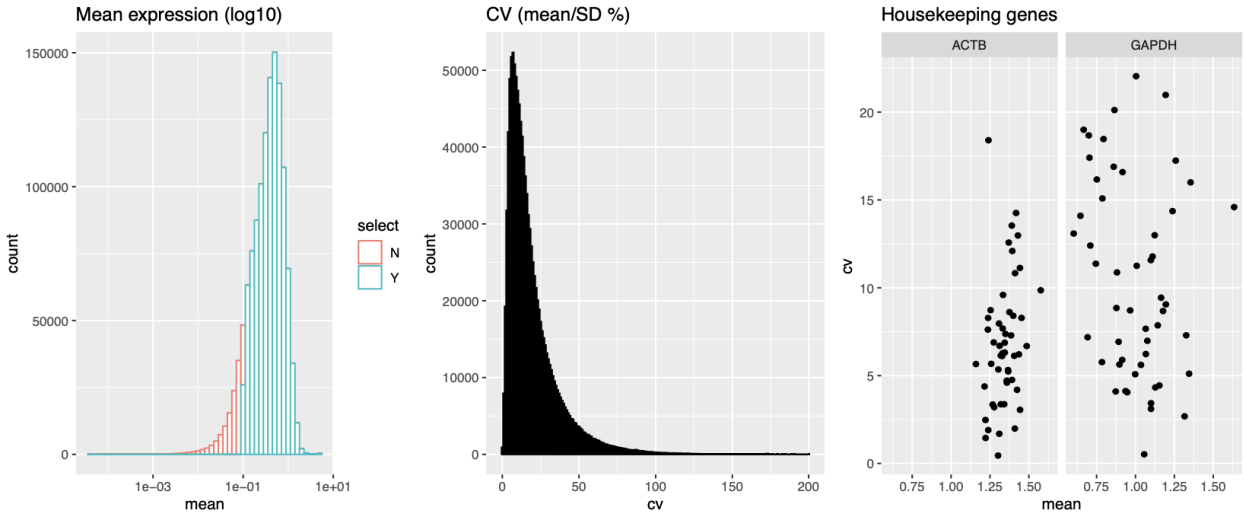
3.3.4 CV profile (Time ~2min)

To calculate longitudinal stability within donor first visualize the CV vs mean distribution for given data. Define CV threshold using the histogram or selecting few features of interest CV as cut-off. Here we used housekeeping genes GAPDH and ACTB to define mean and CV threshold.

```
palmo_obj <- cvCalcSCProfile(data_object=palmo_obj,
                             housekeeping_genes=c("GAPDH", "ACTB"),
                             fileName="scatac")
```




```
palmo_obj <- cvCalcSCProfile(data_object=palmo_obj,
                             housekeeping_genes=c("GAPDH", "ACTB"),
                             meanThreshold = 0.1,
                             fileName="scatac")
```



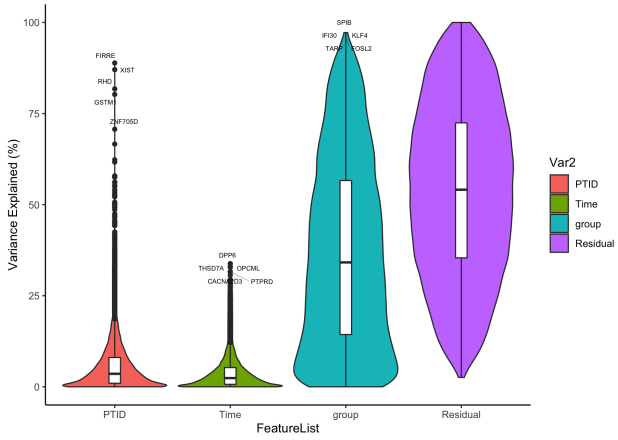
Optional step. To check the donor or participant wise CV profile run the CV sample profile analysis. Plots are saved in output directory if directory not mentioned.

```
#Sample Celltype Mean-CV plot
cvSCsampleprofile(data_object=palmo_obj, meanThreshold = 0.1,
                  cvThreshold = 10, fileName="scatac")
```

3.3.5 Features contributing towards donor variations (Time ~ 8min)

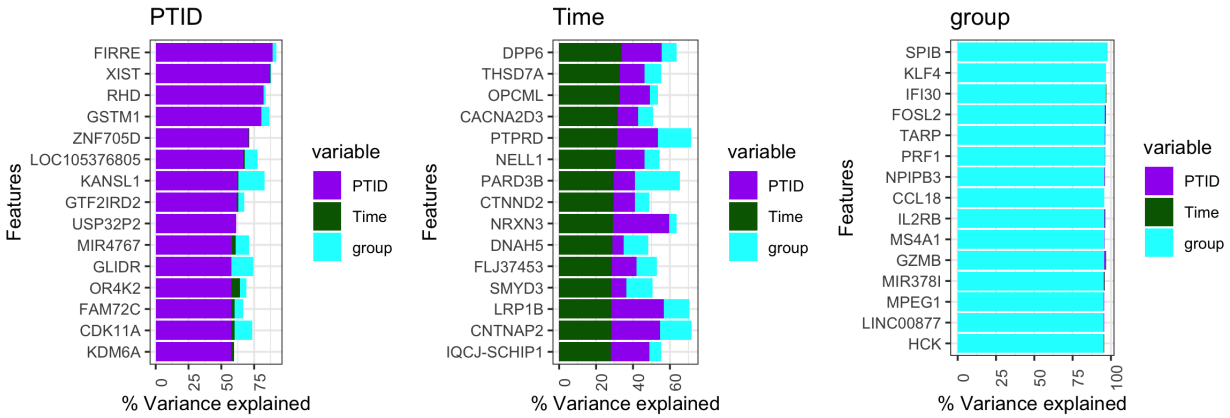
Variance decomposition analysis was performed to identify the features associated with attributes of interest such as PTID (participants) , Time (weeks), and group (celltype). The variance explained by each gene towards the `featureSet` of interest given in percentage. *(Note: To reduce the processing time use nodes cl=8 (or greater) and lmer_control=TRUE)*

```
#Variance decomposition
featureSet <- c("PTID","Time","group")
palmo_obj <- lmeVariance(data_object=palmo_obj,
                         featureSet=featureSet,
                         meanThreshold=0.1, cl=4,
                         fileName="scatac")
var_decomp <- palmo_obj@result$variance_decomposition
head(var_decomp[,featureSet])
#                  PTID      Time     group
#FIRRE         88.88239 0.2606612 2.8339971
#XIST          87.05064 0.3986387 0.5178851
#RHD           81.77230 0.3641998 1.4574140
#GSTM1         80.29766 0.1786628 6.2277645
#ZNF705D       70.69163 0.4342325 0.4752960
#LOC105376805 66.62977 1.2741294 9.8918653
```



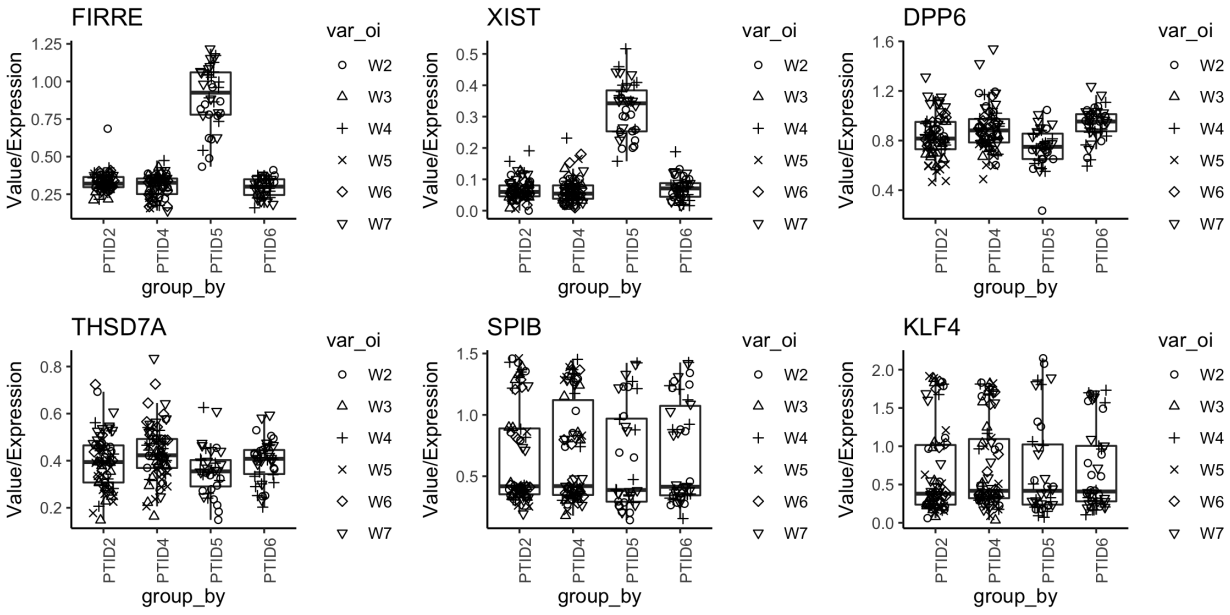
The top 15 features contributing to donor, time or group/celltype attributes variance can be visualized in barplot.

```
#Variance contributing features
plots <- variancefeaturePlot(vardata=var_decomp, featureSet=featureSet,
                             cols=c("purple", "darkgreen", "cyan"),
                             ncol=3)
```



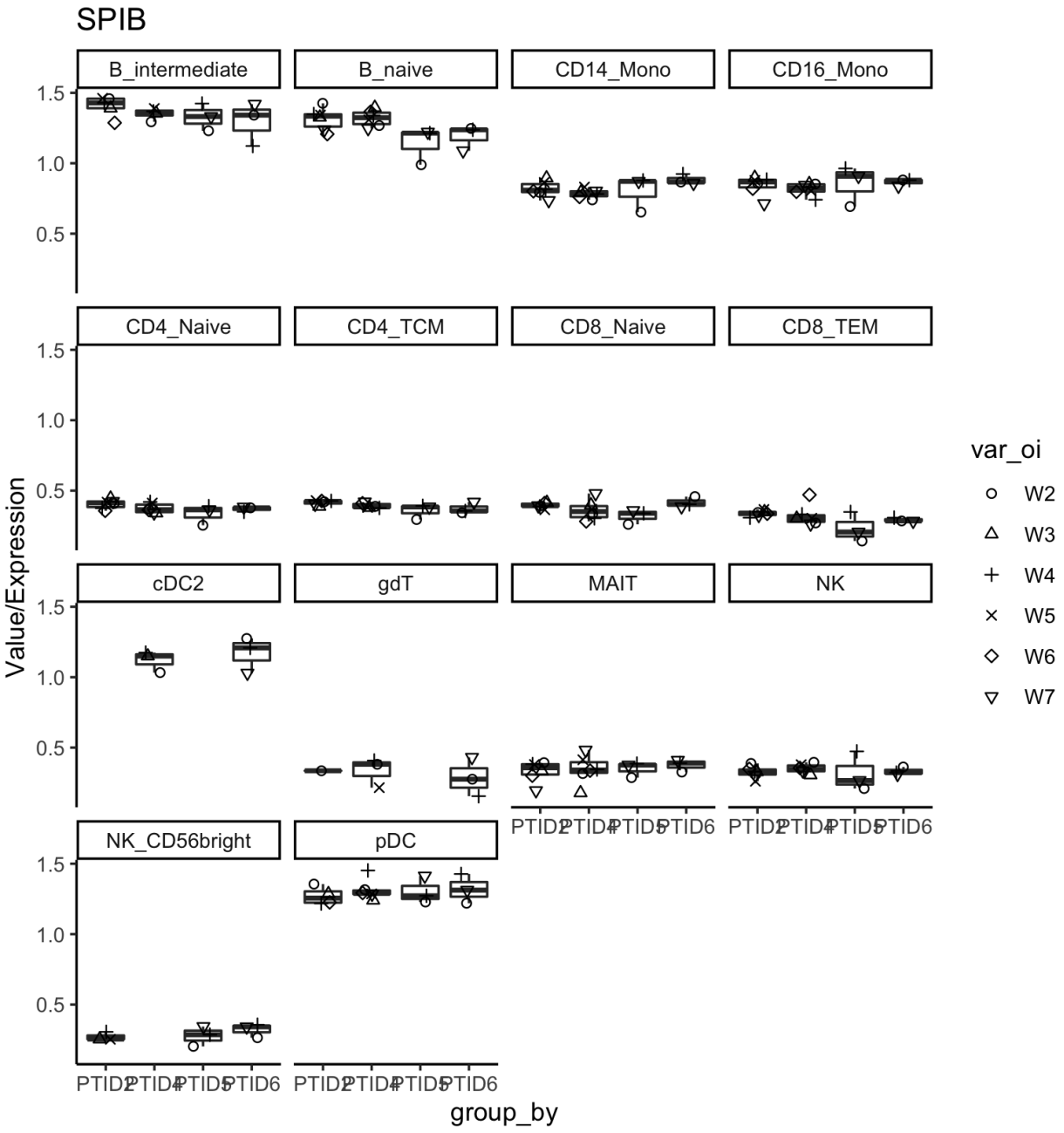
Gene expression plot for interested features can be visualized by donors or by time.

```
#Top genes
plots <- gene_featureplot(data_object=palmo_obj,
                          featureList=c("FIRRE", "XIST",
                                        "DPP6", "THSD7A",
                                        "SPIB", "KLF4"),
                          x_group_by="PTID", var_oi="Time",
                          x_text_angle=90)
```



Gene expression plot for interested features can be visualized by group (celltypes).

```
gene_featureplot(data_object=palmo_obj, featureList="SPIB", facet_by="group")
```

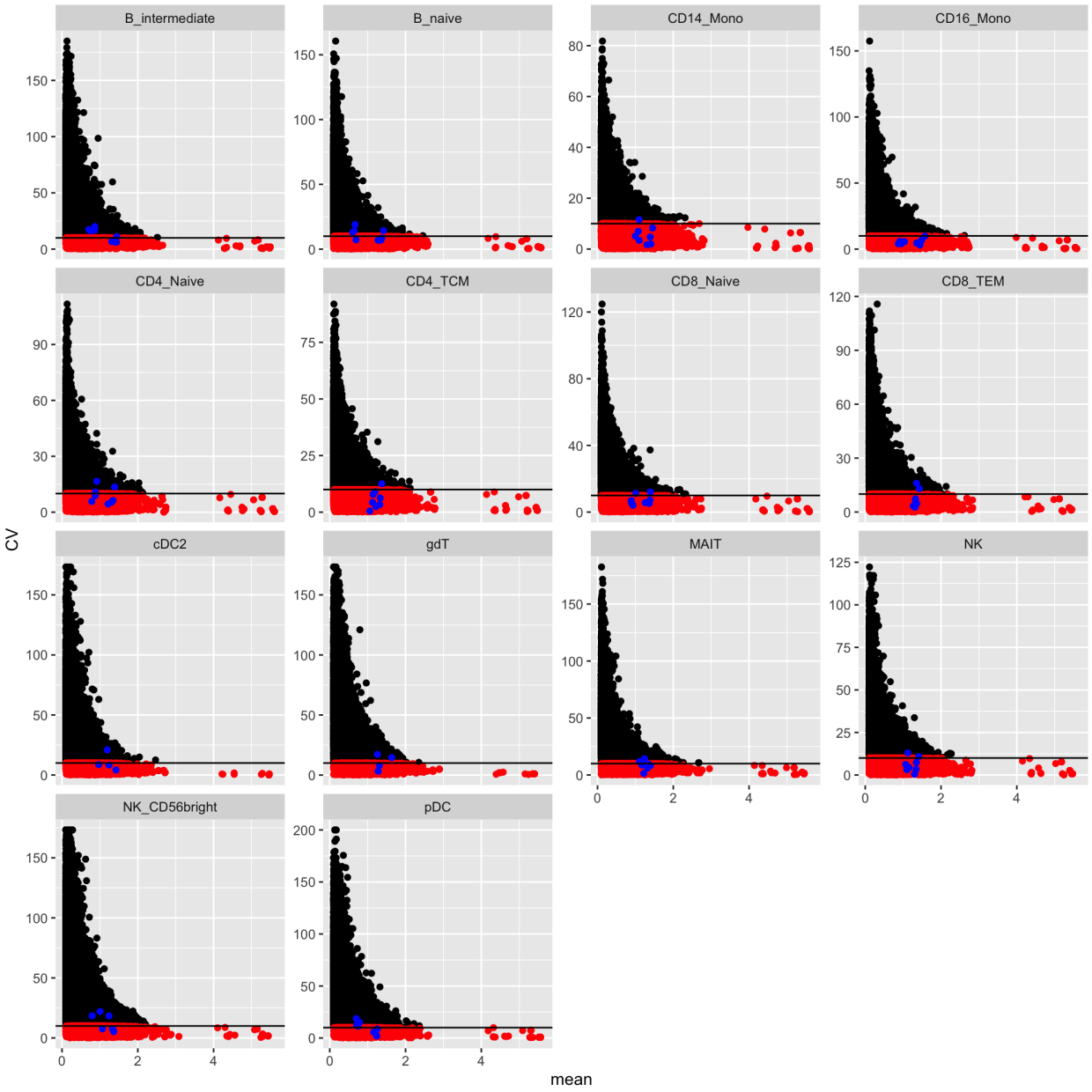


3.3.6 Intra-donor variations over time (Time ~ 5min)

To calculate longitudinal stability within donor define meanThreshold and cvThreshold parameters as discussed above. The analysis will calculate the CV across average group ("celltype") mentioned by user. It will create a CV-Mean plots for individual donor over longitudinal timepoints that shows the highly variable and stable features in each donor. The plots are stored in output directory if not mentioned. It also provides a housekeeping genes position in CV-Mean plot across each celltype/group of interest. *(Note: To reduce the processing time use nodes cl=8 (or greater))*

```
palmo_obj <- cvCalcSC(data_object=palmo_obj,
                  meanThreshold=0.1, cvThreshold=10,
```

```
housekeeping_genes=c("GAPDH", "ACTB"),
fileName="scatac")
```

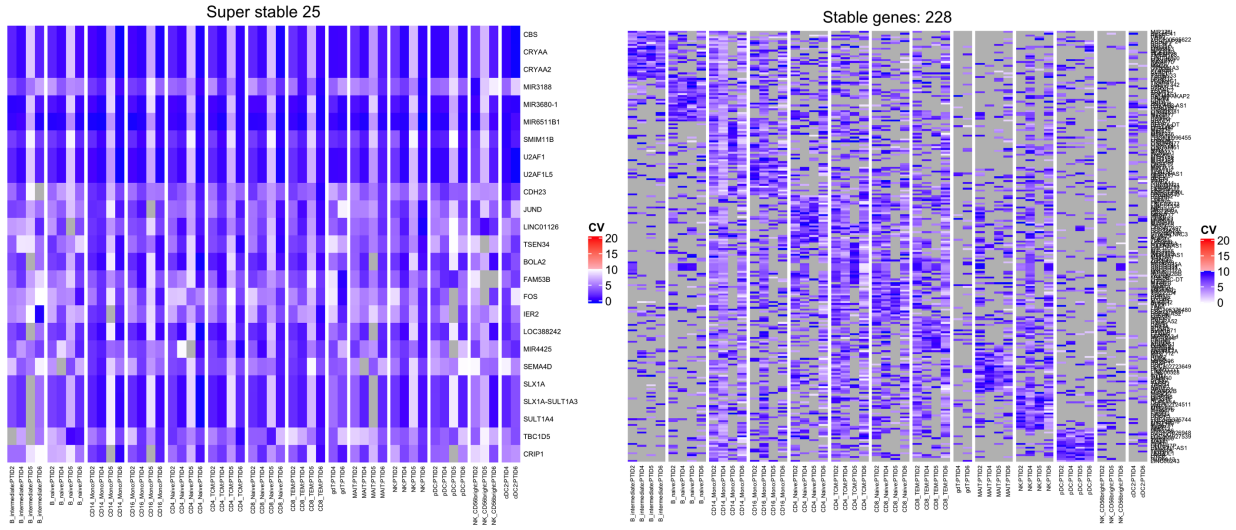


(Note: Black dots, all genes; red dots, genes with given mean and CV threshold; blue dots, housekeeping genes)

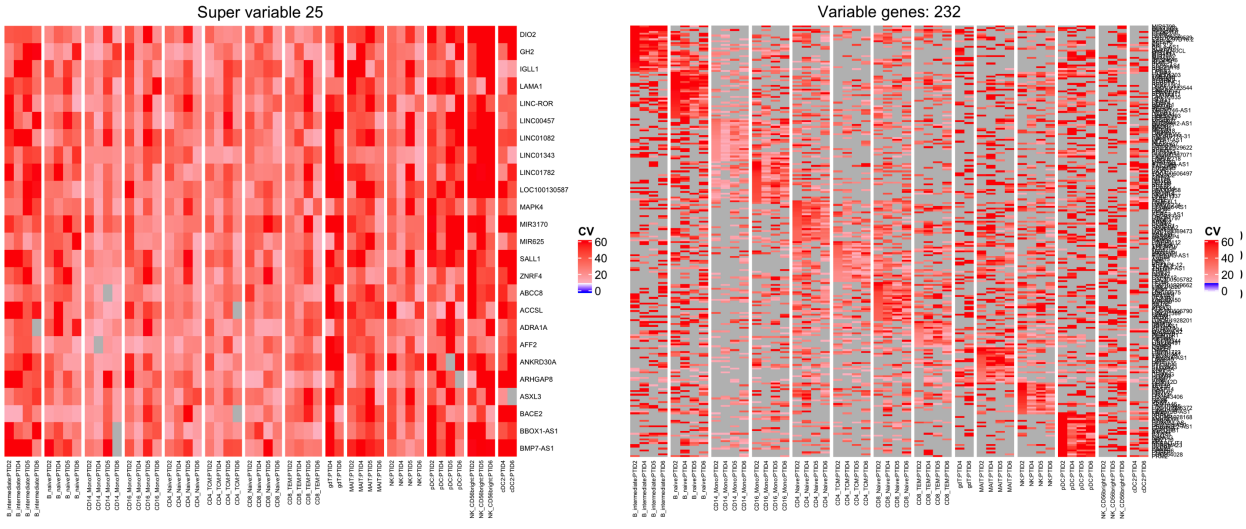
3.3.7 Find stable and variable features in longitudinal data (Time 30sec)

To identify Stable and variable features across celltype or group of interest CV values for each donor at celltype retrieved from above step. Minimum number of donors (donorThreshold) and group threshold (groupThreshold) defined to retrieve features with consensus stable/variable profile. Here, donorThreshold=4 and groupThreshold=28 (4*14/2) is selected.

```
#Find stable and variable features in longitudinal data
palmo_obj <- StableFeatures(data_object=palmo_obj,
                            cvThreshold=10,
                            donorThreshold=4,
                            topFeatures=25,
                            fileName="scatac")
stable_genes <- palmo_obj@result$stable_genes
```




```
palmo_obj <- VarFeatures(data_object=palmo_obj,
                         cvThreshold=10,
                         donorThreshold=4, groupThreshold=28,
                         topFeatures=25,
                         fileName="scatac")
var_genes <- palmo_obj@result$var_genes
```



To visualize how these stable or variable features performing, users can reduce high dimensional single cell ATAC data into lower dimensions using selected Top features from stable/variable analysis. The UMAP plot shows whether identified stable features can reproduce celltype/group identity based on few but stable features.

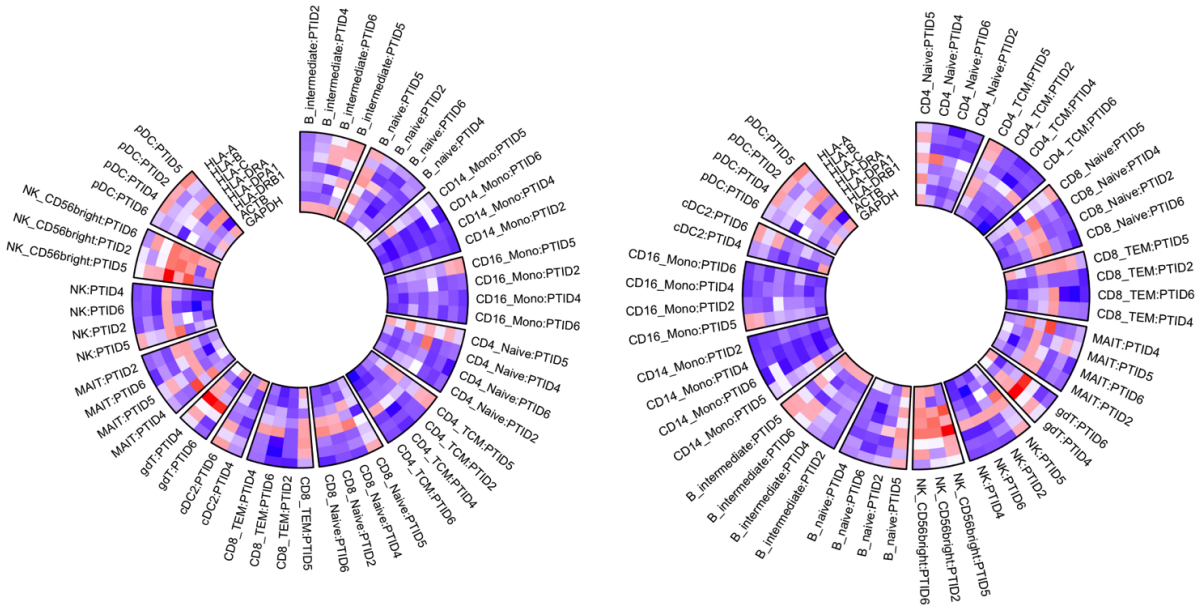
3.3.8 Circos CV plot (Time ~ 10sec)

The top features or gene families can be visualized in circos plot for easy comparison.

```
geneList <- c("HLA-A","HLA-B","HLA-C","HLA-DRA","HLA-DPA1","HLA-DRB1",
              "ACTB","GAPDH")
plotmatrix <- genecircosPlot(data_object=palmo_obj, geneList=geneList,
                             colorThreshold=15)
```

To order the circos plot by user-defined group order follow following steps.

```
#order by user-defined group order
celltype_oi <- c("CD4_Naive","CD4_TEM","CD4_TCM","CD4_CTL",
              "CD8_Naive","CD8_TEM","CD8_TCM","Treg","MAIT","gdT",
              "NK", "NK_CD56bright",
              "B_naive","B_memory", "B_intermediate",
              "CD14_Mono","CD16_Mono",
              "cDC2","pDC")
plotmatrix <- genecircosPlot(data_object=palmo_obj, geneList=geneList,
                             group_oi=celltype_oi, colorThreshold=15)
```



3.4 Tutorial-4: Multi-modal data integration/vizualization

This tutorial allows users to combine intra-donor variation value between different modalities like scRNA and scATAC data here. Load CV result from scRNA and scATAC as described above (check output directory for the output files). To integrate variability across modalities, please follow following steps.

3.4.1 Load Library

```
library("PALMO")
```

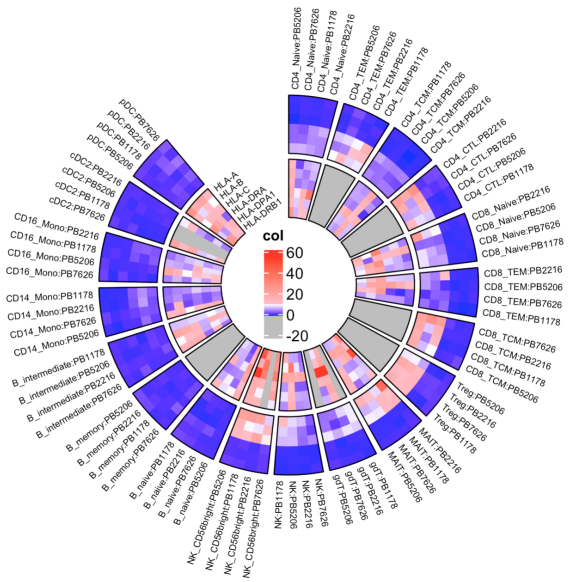
3.4.2 Load data from output folder (previous steps result)

```
#From scRNA analysis obtain the CV data
load("output/scrna-CV-allgenes-raw.Rda")
scrna_cv_res <- cv_res

#From scATAC analysis obtain the CV data
load("output/scatac-CV-allgenes-raw.Rda")
scatac_cv_res <- cv_res

#Cell type of interest
celltype_oi <- c("CD4_Naive","CD4_TEM","CD4_TCM","CD4_CTL",
                 "CD8_Naive","CD8_TEM","CD8_TCM","Treg",
                 "MAIT","gdT", "NK", "NK_CD56bright",
                 "B_naive", "B_memory", "B_intermediate",
                 "CD14_Mono","CD16_Mono",
                 "cDC2","pDC")

#HLAs
geneList <- c("HLA-A","HLA-B","HLA-C","HLA-DRA","HLA-DPA1","HLA-DRB1")
```

3.4.3 Run (Time ~ 10sec)

The circos plot shows integrated view of stable and variable features CVs across celltypes.

```
plot <- multimodalView(modality1=scrna_cv_res,
               modality2=scatac_cv_res,
               geneList=geneList, group_oi=celltype_oi,
               colorscale=TRUE)
```


CD4_Naive:PB5206
CD4_Naive:PB7626
CD4_Naive:PB1178
CD4_Naive:PB2216
CD4_TEM:PB5206
CD4_TEM:PB7626
CD4_TEM:PB2216
CD4_TEM:PB1178
CD4_TCM:PB1178
CD4_TCM:PB7626
CD4_TCM:PB5206
CD4_TCM:PB2216
CD4_CTL:PB2216
CD4_CTL:PB7626
CD4_CTL:PB5206
CD4_CTL:PB1178
CD8_Naive:PB2216
CD8_Naive:PB5206
CD8_Naive:PB7626
CD8_Naive:PB1178
CD8_TEM:PB2216
CD8_TEM:PB5206
CD8_TEM:PB7626
CD8_TEM:PB1178
CD8_TCM:PB7626
CD8_TCM:PB2216
CD8_TCM:PB1178
CD8_TCM:PB5206
Treg:PB5206
Treg:PB2216
Treg:PB7626
Treg:PB1178
MAIT:PB2216
MAIT:PB1178
MAIT:PB7626
MAIT:PB5206
gdT:PB1178
gdT:PB2216
gdT:PB7626
gdT:PB5206
NK:PB7626
NK:PB5206
NK:PB2216
NK:PB1178
NK_CD56bright:PB7626
NK_CD56bright:PB2216
NK_CD56bright:PB1178
NK_CD56bright:PB5206
B_naive:PB5206
B_naive:PB7626
B_naive:PB2216
B_naive:PB1178
B_memory:PB1178
B_memory:PB7626
B_memory:PB2216
B_memory:PB5206
B_intermediate:PB7626
B_intermediate:PB2216
B_intermediate:PB5206
B_intermediate:PB1178
CD14_Mono:PB5206
CD14_Mono:PB7626
CD14_Mono:PB2216
CD14_Mono:PB1178
CD16_Mono:PB7626
CD16_Mono:PB5206
CD16_Mono:PB1178
CD16_Mono:PB2216
cDC2:PB7626
cDC2:PB1178
cDC2:PB5206
cDC2:PB2216
pDC:PB5206
pDC:PB1178
pDC:PB2216
pDC:PB7626
HLA-A
HLA-B
HLA-C
HLA-DRA
HLA-DPA1
HLA-DRB1
col
60
40
20
0
-20

3.5 Tutorial-5: COVID19 longitudial dataset (CNP0001102)

This tutorial allows to explore single cell RNAseq data variability across COVID and FLU donors. PBMCs from the healthy, COVID and FLU donors were collected longitudinally. Single cell data from Zhu et al. 2020 downloaded from CNP0001102. Metadata is downloaded from Supplementary table and curated version can be found in the metadata. To infer variability (inter- and Intra-) and identify stable genes, please follow following steps.

3.5.1 Load Library

```
#Load Library
library("PALMO")
```

3.5.2 Load data and assign paramaters (Time < 1min)

Load single cell data and define the Sample column which is same as Sample column in metadata.

```
#Load scRNA data
pbmc <- readRDS("data/CNP0001102_Final_nCoV_0716_upload.RDS")
#Add column Sample
pbmc@meta.data$Sample <- pbmc@meta.data$batch
#check celltypes
sort(unique(pbmc@meta.data$cell_type))
#[1] Cytotoxic CD8 T cells Naive T cells         NKs
#[4] MAIT                  Activated CD4 T cells Naive B cells
#[7] Plasma                Memory B cells        XCL+ NKs
#[10] Cycling T cells       Monocytes             DCs
#[13] Cycling Plasma        Stem cells            Megakaryocytes

#Clinical annotations Table S1. Clinical data of the enrolled subjects
metadata <- read.csv("data/CNP0001102-annotation.csv", stringsAsFactors = F)

#Exploring only COVID samples
metadata <- metadata[metadata$Participant %in% c("COV-1", "COV-2", "COV-3", "COV-4", "COV-5"),]
#Exploring only FLU samples
#metadata <- metadata[metadata$Participant %in% c("IAV-1","IAV-2"),]
```

3.5.3 Create PALMO object (Time < 1min)

First create the PALMO S4 object using input scRNA object and annotation dataframe. The expression dataframe columns merged with input annotation dataframe. Only overlapping samples kept.

```
#Create PALMO object
palmo_obj <- createPALMOobject(anndata=metadata, data=pbmc)

#Assign Sample, PTID and Time parameters
palmo_obj <- annotateMetadata(data_object=palmo_obj,
                              sample_column= "Sample",
                              donor_column= "Participant",
                              time_column= "Day")
```

Assign sample_column, donor_column and time_column variables in this step.

```
#Sample overlap and final matrix
palmo_obj <- mergePALMOdata(data_object=palmo_obj, datatype="singlecell")
```

Single cell data is aggregated by average method at sample group level. Features with average expression greater than zero across all samples are kept.

```
#Aggregate data (Psuedo-bulk)
palmo_obj <- avgExpCalc(data_object=palmo_obj, assay="RNA",
                        group_column="cell_type")
head(palmo_obj@curated[["anndata"]]) #merged annotation data
head(palmo_obj@curated[["data"]]) #scRNA average expression data
```

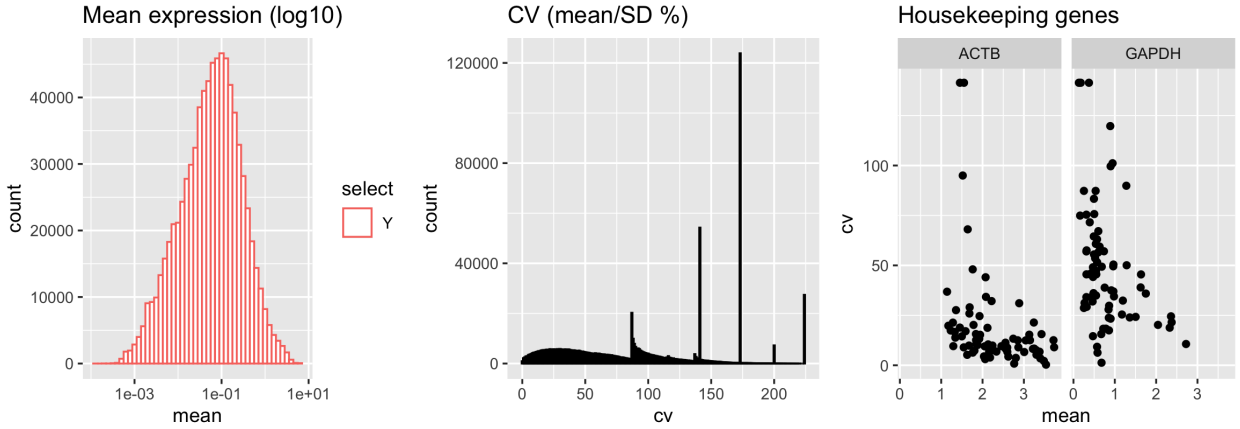
Optional step. If data consists of replicates can be merged by mergeReplicates = TRUE.

```
#Check for replicates
palmo_obj <- checkReplicates(data_object=palmo_obj, mergeReplicates = T)
```

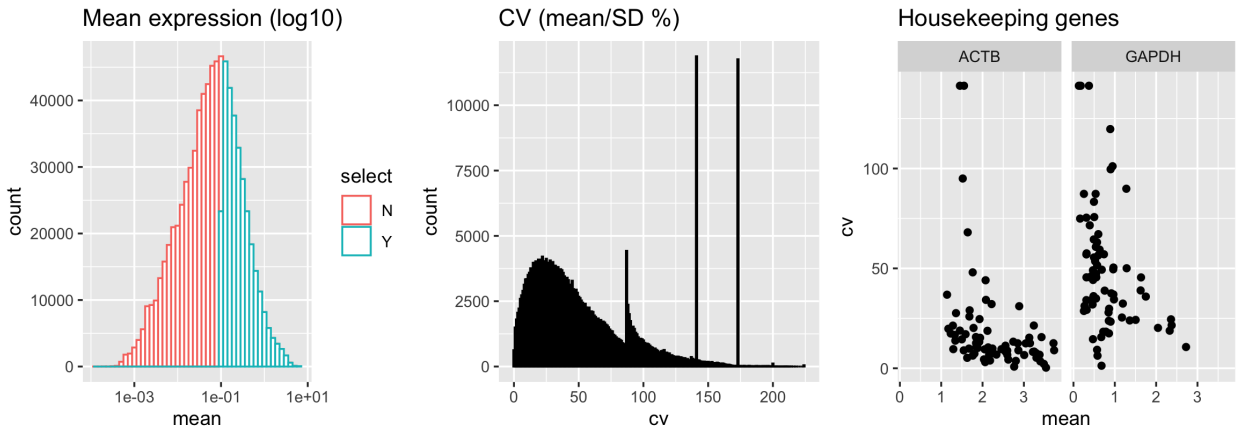
3.5.4 CV profile (Time ~ 1min)

To calculate longitudinal stability within donor first visualize the CV vs mean distribution for given data. Define CV threshold using the histogram or selecting few features of interest CV as cut-off. Here we used housekeeping genes GAPDH and ACTB to define CV threshold.

```
palmo_obj <- cvCalcSCProfile(data_object=palmo_obj,
                        housekeeping_genes=c("GAPDH", "ACTB"),
                        fileName="CNP0001102")
```




```
#Sample Celltype Mean-CV plot (output directory)
palmo_obj <- cvCalcSCProfile(data_object=palmo_obj,
                        housekeeping_genes=c("GAPDH", "ACTB"),
                        meanThreshold = 0.1,
                        fileName="CNP0001102")
```



Optional step. To check the donor or participant wise CV profile run the CV sample profile analysis.

```
cvSCsampleprofile(data_object=palmo_obj,
                  meanThreshold = 0.1, plot_log10=T,
                  cvThreshold = 25)
```

3.5.5 Features contributing towards donor variations (Time ~ 5min)

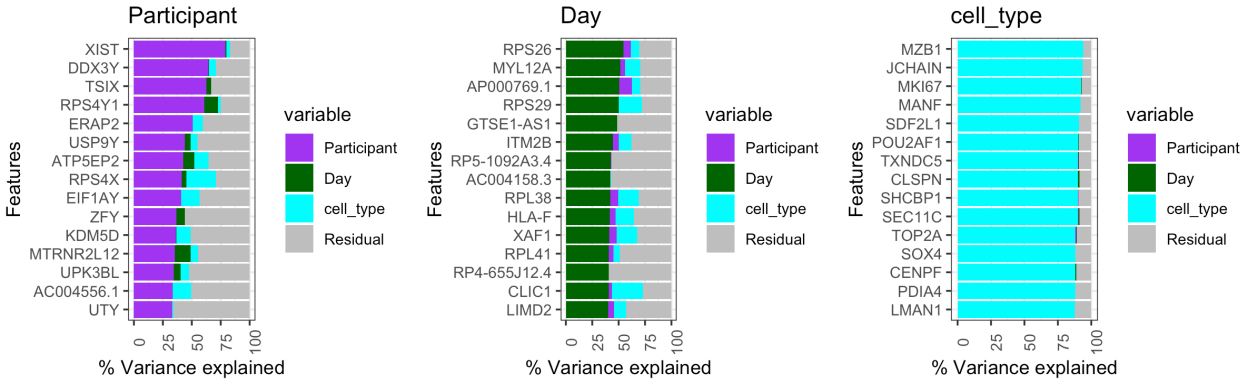
Variance decomposition analysis was performed to identify the features associated with attributes of interest such as participants, sex, disease type, celltype, or batch. The `featureSet` is a list of variables for which fraction of variance explained by each gene is calculated. The variance explained by each gene towards the `featureSet` of interest given in percentage. *(Note: To reduce the processing time use nodes cl=8 (or greater) and lmer_control=TRUE)*

```
#Check the group of interest
head(palmo_obj@curated$anndata)

#Variance decomposition
featureSet <- c("Participant", "Day","cell_type")
palmo_obj <- lmeVariance(data_object=palmo_obj,
                         featureSet=featureSet,
                         meanThreshold=0.1, cl=4,
                         fileName="CNP0001102")
var_decomp <- palmo_obj@result$variance_decomposition
head(var_decomp[,featureSet])
       Participant       Day cell_type
#XIST      78.82243  0.936695  3.510383
#DDX3Y     64.44897  0.637655  5.824030
#TSIX      62.57711  3.997646  0.339527
#RPS4Y1    60.73146 11.768212  2.346928
#ERAP2     50.92503  0.000000  8.745452
#USP9Y     44.21867  4.657177  5.997529
```

The top 15 features contributing to donor, time or celltype attributes variance can be seen in barplot.

```
#Variance explained (Participant, Day, cell_type)
plots <- variancefeaturePlot(vardata=var_decomp,
                         featureSet=featureSet,
```

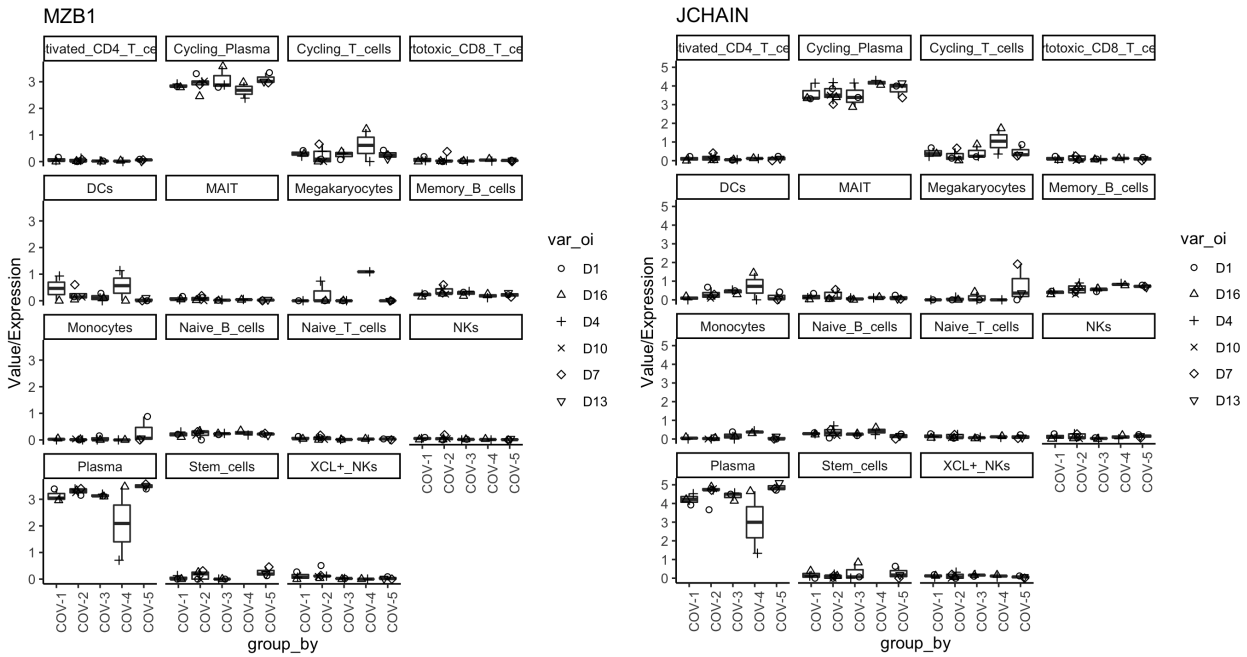
```
                    Residual=T,
                    cols=c("purple", "darkgreen", "cyan"),
                    ncol=3)
```



3.5.6 Plot the variables (Time ~ 10sec)

Gene expression plot for top features by cell_type can be visualized by donors or by time.

```
plots <- gene_featureplot(data_object=palmo_obj,
                          featureList=c("MZB1", "JCHAIN"),
                          facet_by="cell_type", x_text_angle=90)
```

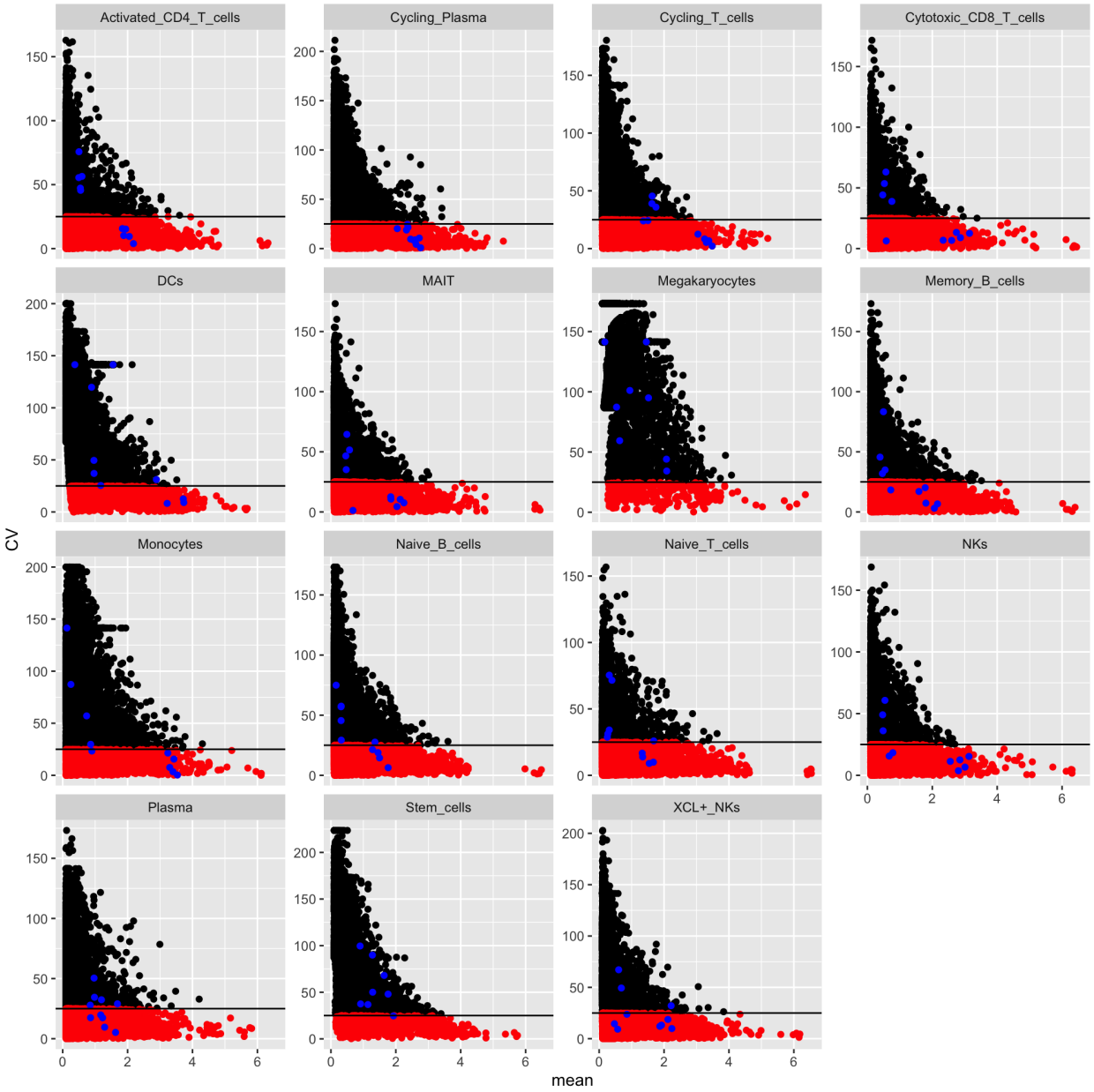


3.5.7 Intra-donor variations over time (Time ~ 4min)

To calculate longitudinal stability within donor define meanThreshold and cvThreshold parameters as discussed above. The analysis will calculate the CV across average group ("cell_type"). It will create a CV-Mean plots for individual donor over longitudinal timepoints that shows the highly variable and stable

features in each donor. The plots are stored in output directory. It also provides a housekeeping genes position in CV-Mean plot across each celltype/group of interest.

```
#Calculate CV
palmo_obj <- cvCalcSC(data_object=palmo_obj,
                      meanThreshold=0.1, cvThreshold=25,
                      housekeeping_genes=c("GAPDH", "ACTB"),
                      fileName="CNP0001102")
```

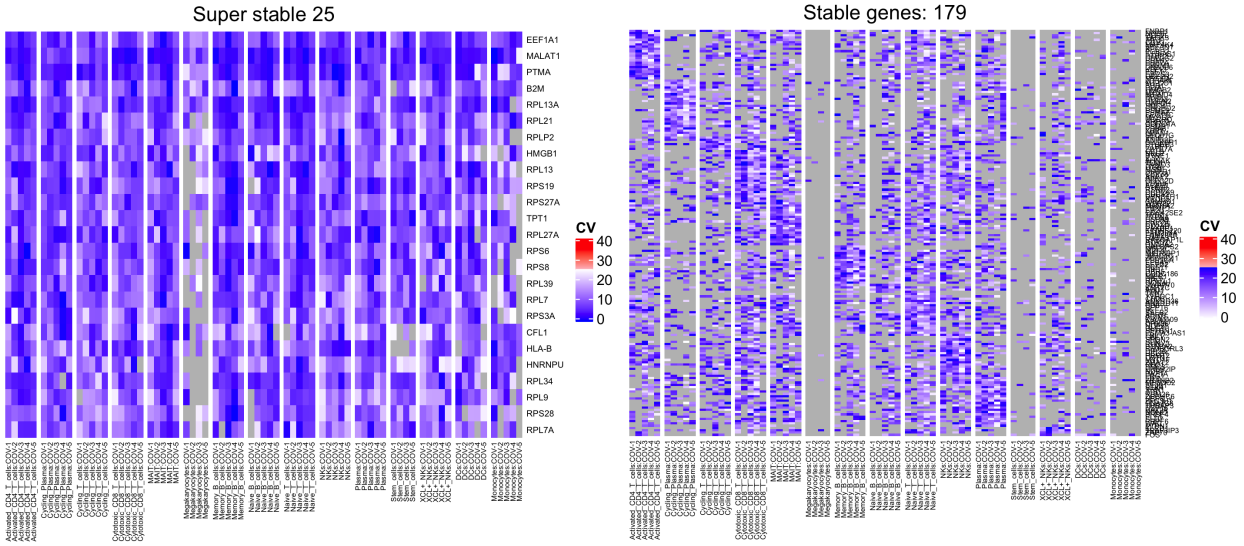


(Note: Black dots, all genes; red dots, genes with given mean and CV threshold; blue dots, housekeeping genes)

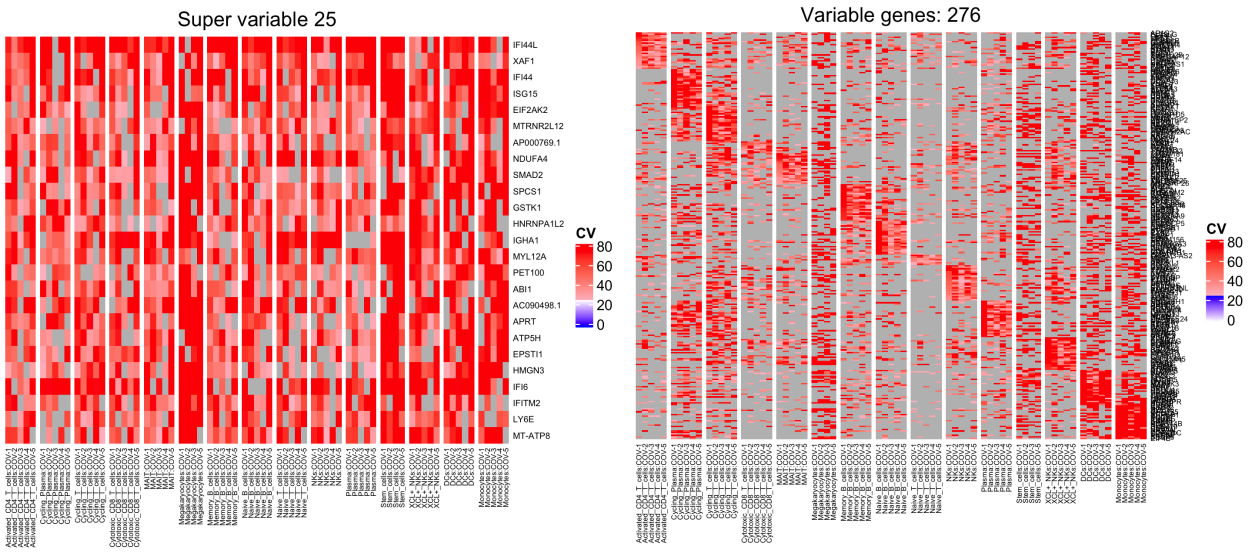
3.5.8 Find stable and variable features in longitudinal data (Time ~ 10sec)

Above step resulted CV values used to identify Stable and variable features across cell_type. Minimum number of donors (`donorThreshold`) considered 5 to have consensus stable/variable profile over timepoints. `groupThreshold` considered 38 (number of donors * number of celltypes/2 = 5x15/2 ~ 37.5).

```
palmo_obj <- StableFeatures(data_object=palmo_obj,
                            cvThreshold=25,
                            donorThreshold=5,
                            topFeatures=25,
                            fileName="CNP0001102")
stable_genes <- palmo_obj@result$stable_genes
```



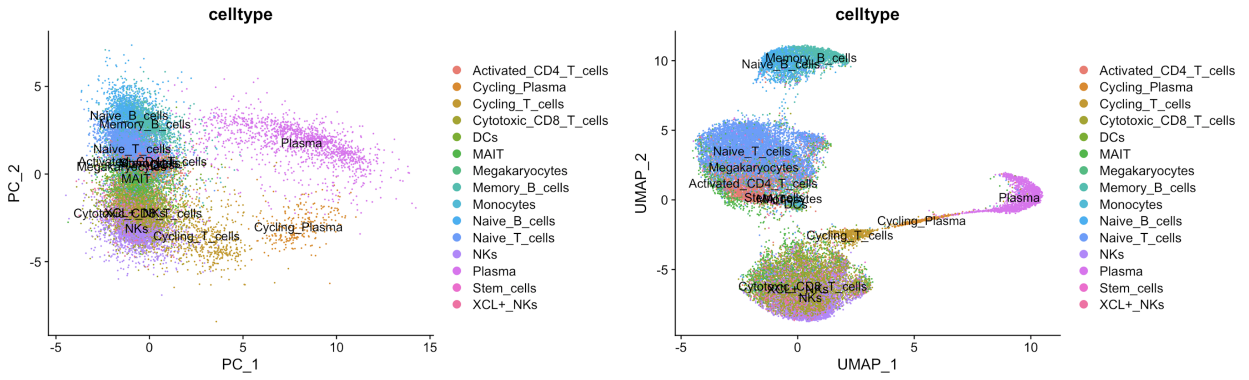

```
palmo_obj <- VarFeatures(data_object=palmo_obj,
                         cvThreshold=25,
                         donorThreshold=5,
                         topFeatures=25,
                         fileName="CNP0001102")
var_genes <- palmo_obj@result$var_genes
```



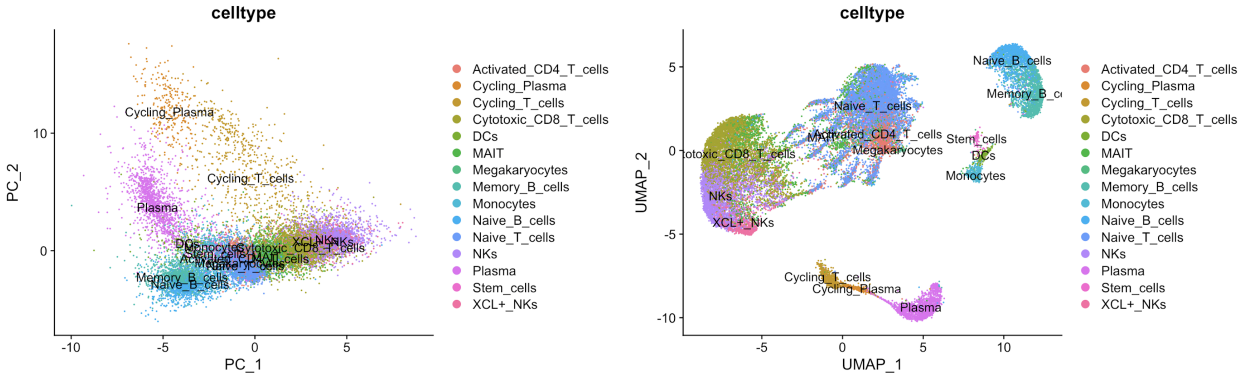
3.5.9 UMAP Plot (Time ~ 2min)

To visualize how these stable or variable features performing, users can reduce high dimensional single cell data into lower dimensions using selected Top features from stable/variable analysis. The UMAP plot shows whether identified stable features can reproduce celltype/group identity based on few but stable features.

```
#Stable genes UMAP
dimUMAPPlot(data_object=palmo_obj, nPC=15,
            gene_oi=unique(stable_genes$gene),
            group_column="cell_type", plotname="stable",
            fileName="CNP0001102")
```



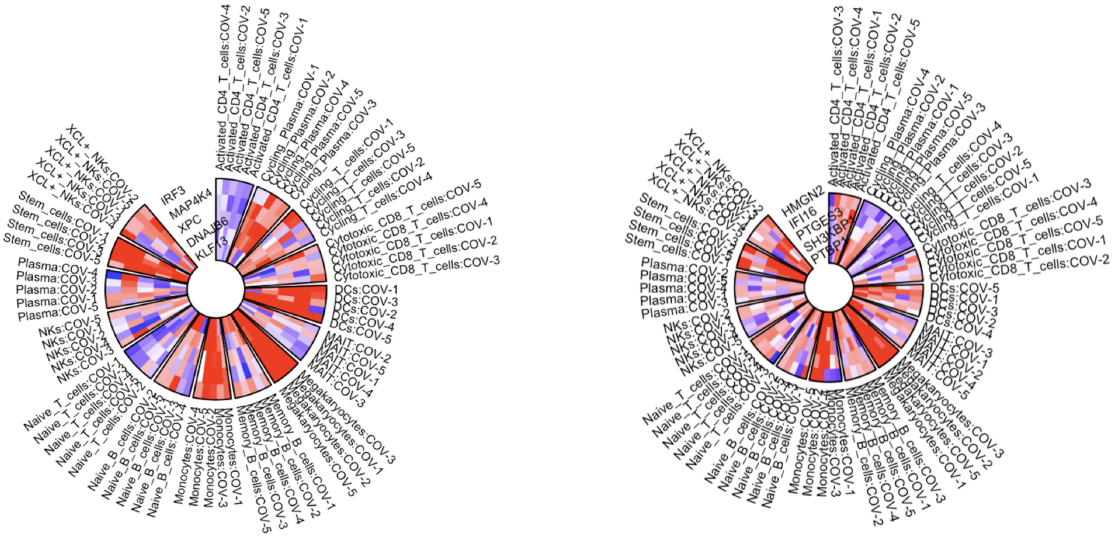

```
#Variable genes UMAP
dimUMAPPlot(data_object=palmo_obj, nPC=15,
            gene_oi=unique(var_genes$gene),
            group_column="cell_type", plotname="variable",
            fileName="CNP0001102")
```



3.5.10 Celltype-specific Circos CV Plot (Time ~ 30sec)

The top features from Activated CD4 T-cells and Cycling T-cells can be visualized in circos plot for easy comparison.

```
#Activated CD4 T-cells
geneList <- c("IRF3","MAP4K4","XPC","DNAJB6", "KLF13")
plotres <- genecircosPlot(data_object=palmo_obj,
                          geneList=geneList, colorThreshold=25)

#Cycling T-cells
geneList <- c("HMGN2", "IFI16", "PTGES3", "SH3KBP1", "PTBP1")
plotres <- genecircosPlot(data_object=palmo_obj,
                          geneList=geneList, colorThreshold=25,
                          colorscale = TRUE)
```



3.6 Tutorial-6: Differential Gene analysis in longitudinal data (CNP0001102)

This tutorial allows users to identify differential expressed genes in direction of longitudinal time-points. As an example single cell data from Zhu et al. 2020 downloaded from CNP0001102. Metadata is downloaded from Supplementary table and curated version can be found in the metadata. The dataset consists of 5 Covid-19 donors, 2 Flu donors with longitudinal data and 3 controls. To explore differential expressed genes in each celltype of each donor we used hurdle model-based modeling on input data to retrieve the DEGs. To infer DEGs in each celltype over time progression (3 or more timepoints) timepoints were considered as continuous variable (otherwise discrete). To infer differential genes over time in COVID dataset please follow following steps.

3.6.1 Load data and clinical metadata (Time ~ 30sec)

Single cell Seurat object from study CNP0001102.

```
#Single cell object CNP0001102
pbmc <- readRDS("data/CNP0001102_Final_nCoV_0716_upload.RDS")
#Add column Sample
pbmc@meta.data$Sample <- pbmc@meta.data$batch
```

3.6.2 Clinical annotations Table S1. Clinical data of the enrolled subjects (Time ~ 5sec)

```
metadata <- read.csv("data/CNP0001102-annotation.csv", stringsAsFactors = F)
```

3.6.3 Load library and run (Time ~15 min)

```
#Load Library
library("PALMO")
library("tidyverse")
```

First create the PALMO S4 object using input scRNA object and annotation dataframe (metadata). The expression dataframe columns merged with input annotation dataframe. Only overlapping samples kept.

```
#Create PALMO object
palmo_obj <- createPALMOobject(anndata=metadata, data=pbmc)
```

Assign sample_column, donor_column and time_column variables in this step.

```
#Assign Sample, PTID and Time parameters
palmo_obj <- annotateMetadata(data_object=palmo_obj,
                              sample_column= "Sample",
                              donor_column= "Participant",
                              time_column= "Day")
```

For single cell data merge annotation and single cell metadata by mentioned sample_column.

```
#Sample overlap and final matrix
palmo_obj <- mergePALMOdata(data_object=palmo_obj, datatype="singlecell")
```

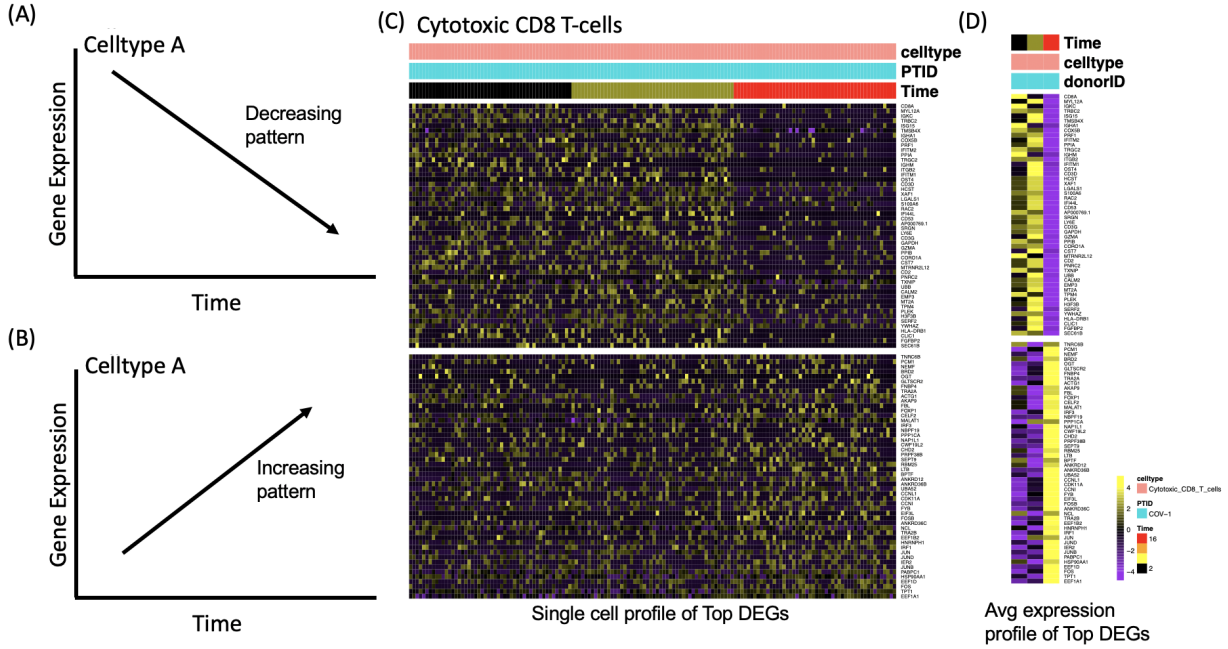
Perform longitudinal differential analysis in each donor and celltype over timepoints.

```
#Perform longitudinal differential analysis
palmo_obj <- sclongitudinalDEG(data_object=palmo_obj, scassay="RNA",
                               group_column="cell_type")
>Fitting a ZLM model for donorID:  COV-1 ...
>Fitting a ZLM model for donorID:  COV-2 ...
>Fitting a ZLM model for donorID:  COV-3 ...
>Fitting a ZLM model for donorID:  COV-4 ...
>Fitting a ZLM model for donorID:  COV-5 ...
>Fitting a ZLM model for donorID:  IAV-1 ...
>Fitting a ZLM model for donorID:  IAV-2 ...
```

Check output folder for tabular results and visualization. The `coef` is equivalent to `log2FC`, `nomP` suggest nominal p-value and `adjP` suggests the adjusted p-value.

```
#Plots can be seen in output directory output
DEGres <- palmo_obj@result$degs
head(DEGres[order(DEGres$coef, decreasing = T),])
#primerid contrast         nomP     coef         adjP donorID celltype dir
#IGHG4   TimeD9 1.701579e-26 3.056092 1.453999e-23   IAV-2   Plasma upregulated at D9
#JCHAIN   TimeD9 8.759407e-32 2.647757 2.245474e-28   IAV-2   Plasma upregulated at D9
#IGHG3   TimeD9 8.470810e-22 2.485250 2.412769e-19   IAV-2   Plasma upregulated at D9
#IGLC2   TimeD9 1.352490e-16 2.289544 8.890022e-15   IAV-2   Plasma upregulated at D9
#IGHG1   TimeD9 1.292885e-16 2.219146 8.608601e-15   IAV-2   Plasma upregulated at D9
#SYNE2   TimeD4 7.249010e-13 2.209541 1.215659e-09   COV-4 XCL+_NKs upregulated at D4

#Or interested celltypes
celltype_oi <- c("Activated CD4 T cells", "Naive B cells")
palmo_obj <- sclongitudinalDEG(data_object=palmo_obj, scassay="RNA",
                               group_column="cell_type",
                               group_oi = celltype_oi)
```

General analysis schema and differential results in each donor over timepoints in celltype Cytotoxic CD8 T-cells using `PALMO` shown below.



3.7 Tutorial-7: Mouse brain dataset (GSE129788)

This tutorial allows users to explore single cell RNAseq data from Mouse brain to show the application of PALMO on tissue samples. Ximerakis et al (2019) study was used out to explore the transcriptomic difference in aging brain. In this case study we used PALMO to identify stable features associated with brain celltypes across aging brain samples. The dataset includes total of 16 mice brains samples (8 young and 8 old) with 37,089 single cells GSE129788. The metadata file for the samples can be obtained from here. To infer variability (inter- and Intra-) and identify stable genes, please follow following steps.

3.7.1 Load Library

```
#Load Library
library("PALMO")
```

3.7.2 Load scRNA data and assign paramaters (Time ~ 30sec)

Load scRNA Seurat object.

```
#Load scRNA data
mbrain <- readRDS("data/GSE129788_seurat.RDS")
metaDF <- mbrain@meta.data
#check celltypes
sort(unique(mbrain@meta.data$cluster))
#[1] "ABC"    "ARP"    "ASC"    "CPC"    "DC"     "EC"     "EPC"
#[8] "Hb_VC"  "HypEPC" "ImmN"   "MAC"    "MG"     "MNC"    "mNEUR"
#[15] "NendC"  "NEUT"   "NRP"    "NSC"    "OEG"    "OLG"    "OPC"
#[22] "PC"     "TNC"    "VLMC"   "VSMC"

#Clinical annotations Table S1. Clinical data of the enrolled subjects
metadata <- read.csv("data/GSE129788-annotation.csv", stringsAsFactors = F)
```

3.7.3 Create PALMO object (Time ~ 1min)

First create the PALMO S4 object using input scRNA object and annotation dataframe. The expression dataframe columns merged with input annotation dataframe. Only overlapping samples kept. Missing annotations with Sample, Donor/participant, or Time columns are removed from downstream analysis.

```
#Create PALMO object
palmo_obj <- createPALMOobject(anndata=metadata, data=mbrain)
```

Assign sample_column, donor_column and time_column variables in this step.

```
#Assign Sample, PTID and Time parameters
palmo_obj <- annotateMetadata(data_object=palmo_obj,
                              sample_column= "Sample",
                              donor_column= "Age_group",
                              time_column= "Subject_id")
```

For single cell data merge annotation and single cell metadata by mentioned sample_column.

```
#Sample overlap and final matrix
palmo_obj <- mergePALMOdata(data_object=palmo_obj, datatype="singlecell")
```

Single cell data is aggregated by average method at sample group level. Features with average expression greater than zero across all samples are kept.

```
#Aggregate data (Psuedo-bulk)
avgGroup <- "cluster"
palmo_obj <- avgExpCalc(data_object=palmo_obj, assay="RNA",
                        group_column="cluster")
head(palmo_obj@curated[["anndata"]]) #merged annotation data
head(palmo_obj@curated[["data"]]) #scRNA average expression data
```

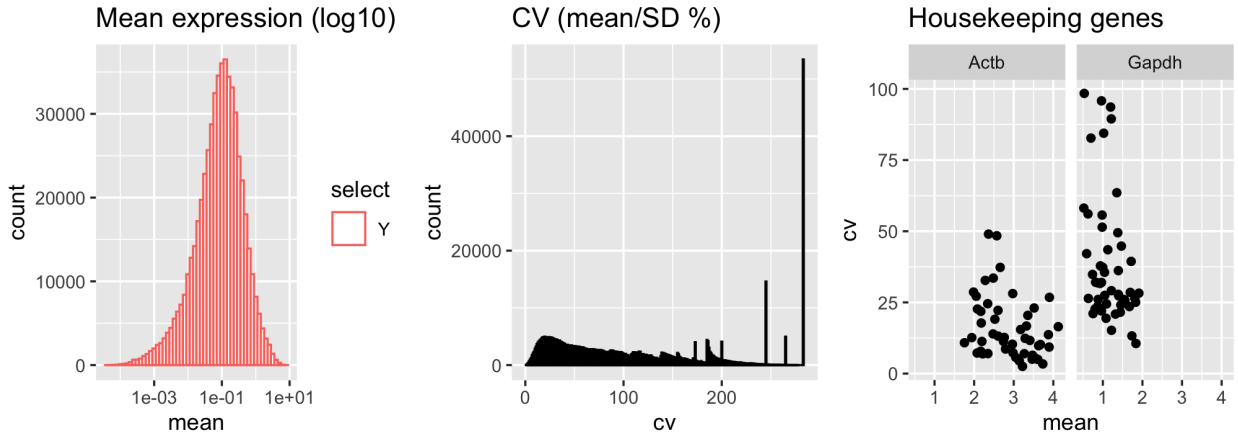
Optional step. If data consists of replicates can be merged by mergeReplicates = TRUE.

```
#Check for replicates
palmo_obj <- checkReplicates(data_object=palmo_obj, mergeReplicates = T)
```

3.7.4 CV profile (Time ~ 2min)

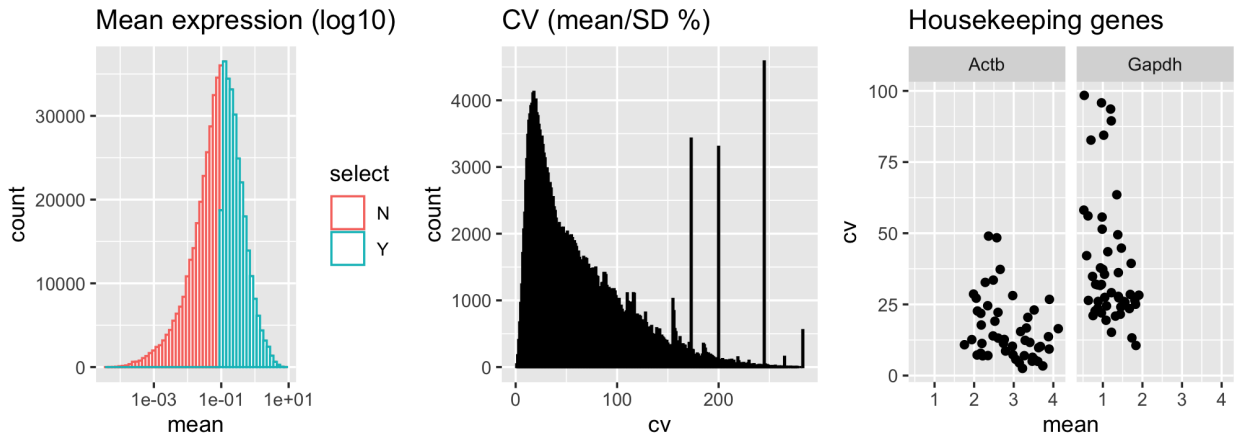
To calculate longitudinal stability within donor first visualize the CV vs mean distribution for given data. Define CV threshold using the histogram or selecting few features of interest CV as cut-off. Here we used housekeeping genes GAPDH and ACTB to define CV threshold.

```
palmo_obj <- cvCalcSCProfile(data_object=palmo_obj,
                        housekeeping_genes=c("Gapdh", "Actb"),
                        fileName="GSE129788")
```



Optional step. To check the donor or participant wise CV profile run the CV sample profile analysis.

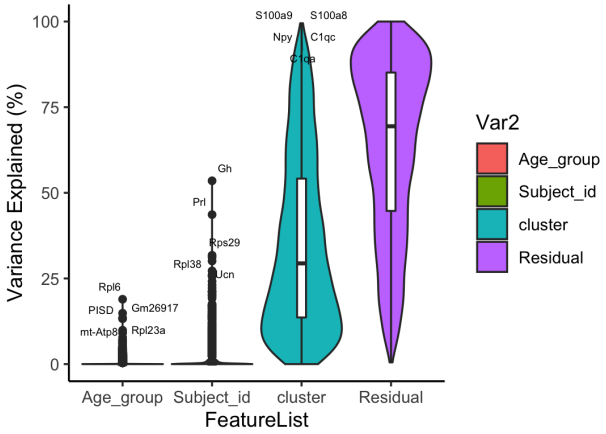
```
#Sample Celltype Mean-CV plot (output directory)
palmo_obj <- cvCalcSCProfile(data_object=palmo_obj,
                        housekeeping_genes=c("Gapdh", "Actb"),
                        meanThreshold = 0.1,
                        fileName="GSE129788")
```



3.7.5 Features contributing towards donor variations (Time ~ 5min)

Variance decomposition analysis was performed to identify the features associated with attributes of interest such as Age_group, Subject_id, cluster types. The `featureSet` is a list of variables for which fraction of variance explained by each gene is calculated. The variance explained by each gene towards the `featureSet` of interest given in percentage. *(Note: To reduce the processing time use nodes cl=8 (or greater) and lmer_control=TRUE)*

```
#Check the group of interest
head(palmo_obj@curated$anndata)

#Variance decomposition
featureSet <- c("Age_group","Subject_id","cluster")
palmo_obj <- lmeVariance(data_object=palmo_obj,
                         featureSet=featureSet,
                         meanThreshold=0.1,cl=4,
                         fileName="GSE129788")
```

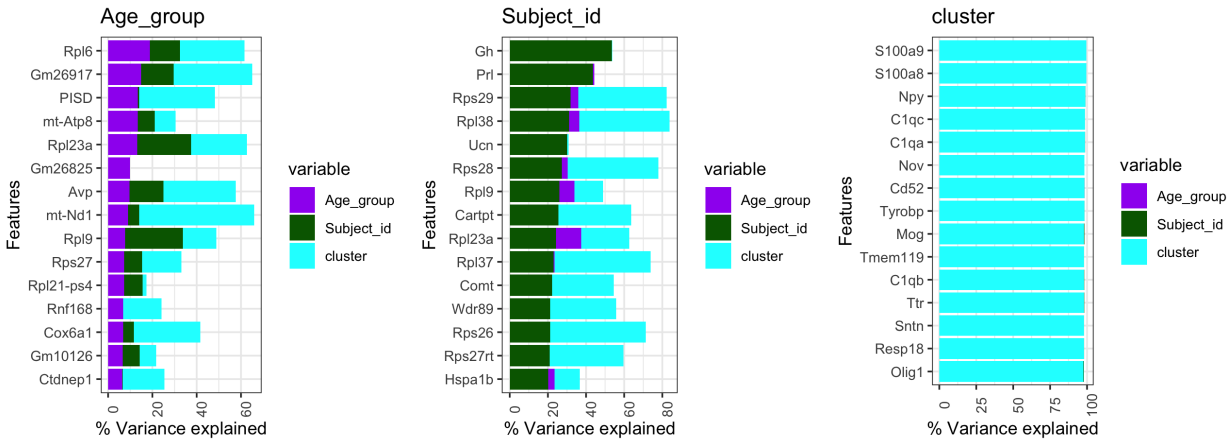



```
var_decomp <- palmo_obj@result$variance_decomposition
head(var_decomp[,featureSet])
#        Age_group Subject_id   cluster
```

```
#Rpl6    18.927915 13.6210861 29.057360
#Gm26917 14.853344 14.8230225 35.393884
#PISD    13.413587  0.5785149 34.217585
#mt-Atp8 13.361711  7.6618346  9.437131
#Rpl23a  13.238554 24.1634462 25.245072
#Gm26825  9.885817  0.0000000  0.000000
```

The top 15 features contributing to Age_group, Subject_id and cluster attributes variance can be seen in barplot.

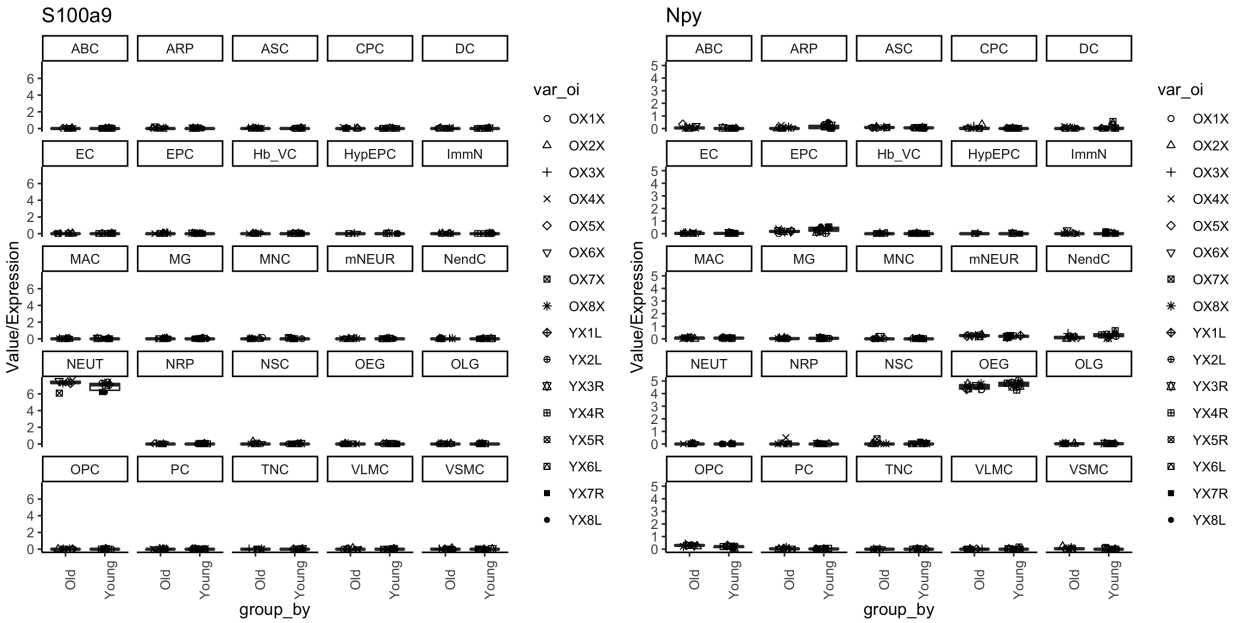
```
#Variance explained (Age_group, Subject_id and cluster)
plots <- variancefeaturePlot(vardata=var_decomp,
                             featureSet=featureSet,
                             cols=c("purple", "darkgreen", "cyan"),
                             ncol=3)
```



3.7.6 Plot the variables (Time ~ 10sec)

Gene expression plot for S100a9 (NEUT) and Npy (OEG) can be visualized by donors or by time and in celltypes.

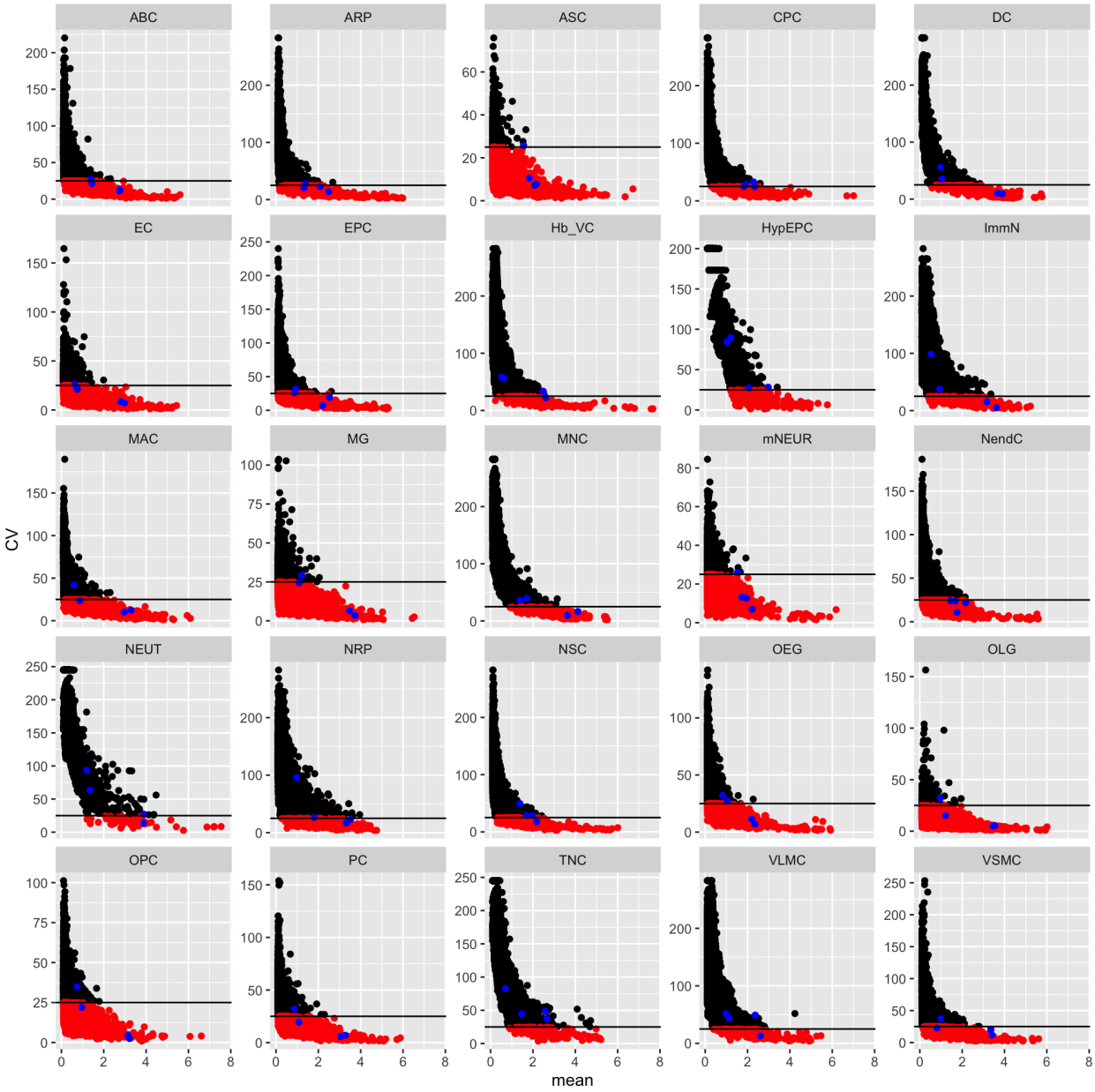
```
plots <- gene_featureplot(data_object=palmo_obj,
                          featureList=c("S100a9", "Npy"),
                          x_group_by="Age_group",
                          var_oi="Subject_id",
                          facet_by = "cluster",
                          x_text_angle=90)
```



3.7.7 Intra-donor variations over time (Time ~ 4min)

To calculate longitudinal stability within donor first visualize the CV vs mean distribution for given data. Define CV threshold using the histogram or selecting few features of interest CV as cut-off. Here we used housekeeping genes GAPDH and ACTB to define CV threshold.

```
#Calculate CV
palmo_obj <- cvCalcSC(data_object=palmo_obj,
                      meanThreshold=0.1, cvThreshold=25,
                      housekeeping_genes=c("Gapdh", "Actb"),
                      fileName="GSE129788")
```

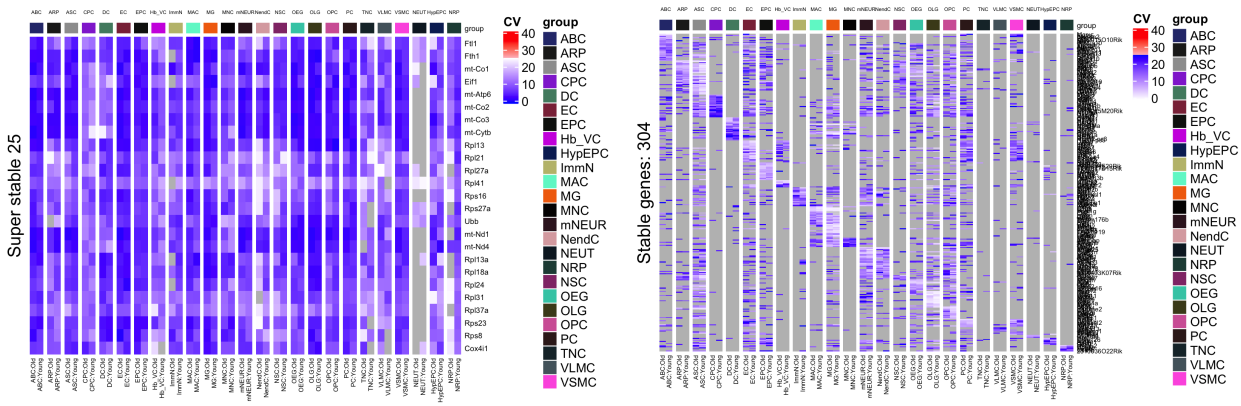


(Note: Black dots, all genes; red dots, genes with given mean and CV threshold; blue dots, housekeeping genes)

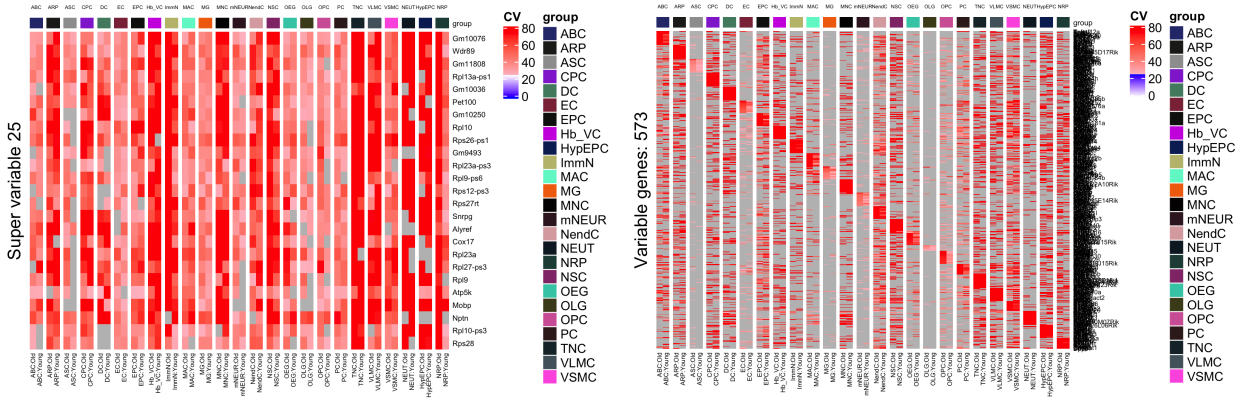
3.7.8 Find stable and variable features in longitudinal data (Time ~ 30sec)

To calculate longitudinal stability within donor define `meanThreshold` and `cvThreshold` parameters as discussed above. The analysis will use the CV across average group ("cluster") to identify Stable and variable features across celltype (cluster).

```
palmo_obj <- StableFeatures(data_object=palmo_obj,
                            cvThreshold=25,
                            topFeatures=25,
                            fileName="GSE129788")
stable_genes <- palmo_obj@result$stable_genes
```



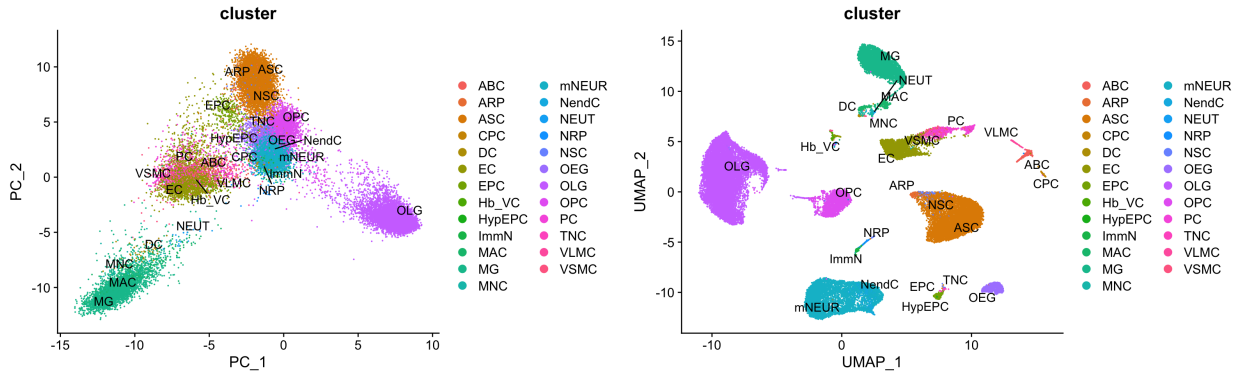

```
palmo_obj <- VarFeatures(data_object=palmo_obj,
                         cvThreshold=25,
                         topFeatures=25,
                         fileName="GSE129788")
var_genes <- palmo_obj@result$var_genes
```



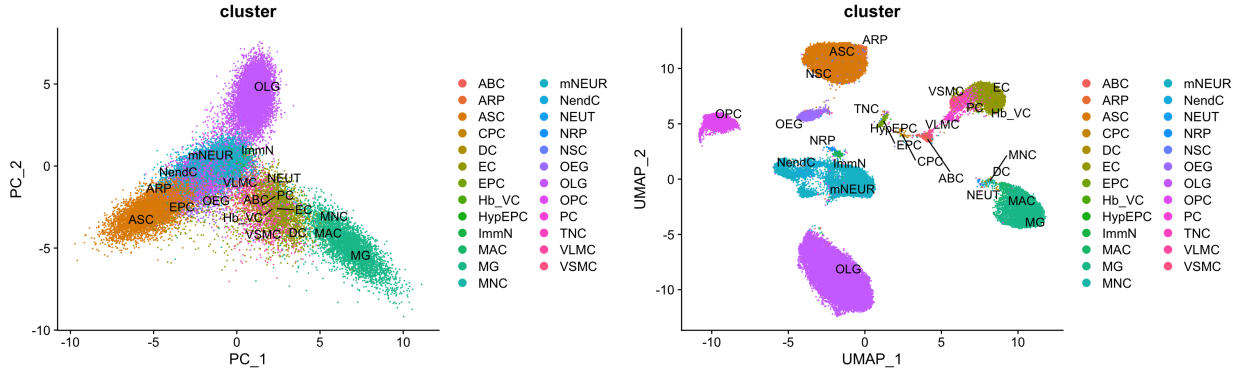
3.7.9 UMAP Plot (Time ~ 2min)

To visualize how these stable (304) or variable (573) features performing, users can reduce high dimensional single cell data into lower dimensions using selected Top features from stable/variable analysis. The UMAP plot shows whether identified stable features can reproduce celltype/group identity based on few but stable features.

```
#Stable genes UMAP
dimUMAPPlot(data_object=palmo_obj, nPC=15,
            gene_oi=unique(stable_genes$gene),
            group_column="cluster", plotname="stable",
            fileName="GSE129788")
```



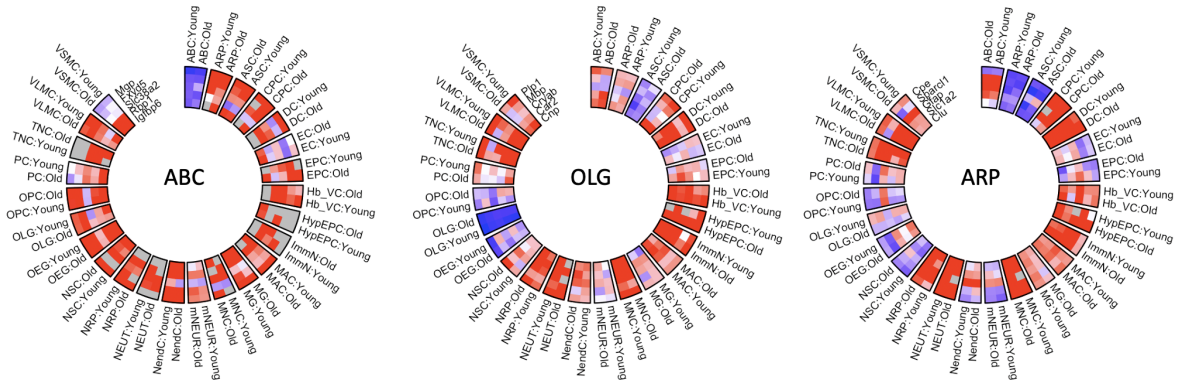

```
#Variable genes UMAP
dimUMAPPlot(data_object=palmo_obj, nPC=15,
            gene_oi=unique(var_genes$gene),
            group_column="cluster", plotname="variable",
            fileName="GSE129788")
```



3.7.10 Celltype-specific Circos CV Plot (Time ~ 30sec)

The top stable features across mouse brain celltypes Arachnoid barrier cells (ABC), Oligodendrocytes (Olg), Astrocyte-restricted precursors (ARP) can be visualized in circos plot for easy comparison.

```
#ABC: Arachnoid barrier cells
geneList <- c("Mgp","Fxyd5","Slc38a2","Rbp1", "Igfbp6")
plotres <- genecircosPlot(data_object=palmo_obj,
                          geneList=geneList, colorThreshold=25)

#Olg: Oligodendrocytes
geneList <- c("Plp1", "Mbp", "Cryab", "Car2", "Cnp")
plotres <- genecircosPlot(data_object=palmo_obj,
                          geneList=geneList, colorThreshold=25)

#ARP: Astrocyte-restricted precursors
geneList <- c("Cpe", "Sparcl1", "Gfap", "Slc1a2", "Clu")
plotres <- genecircosPlot(data_object=palmo_obj,
                          geneList=geneList, colorThreshold=25)
```


ABC
OLG
ARP

4 Authors

Suhas Vasaikar, Aarthi talla and Xiaojun Li designed the PALMO algorithm. Suhas Vasaikar implemented the PALMO package.

5 License

PALMO is licensed under the MIT-License.

6 Session info

```
sessionInfo()
#> R version 4.0.3 (2020-10-10)
#> Platform: x86_64-apple-darwin17.0 (64-bit)
#> Running under: macOS Catalina 10.15.7
#>
#> Matrix products: default
#> BLAS:   /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRblas.dylib
#> LAPACK: /Library/Frameworks/R.framework/Versions/4.0/Resources/lib/libRlapack.dylib
#>
#> locale:
#> [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
#>
#> attached base packages:
#> [1] grid      stats     graphics  grDevices utils     datasets  methods
#> [8] base
#>
#> other attached packages:
#> [1] PALMO_0.1.1
#>
#> loaded via a namespace (and not attached):
#>   [1] readxl_1.3.1                backports_1.2.0
#>   [3] circlize_0.4.11             plyr_1.8.6
#>   [5] igraph_1.2.8                lazyeval_0.2.2
#>   [7] splines_4.0.3               listenv_0.8.0
#>   [9] scattermore_0.7             GenomeInfoDb_1.24.2
#>  [11] ggplot2_3.3.5               digest_0.6.28
#>  [13] htmltools_0.5.2             fansi_0.5.0
#>  [15] magrittr_2.0.1              tensor_1.5
#>  [17] cluster_2.1.0               ROCR_1.0-11
#>  [19] ComplexHeatmap_2.4.3        globals_0.14.0
#>  [21] readr_1.4.0                 modelr_0.1.8
#>  [23] matrixStats_0.61.0          colorspace_2.0-2
#>  [25] rvest_0.3.6                 blob_1.2.1
#>  [27] ggrepel_0.9.1               haven_2.3.1
#>  [29] xfun_0.25                   dplyr_1.0.7
#>  [31] RCurl_1.98-1.2              crayon_1.4.2
#>  [33] jsonlite_1.7.2              lme4_1.1-25
#>  [35] spatstat_1.64-1             spatstat.data_2.1-0
#>  [37] survival_3.2-7              zoo_1.8-9
#>  [39] glue_1.5.0                  polyclip_1.10-0
#>  [41] gtable_0.3.0                zlibbioc_1.34.0
#>  [43] XVector_0.28.0              leiden_0.3.9
#>  [45] DelayedArray_0.14.1         GetoptLong_1.0.4
#>  [47] SingleCellExperiment_1.10.1 future.apply_1.8.1
#>  [49] shape_1.4.5                 BiocGenerics_0.34.0
#>  [51] abind_1.4-5                 scales_1.1.1
#>  [53] pheatmap_1.0.12             DBI_1.1.0
#>  [55] miniUI_0.1.1.1              Rcpp_1.0.7
#>  [57] viridisLite_0.4.0           xtable_1.8-4
#>  [59] clue_0.3-57                 reticulate_1.22
#>  [61] stats4_4.0.3                htmlwidgets_1.5.4
```

```
#>  [63] httr_1.4.2                 RColorBrewer_1.1-2
#>  [65] ellipsis_0.3.2             Seurat_4.0.0
#>  [67] factoextra_1.0.7.999       ica_1.0-2
#>  [69] farver_2.1.0               pkgconfig_2.0.3
#>  [71] uwot_0.1.10                dbplyr_1.4.4
#>  [73] deldir_1.0-6               utf8_1.2.2
#>  [75] tidyselect_1.1.1           rlang_0.4.12
#>  [77] reshape2_1.4.4             later_1.3.0
#>  [79] munsell_0.5.0              cellranger_1.1.0
#>  [81] tools_4.0.3                generics_0.1.1
#>  [83] broom_0.7.2                ggridges_0.5.3
#>  [85] evaluate_0.14              stringr_1.4.0
#>  [87] fastmap_1.1.0              yaml_2.2.1
#>  [89] goftest_1.2-3              knitr_1.30
#>  [91] fs_1.5.0                   fitdistrplus_1.1-6
#>  [93] purrr_0.3.4                RANN_2.6.1
#>  [95] pbapply_1.5-0              future_1.23.0
#>  [97] nlme_3.1-149               mime_0.12
#>  [99] xml2_1.3.2                 compiler_4.0.3
#> [101] plotly_4.10.0              png_0.1-7
#> [103] spatstat.utils_2.2-0       reprex_0.3.0
#> [105] tweenr_1.0.1               tibble_3.1.6
#> [107] statmod_1.4.35             stringi_1.7.5
#> [109] forcats_0.5.0              lattice_0.20-41
#> [111] Matrix_1.3-4               nloptr_1.2.2.2
#> [113] vctrs_0.3.8                pillar_1.6.4
#> [115] lifecycle_1.0.1            lmtest_0.9-39
#> [117] GlobalOptions_0.1.2        RcppAnnoy_0.0.19
#> [119] bitops_1.0-7               data.table_1.14.2
#> [121] cowplot_1.1.1              irlba_2.3.3
#> [123] GenomicRanges_1.40.0       httpuv_1.6.3
#> [125] patchwork_1.1.1            R6_2.5.1
#> [127] promises_1.2.0.1           KernSmooth_2.23-17
#> [129] gridExtra_2.3              IRanges_2.22.2
#> [131] parallelly_1.28.1          codetools_0.2-16
#> [133] boot_1.3-25                MASS_7.3-53
#> [135] assertthat_0.2.1           SummarizedExperiment_1.18.2
#> [137] MAST_1.14.0                rjson_0.2.20
#> [139] SeuratObject_4.0.2         sctransform_0.3.2
#> [141] GenomeInfoDbData_1.2.3     S4Vectors_0.26.1
#> [143] mgcv_1.8-33                parallel_4.0.3
#> [145] hms_0.5.3                  rpart_4.1-15
#> [147] tidyverse_1.3.0            tidyr_1.1.4
#> [149] minqa_1.2.4                rmarkdown_2.5
#> [151] Rtsne_0.15                 ggforce_0.3.2
#> [153] Biobase_2.48.0             shiny_1.7.1
#> [155] lubridate_1.7.9
```