DE Methods Notes

Mike & Jesslyn

5/7/2020

Understanding Seurat functions

`FindNeighbors()`

`AddModuleScore()` (mike)

Description: calculate the average expression levels of each program, subtracted by the aggregated expression of controlled feature sets. Analyzed features binned based on average expression, control features randomly selected from each bin

Inputs:

1. `object`, Seurat object
2. `features`, List of features to compute module score for

- `pool` = NULL, control features
- `nbin` = 24, number of bins of aggregate expression levels for all analyzed features (what effects do higher/lower `nbin` values have?)

From Tirosh 2016 for MITF/AXL program scores (100 genes per program)

The top 100 MITF-correlated genes across the entire set of malignant cells were defined as the MITF program, and their average relative expression as the MITF-program cell score. The average expression of the top 100 genes that negatively correlate with the MITF program scores were defined as the AXL program and used to define AXL program cell score.

To decrease **the effect that the quality and complexity of each cell's data** might have on its MITF/AXL scores we defined control gene-sets and their average relative expression as control scores, for both the MITF and AXL programs. These control cell scores were subtracted from the respective MITF/AXL cell scores.

The control gene-sets were defined by first binning all analyzed genes into 25 bins of aggregate expression levels and then, for each gene in the MITF/AXL gene-set, randomly selecting 100 genes from the same expression bin as that gene. In this way, a control gene-sets have a comparable distribution of expression levels to that of the MITF/AXL gene-set and the control gene set is 100-fold larger, such that its average expression is analogous to averaging over 100 randomly-selected gene-sets of the same size as the MITF/AXL gene-set.

To calculate significance of the changes in AXL and MITF programs upon relapse, we defined the expression log2-ratio between matched pre- and post- samples for all AXL and MITF program genes (Fig. 3D). Since AXL and MITF programs are inversely related, we flipped the signs of the log-ratios for MITF program genes and used a t-test to examine if the average of the combined set of AXL program and (sign-flipped) MITF program genes is significantly higher than zero, which was the case for four out of six matched sample pairs (Fig. 3D, black arrows).

`FindAllMarkers()`

`FindMarkers()` (jesslyn)

Description: Finds differentially expressed genes between two specific groups of cells specified by the ident.1 and ident2 arguments.

Inputs:

1. `object` - Seurat object
2. `ident.1` - identity class to define markers for
3. `ident.2` - identity class for comparison (if null, use all other cells for comparison)
4. `test.use` - algorithm to statistically calculate differential expression values for each gene
5. `min.pct` - only test genes that are expressed in x% cells in either of the two populations (default = 0.1)
6. `logfc.threshold` - only test genes that show at least x-logfold difference between the two idents

test.use:

1. **wilcox** (default)
   - Uses a Wilcoxon Rank Sum test (aka Mann-Whitney U test)
   - Non-parametric test
   - For continuous variables
   - Compares a non-normally distributed, but at least ordinally scaled, parameter in two unpaired samples.

   How it works:
   - Have a big expression matrix
   - If look at all of the cells together, can rank the cells based on the count you read
   - Look at the sum of the ranks of all the cells in one cluster, and compare with another cluster
2. **MAST**
   - Parametric test
   - Based on a zero-inflated negative binomial model
   - Uses a hurdle model tailored to scRNA-seq data
3. **DESeq2**
   - Uses a negative binomial distribution
   - Normalize raw counts in different samples
   - Stabilize/ shrink variance by borrowing info from other genes
4. **bimod**
   - Likelihood-ratio test for single cell gene expression
5. **roc**
   - Uses ROC analysis
   - Evaluates a classifier for each gene to classify between two groups of cells using AUC values
   - AUC=1 means that expression values for this gene alone can perfectly classify the two groupings (each cell in cells.1 exhibit a higher level than each cell in cells.2)
   - AUC=0 also means that there is perfect classification but in the other direction
   - AUC=0.5 means that there is no predictive power to classify the two groups
   - Returns a ranked matrix of differentially expressed genes
6. **t**
   - Uses the Student's t-test
7. **LR**
   - Uses a logistic regression model

`FindVariableFeatures()`

Description: "Identifies features that are outliers"

Inputs:

1. `object` - Seurat object

2. `selection.method` - algorithm to use to select most variable features
3. `nfeatures` - number of genes to select as top variable, can only use when selection.method is not 'mvp'

selection.methods:

1. **vst** *default* (appears to be similar to DESeq2), - fits a line to $log(variance)$ vs $log(mean)$ using local polynomial regression (loess). - standardizes feature values using observed mean and expected variance - feature variance is calculate on the standarized values after clipping to a maximum
    - **clip.max**: After standardization values larger than clip.max will be set to clip.max; default is 'auto' which sets this value to the square root of the number of cells.
2. **mean.var.plot (mvp)** (identify variable features while controlling for relationship between variability and avg expression) - calculates average expression and dispersion for each feature - divides features into num.bin and calculates z-scores for dispersion within each bin
    - **num.bin**: total number of bins to use in the scaled analysis; default is 20
3. **dispersion (disp)** simply selects the genes with the highest dispersion