**Use of the “Dimension Reduction” DSP DA script**

**Intended use**

The Dimension Reduction DSP DA script was designed for data from the GeoMx DSP high-plex RNA assays, such as the CTA NGS readout application, but can be used with other assays.

This script does the following:

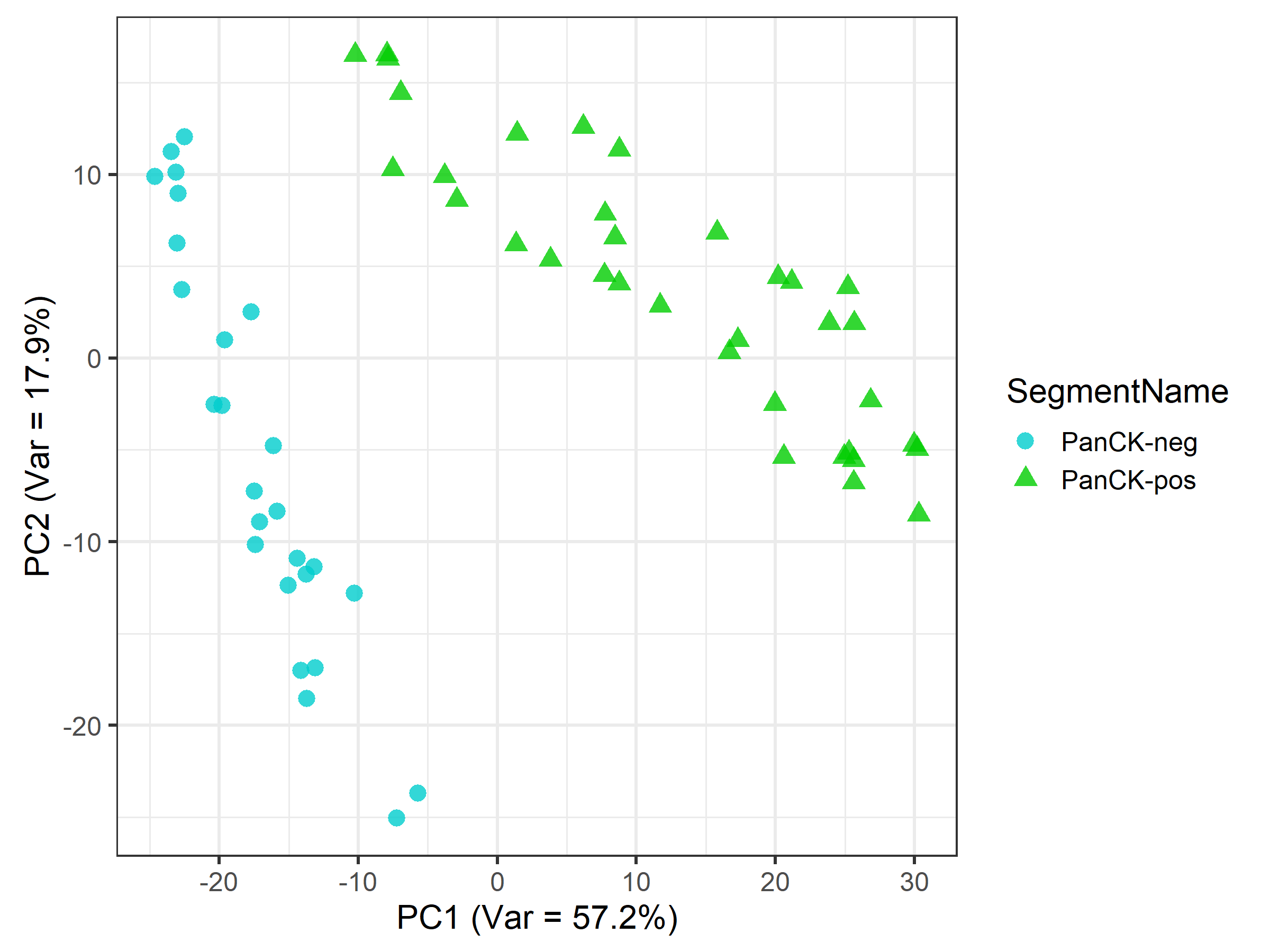
1. Performs dimension reduction analysis as specified by the user on the segments within a study. Dimension reduction analysis options include:
   1. PCA
   2. tSNE
   3. UMAP
2. Generates scatter plots of the resulting reduced dimensions. There are options of users to control the color, shape, point size, font, font size, and file type. Plots are produced that are 6” tall by 8” wide, at 300 DPI (if applicable).
3. If a PCA is being shown, it will also graph the cumulative proportion of variance explained by each principal component up to the first 15 components
4. Saves an updated annotation sheet to allow for re-graphing with external software as a excel spreadsheet. If a PCA is graphed, all principal components, loadings, and variance estimates are captured within the spreadsheet as well.

**Contents**

*Page 2*: Loading into the DSP-DA

*Page 3*: Setting User Parameters

*Page 6*: Interpretation of resulting files and figures

****

*For more example graphs see page 4*

**Loading into the DSP-DA:**

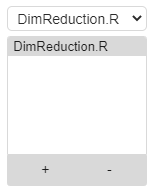
The DimReduction.R file may be loaded into the custom scripts section of the DSP-DA after you have a dataset processed and ready for analysis. To do so open the custom script section by clicking on the button shown below:



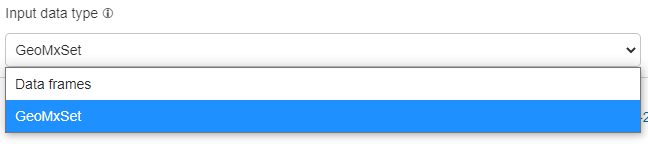
Select the “Manage” tab to open the area to load and edit scripts:



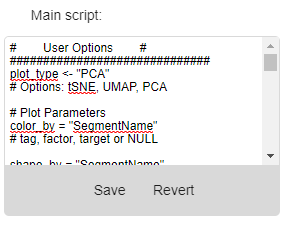
In the Management tab to add a new script and adjust parameters, fill out and then scroll to the bottom of the page. Use the “+” button to add the DimReduction.R file to the script:



This script will work with both DataFrame and GeoMxSet inputs.



Once added the parameters of the script can be adjusted by editing the top lines in the script and hitting the “Save” button. You do **not** need to check the Create new dataset button.

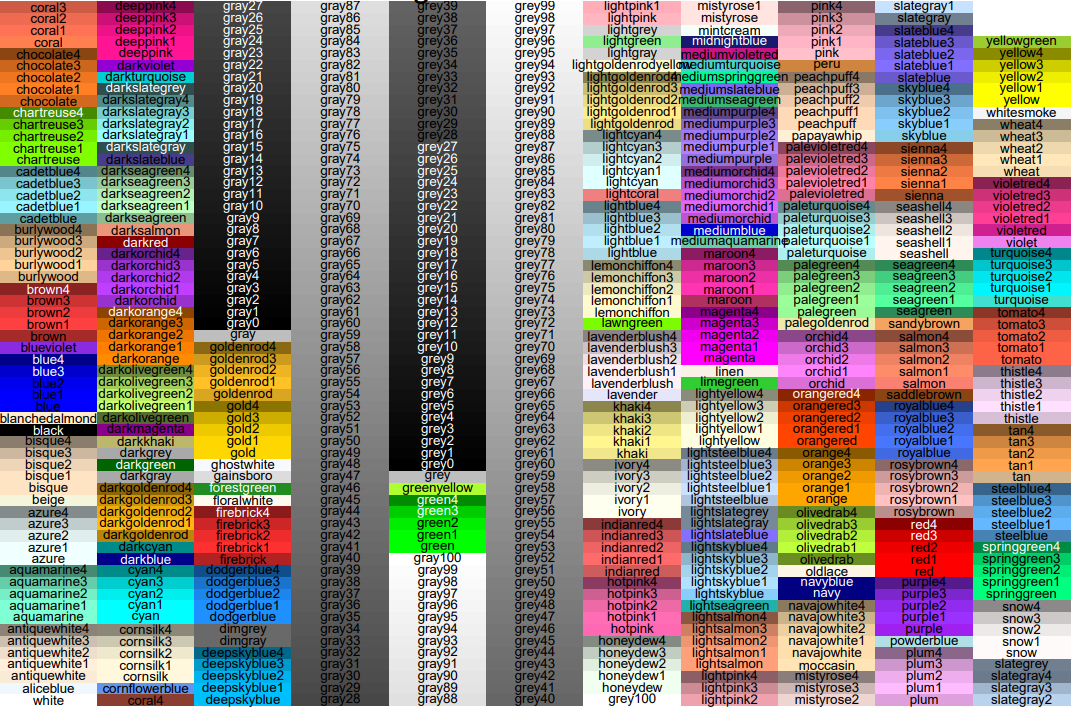


Parameter options are described in full on the next page.**Setting User Parameters:**

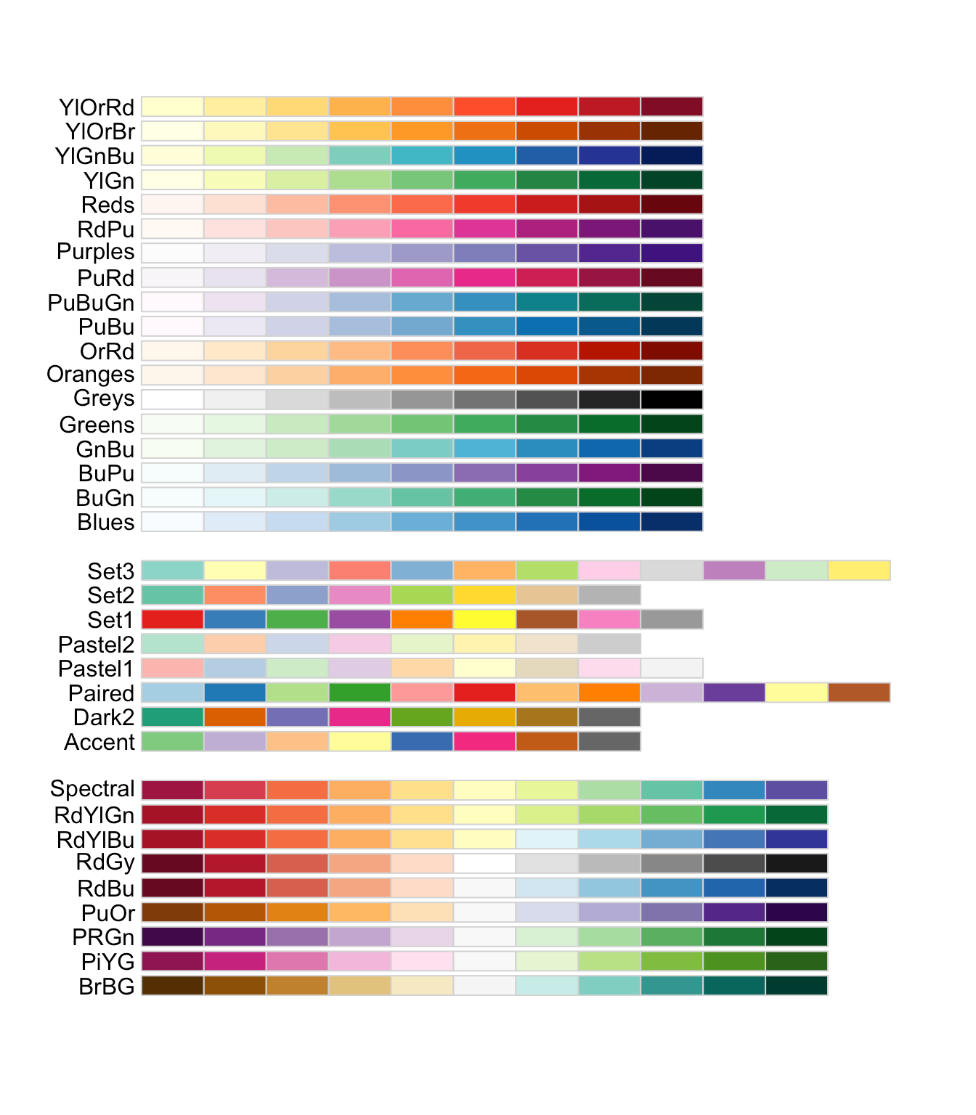
There are a few settings that can be adjusted easily by the user at the top of the script. These include:

* plot\_type – set this to either “PCA”, “UMAP”, or “tSNE” based on user preference. No other values or methods are currently supported
* Plotting parameters: (any of the graphing parameters may be set to NULL to ignore the parameter)
  + *color\_by* – set this to the name (column name) of an annotation tag, factor within the segment annotations, or the display name of a target. For example, you may have a factor named “Location” that you want to visualize, this may be used with the color parameter. Alternatively, “VEGFA” can be used to color points by the continuous expression of VEGFA. Color is shown as log2 counts if a target is selected due to how color distributions are perceived.
  + *shape\_by* - set this to the name of an annotation tag or factor.   
     Note: Target names may not be used for shape.
  + *size\_by* – set this to the display name of a target. Size is shown as linear counts due to how point size distributions are perceived.
  + *plot\_font* – is a list, which includes family and size. Family can be set to ‘sans’, ‘serif’, or ‘mono’ to use Helvetica, Times New Roman, or Courier New fonts, respectively. Additional fonts may be supported as well, but not all fonts are available. Size is relative, increasing the number (default = 15) shall increase font size relative to the plot size for all labels on the plots.
  + *save\_as* – is a string which defines the type of format you would like to save the graphs as. If PDF is selected (default), it will save graphs as a multipage PDF, which is especially useful for PCA analysis which outputs 4 graphs. PNG or SVG formats are most appropriate for inclusion in print documents, and use a dpi resolution of 300. This may be edited further down in the script if higher resolution is needed. Search for “dpi” to change that variable. Similarly, if a different size is needed, search for “width” or “height” within the script.
* Controls for colors: [see Phase 4: Colors & Pallet Options figure below for examples]
  + *plot\_colors* – is a list which should contain either:
    - colors that can be recognized by R. These should be either named colors (e.g. “orange2”) or hexadecimal colors (“#ABABAB”). At least 1 color is needed for each unique entry in the *color\_by* column. See below for a cheat sheet of all named R colors.
    - Alternative, if you do not want to specify all the colors, you may set the first color to the name of a color palette from the palettes listed below. For example, “Dark2” or “Set3”.
  + *color\_levels* – is a list of the annotation tag/factor levels that should be matched to the colors defined in plot\_colors. They will be used in the same order as the levels defined in *plot\_colors*. Additional notes:
    - If coloring by a target set this to “High”, “Mid”, and “Low”. “Mid” is optional, if you only want to specify the colors to be used with the minimum and the maximum values. “Mid” defines the color for the median value of a target.
    - If you have more values in your annotation tag than defined, the script will use “Set1” below to add new colors to the palette to represent the values. You do not have to define all of levels within the tag or factor, but undefined levels will be added in alphanumeric order.

**Named R Colors:**



**Palette Options:** Names to the left of each palette represent the text that can be used in the first entry in *plot\_colors*. Any of the values shown here may be used. Note that palettes with light colors may be harder to distinguish on the graph.



**Example parameter setup:**

Desired plot

* Example 2: a UMAP with color based on PanCK segmentation, with PanCK+ in green and PanCK- in cyan. Shape should be based on the slide name

plot\_type = "UMAP"

color\_by = "SegmentName"

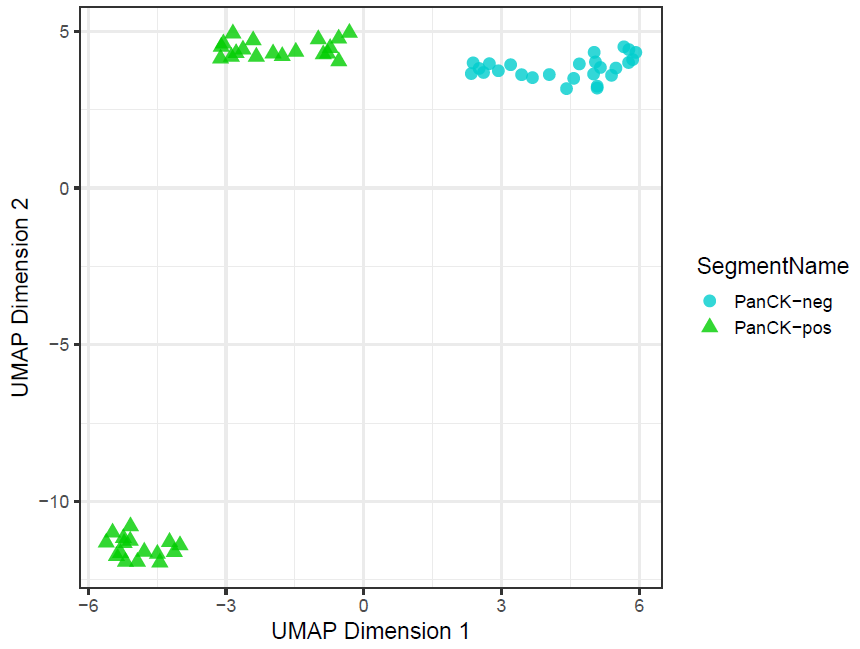
shape\_by = "SegmentName"

size\_by = NULL

plot\_colors = list("green3", "cyan2")

color\_levels = c("PanCK-pos", "PanCK-neg”)

resulting graph:



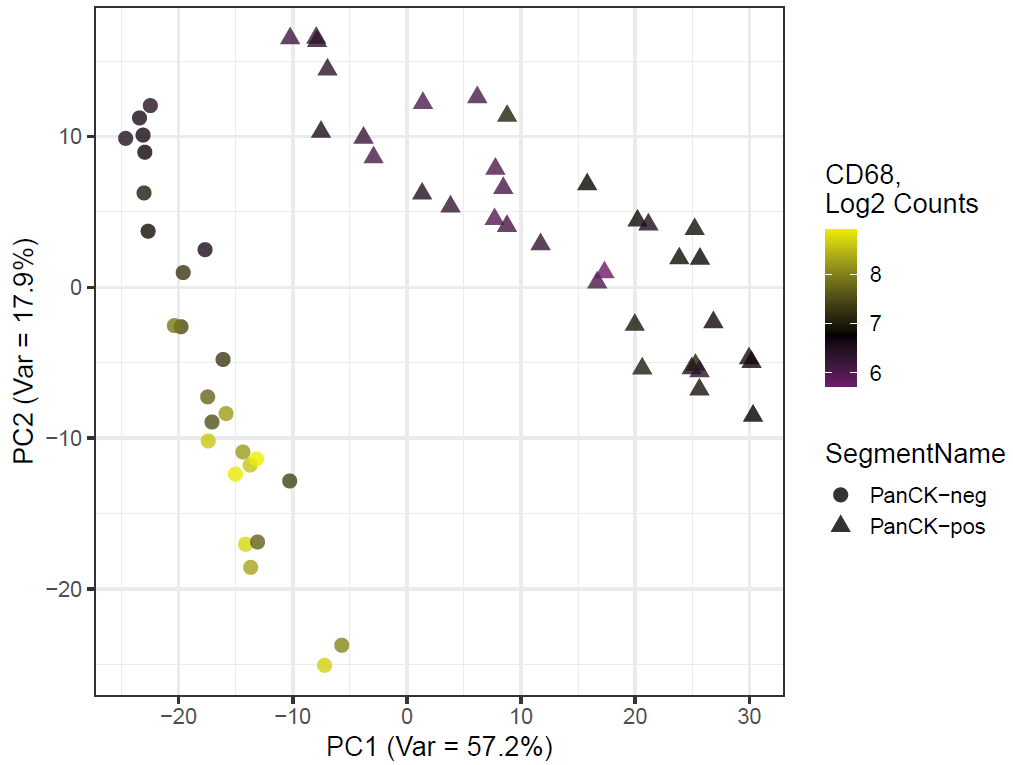
* Example 2:

plot\_type = "PCA"

color\_by = "CD68" # tag, factor, or target

shape\_by = "SegmentName" # tag, or factor

plot\_colors = list("yellow2", "black", "magenta2")  
color\_levels = c("High", "Mid", "Low")



* Example 2:

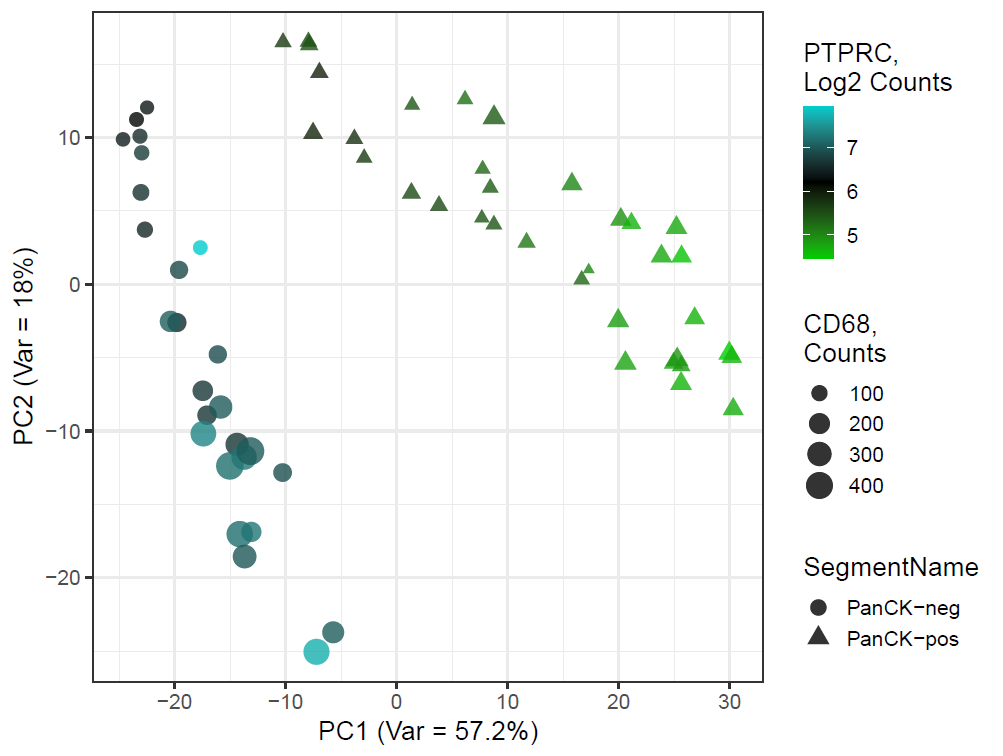
plot\_type = "PCA"

color\_by = "CD68"

size\_by = "PTPRC"

shape\_by = "SegmentName"

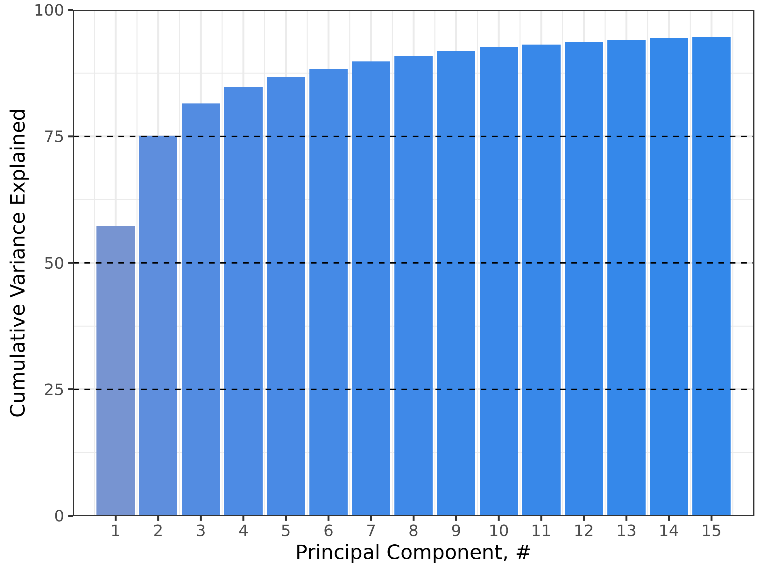
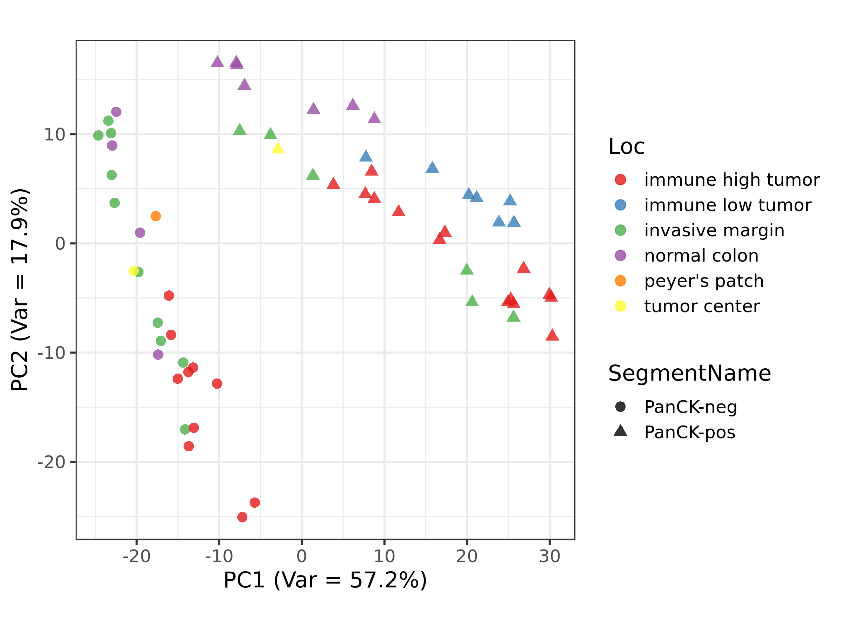
plot\_colors = list("cyan3", "black", "green3")  
color\_levels = c("High", "Mid", "Low")



**Interpretation of resulting files and figures**

**PCA**

Principal component analysis (PCA), is a method for reducing high dimension data down to lower data spaces. It iteratively identifies the linear component that explains the most variation in a dataset, through a process called singular value decomposition. The process is performed on scaled, log2 transformed expression data and captures each dimension and then looks for orthogonal axes of variation that would explain the next most amount of variation within the dataset. These principal components (PCs) can then be used to visualize samples in a much smaller sample space, as well as understanding the amount of variation that the analysis has captured for any given number of components. Samples which appear closer on a given principal component have similar aggregate expression of the genes that comprise that component. An example is shown below of the two graphs output by this method:

****

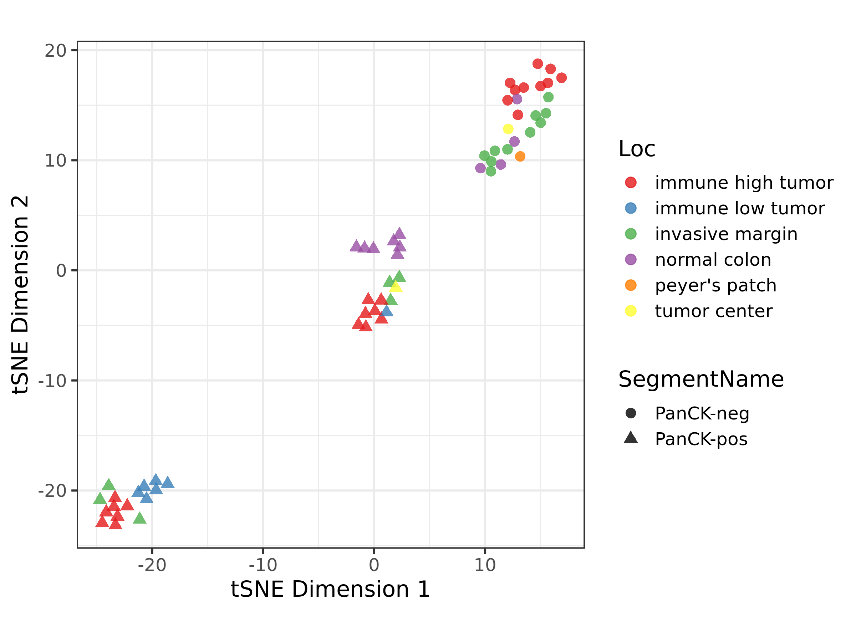
In this example, we observe 2 strong clusters that separate based on both PC1 and PC2 based on the segmentation strategy used, where PanCK- regions (stroma) are on the left of the graph, and PanCK+ regions are on the top right. Color denotes a regional factor used to categories ROIs as they were selected, and so we can further explore within-cluster distributions such as the fact that immune high tumor PanCK+ ROIs separate from normal colon PanCK+ ROIs. The graph on the right is also output, and it shows the cumulative variance explained by each the components measured. While we only output the first 3 PCs in the annotation data, you can see by this graph that additional PCs explain smaller & smaller variances in expression. If your dataset is particularly similar or diverse more or less of the variance will be explained by the first PC. The variance explained is also shown as a percentage on the axes of the graphs.

**tSNE**

tSNE (t-Distributed Stochastic Neighbor Embedding) is a method do cluster samples based on expression that is not linearly or orthogonally constrained like the PCA plot. However, it is a stochastic method, and cannot be used to estimate where a new sample would fit within the defined clusters. As such, it is useful for data exploration, but less so for defining characteristics that may be shared in a new dataset.

Reference: L.J.P. van der Maaten and G.E. Hinton. Visualizing High-Dimensional Data Using t-SNE. Journal of Machine Learning Research 9(Nov):2579-2605, 2008.

An example graph is shown on the next page:



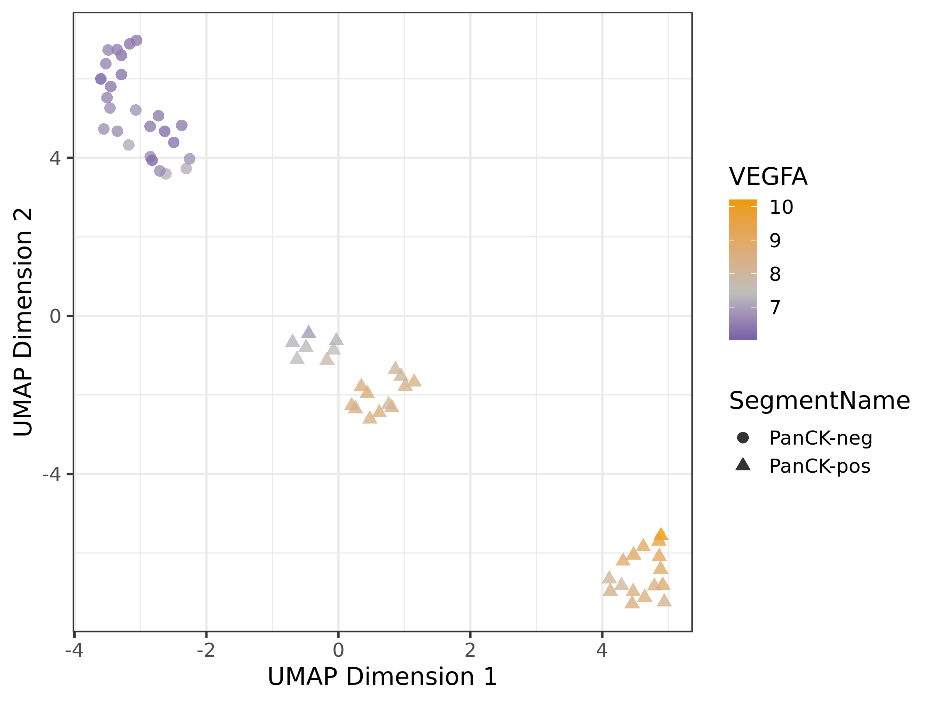
In this particular example, tSNE identifies 3 clusters of samples, 1 that is based on PanCK- segments, and then two separate clusters of PanCK+ ROIs. While not visualized here, these clusters may be patient driven, as disease or cancer samples tissues tend to be less closely related than adjacent normal tissues from the same tissue. To visualize this, you can set the color or shape to the scan name or patient ID.

**UMAP**

UMAP (Uniform Manifold Approximation and Projection) is a method of dimension reduction developed for developing a reproducible method for graphing samples in a non-linearly constrained fashion. It has been heavily used by the single- For more information about this method see the reference below:

Reference: McInnes, L, Healy, J, UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, ArXiv e-prints 1802.03426, 2018

An Example of a UMAP for a set of samples produced is shown below:



In this particular example, color is being defined by the target “VEGFA”, shape is being set by the factor “SegmentName”, and the color palette used is “Set1”. Here we observe 3 clusters. As with tSNE, two clusters are PanCK-positive, which show higher expression of VEGFA than the PanCK-negative segments, which would be expected in the colorectal cancer setting.

**Excel File**

In addition to graphs, the script will also output a excel spreadsheet file with new data columns depending on what variables were used. If a tSNE or UMAP was graphed, the spreadsheet will only have one tab (Segment Annotations).

Within segment annotations at the end of the table you will find the following new data columns:

* If color is set to a target name it shall be included in the table, with the log2 count values shown for the target based on the active data frame selected
* If UMAP or tSNE is selected 2 new columns (Dim1 and Dim2) will be added. If PCA is selected Dim3 will also be added. Dim1&2 represent the graphed values, Dim3 is added in case users are interested in graphing additional PCs.

If a PCA is used there will be 3 additional tabs on the excel file these will be:

* Principal Components (All) – a table with the principal component scores for all segments for all calculated components during the analysis
* PC Loadings – These show the loading weights for each principal component. The loadings are sorted based on the first principal component, with the highest absolute value loadings shown at the top
* Variance Estimates – This shows the standard deviation & proportion of variance explained for each PC as well as the cumulative proportion of variance explained