**Functions**

**compute\_binomial\_deviance\_\_genuine\_and\_randomized\_counts**

* purpose
  + compute binomial deviance [1] for genuine and randomized data
  + provide information to make a filtering decision – to specify a number of genes for analysis
* parameters
  + required
    - pandas dataframe: UMI counts
    - row labels: gene IDs
    - column labels: cell IDs
  + optional
    - number of randomizations to be performed: default = 11
    - percentiles of binomial deviance to be listed in output “summary statistics” data frame – python list. default: = [.25,.5,.75,.9,.95, .96, .97, .98, .99,.995,.999]
* output value: list
  + pandas dataframe: binomial deviance for genuine data
  + pandas dataframe: binomial deviance for each set of randomized counts: one column per randomized data set
  + pandas dataframe: summary statistics
    - median percentiles of binomial deviance for randomized data
    - number of genes with binomial deviance larger than the median percentiles of the randomized data

**data\_prep\_for\_RF\_genes**

* purpose
  + calculate null residuals
  + prepare synthetic data sets for random forest classification
* parameters
  + required
    - pandas dataframe: UMI counts
      * row labels: gene IDs
      * column labels: cell IDs
    - pandas dataframe: binomial deviance for genuine data
    - number of genes to select
    - folder to which output data sets will be written
  + optional
    - number of synthetic data sets to prepare: default=100
    - data set name for standardized null residuals: default = null\_residuals\_std
    - data set name prefix for synthetic data: default = null\_residuals\_std\_synth\_
* output value: list
  + pandas dataframe: standardized null residuals – genuine data
* output data sets
  + CSV file : standardized null residuals – genuine data
  + CSV files: synthetic data sets for random forest classification (randomized standardized null residuals)

**RandomForestClassifier\_for\_proximity\_by\_gene**

* purpose
  + perform classification with random forests to obtain proximities between genes
  + the classification problem is to distinguish between genuine and synthetic data, as described in references [2, 3, and 4]
* parameters
  + required
    - number of trees in each random forest
    - folder containing input data sets
    - folder to which output data sets will be written
  + optional
    - name of input CSV data set containing standardized null residuals – for genuine data: default= ‘null\_residuals\_std’
    - prefix for input CSV data sets containing synthetic data: default = ‘null\_residuals\_std\_synth\_’
    - prefix for output CSV data sets containing random forest proximities: default = 'gene\_proximities\_'
    - first synthetic data set to be input: default=0
* input data sets
  + CSV file : standardized null residuals – genuine data
  + CSV files: synthetic data sets for random forest (randomized standardized null residuals)
* output value: list
  + largest index of the synthetic data sets analyzed; for example, if 100 sets of individual clusterings are to be calculated, the value returned is 100
* output data sets
  + CSV files: proximities; integer: diagonal = number of trees

**gene\_clusters\_\_individual\_and\_consensus**

* purpose
  + calculate spectral clusterings [5] for each set of random forest proximities; refer to these as “individual” clusterings – as opposed to “consensus” clusterings
  + calculate
    - consensus clusterings based on all individual clusterings
    - two sets of consensus clusterings – each based on half of the individual clusterings
    - four sets of consensus clusterings – each based on one quarter of the individual clusterings
  + the variation between the clusterings derived from a quarter or half of the individual clusterings will give estimates of the consensus clusters’ stability
* parameters
  + required
    - maximum number of clusters to be computed
    - folder containing input data sets
  + optional
    - prefix for input CSV data sets containing random forest proximities: default = 'gene\_proximities\_'
* input data sets
  + CSV files: proximities; integer: diagonal = number of trees – one file for each input synthetic data set
* output value: list
  + pandas dataframe: individual clusterings – these are stacked in a single dataframe

Example: suppose that

* + - 200 genes were chosen by the filtering criteria
    - 100 synthetic data sets, hence 100 proximity arrays, were produced
    - the maximum number of clusters equals 20

then this output dataframe

* contains 20,000 rows: the number of genes times the number of proximity arrays
  + - * it contains 20 columns
        + one for each of 19 clusterings – for 2 to 20 clusters
        + a counter, identifying the proximity array from which the clusterings are derived
  + pandas dataframe: consensus clusterings derived from all individual clusterings
  + pandas dataframe: two sets of consensus clusterings, each derived from nonoverlapping halves of the individual clusterings; these are stacked, and distinguished by a counter in the column ‘cluster\_set’
  + pandas dataframe: four sets of consensus clusterings, each derived from nonoverlapping quarters of the individual clusterings; these are stacked, and distinguished by a counter in the column ‘cluster\_set’

**ARI\_and\_Misclassification \_Error\_for\_ individual \_clusterings**

* purpose:
  + evaluate stability of individual clusterings
    - calculate adjusted Rand index and Misclassification Error rate [6] for each pair of individual clusterings
    - calculate the Misclassification Error rate for each clustered item (e.g. gene or cell) – for each number of clusters
* parameter
  + required
    - pandas dataframe: individual clusterings, one for each synthetic data set – stacked – as output by the program **gene clusters – individual and consensus**
* output value: list
  + pandas dataframe: adjusted Rand index for each pair of individual clusterings
  + pandas dataframe: Misclassification Error rates for each pair of individual clusterings
  + pandas dataframe: Misclassification Error rates for each clustered item; contains
    - one row per item
    - one column for each number of clusters

**ARI\_and\_Misclassification\_Error\_for\_individual \_clusterings\_wrt\_consensus\_clusters**

* purpose
  + compare individual clusterings with the consensus clusters derived from them
* parameters
  + required
    - pandas dataframe: individual clusterings, one for each synthetic data set – stacked – as output by the program **gene clusters – individual and consensus**
    - pandas dataframe: consensus clusters – as output by the program **gene clusters – individual and consensus**
* output value: list
  + pandas dataframe: adjusted Rand indexes
  + pandas dataframe: Misclassification Error rates
  + pandas dataframe: Misclassification Error rates for each clustered item (e.g. gene or cell) – for each number of clusters

**ARI\_and\_Misclassification\_Error\_for\_subset\_consensus\_clusterings**

* purpose
  + evaluate stability of consensus clusterings by calculating adjusted Rand index and Misclassification Error [6]. Recall that the program **gene clusters – individual and consensus** returns 3 pandas dataframes containing consensus clusterings
    - a single clustering for each number of clusters
    - two sets of clusterings for each number of clusters – the first derived from half of the individual spectral clusterings each corresponding to a synthetic data set; the second derived from the other half of the data
    - four sets of clusterings for each number of clusters – each derived from one quarter of the individual spectral clusterings
* parameter
  + required
    - pandas dataframe: a collection of two or more consensus clusterings – stacked as output by the program **gene clusters – individual and consensus**
* output value: list
  + pandas dataframe: adjusted Rand index for each pair of consensus clusterings
  + pandas dataframe: Misclassification Error for each pair of consensus clusterings

**genes\_mean\_proximities**

* purpose
  + compute the mean of all proximity arrays generated by the random forest classifications
  + this is used to
    - calculate the gap statistic
    - re-order clusters for presentation in heat maps
* parameters
  + required
    - folder containing input data sets
  + optional
    - prefix for input CSV data sets containing random forest proximities: default = 'gene\_proximities\_'
* input data sets
  + CSV files: proximities; integer: diagonal = number of trees – one file for each input synthetic data set
* output value: list
  + pandas dataframe: mean proximity array, normalized with diagonal equal to 1.

**gap**

* purpose
  + calculate gap statistic [7] with distance defined as (1 - mean proximity)
* parameters
  + required
    - pandas dataframe: mean proximity array
    - pandas dataframe: consensus clusterings
  + optional
    - number of shuffles – default = 100
* output value: list
  + pandas dataframe containing
    - gap statistic
    - gap plus/minus (one and two times) estimated standard deviation
    - lagged “gap – STD” and difference used to identify the appropriate number of clusters
  + number of shuffles

**data\_prep\_for\_RF\_cells**

* purpose
  + specify a number of gene clusters – for dimension reduction
  + for each cell, calculate the mean of the standardized null residuals for each gene cluster
  + prepare synthetic data sets for random forest classification
* parameters
  + required
    - pandas dataframe: standardized null residuals
    - pandas dataframe: gene consensus clusterings
    - specified number of gene clusters
    - folder to which output data sets will be written
  + optional
    - number of synthetic data sets to prepare: default=100
    - data set name for genuine data output: default = ‘gene\_means\_tr’
    - data set name prefix for synthetic data: default = ‘gene\_means\_tr\_synth\_’
* output value: list
  + pandas dataframe: gene cluster means – genuine data
* output data sets
  + CSV file : transpose of array containing the mean of the standardized null residuals for each gene cluster – genuine data; this has one row for each cell, one column for each gene cluster
  + CSV files: synthetic data sets for random forest classification (randomized transposed gene cluster means)

**RandomForestClassifier\_for\_proximity\_by\_cell**

* purpose
  + perform classification with random forests to obtain proximities between cells
  + the classification problem is to distinguish between genuine and synthetic data, as described in references [2, 3, and 4]
* parameters
  + required
    - number of trees in each random forest
    - folder containing input data sets
    - folder to which output data sets will be written
  + optional
    - name of input CSV data set containing standardized null residuals – for genuine data: default= ‘gene\_means\_tr’
    - prefix for input CSV data sets containing synthetic data: default = ‘gene\_means\_tr\_synth \_’
    - prefix for output CSV data sets containing random forest proximities: default = ‘cell\_proximities\_'
    - first synthetic data set to be input: default=0
* input data sets
  + CSV file : transpose of array containing the mean of the standardized null residuals for each gene cluster – genuine data; this has one row for each cell, one column for each gene cluster
  + CSV files: synthetic data sets for random forest classification (randomized transposed gene cluster means)
* output value: list
  + largest index of the synthetic data sets analyzed; for example, if 100 sets of individual clusterings are to be calculated, the value returned is 100
* output data sets
  + CSV files: proximities; integer: diagonal = number of trees

**cell\_clusters\_\_individual\_and\_consensus**

* purpose
  + calculate spectral clusterings [5] for each set of random forest proximities; refer to these as “individual” clusterings – as opposed to “consensus” clusterings
  + calculate
    - consensus clusterings based on all individual clusterings
    - two sets of consensus clusterings – each based on half of the individual clusterings
    - four sets of consensus clusterings – each based on one quarter of the individual clusterings
  + the variation between the clusterings derived from a quarter or half of the individual clusterings will give estimates of the consensus clusters’ stability
* parameters
  + required
    - maximum number of clusters to be computed
    - folder containing input data sets
  + optional
    - prefix for input CSV data sets containing random forest proximities: default = ‘cell\_proximities\_'
* input data sets
  + CSV files: proximities; integer: diagonal = number of trees – one file for each input synthetic data set
* output value: list
  + pandas dataframe: individual clusterings – these are stacked in a single dataframe
  + pandas dataframe: consensus clusterings derived from all individual clusterings
  + pandas dataframe: two sets of consensus clusterings, each derived from nonoverlapping halves of the individual clusterings; these are stacked, and distinguished by a counter in the column ‘cluster\_set’
  + pandas dataframe: four sets of consensus clusterings, each derived from nonoverlapping quarters of the individual clusterings; these are stacked, and distinguished by a counter in the column ‘cluster\_set’

**cells\_mean\_proximities**

* purpose
  + compute the mean of all proximity arrays generated by the random forest classifications
  + this is used to
    - calculate the gap statistic
    - re-order clusters for presentation in heat maps
* parameters
  + required
    - folder containing input data sets
  + optional
    - prefix for input CSV data sets containing random forest proximities: default = 'cell\_proximities\_'
* input data sets
  + CSV files: proximities; integer: diagonal = number of trees – one file for each input synthetic data set
* output value: list
  + pandas dataframe: mean proximity array, normalized with diagonal equal to 1.

**data\_prep\_and\_heat\_maps**

* purpose
  + for specified numbers of gene and cell clusters, calculate mean of proximities for each set of clusterings
  + re-order clusters for visualization using Ding’s algorithm for reordering leaves in a dendrogram [8]
  + re-order genes and cells for presentation
  + generate image files
    - gene and cluster mean proximities – as ordered initially by clustering algorithm and then re-ordered with Ding’s algorithm
    - heat maps for standardized binomial deviance null residuals
      * clipped: values greater than 1 are set to 1; values less than -1 are set to -1
      * percentiles: data in each row – corresponding to genes – are represented as percentiles
* parameters
  + required
    - data\_descriptor – string used in heat map title
    - specified numbers of gene clusters
    - specified numbers of cell clusters
    - pandas dataframe: standardized null residuals
    - pandas dataframe: gene consensus clusterings
    - pandas dataframe: cell consensus clusterings
    - pandas dataframe: gene mean proximity array
    - pandas dataframe: cell mean proximity array
    - folder to which graphic output PNG files will be written
* output value: list of lists
  + list containing two arrays: mean proximities for gene and cell clusters, for specified clusterings
  + list containing two lists: permutations for re-ordering gene and cell clusters
  + list containing three pandas dataframes: re-ordered data frames of genes, cells, and standardized expression data
* graphic output PNG files

these are written to the folder specified by the last (ninth) input parameter; file names are fixed

* + representations of mean proximities for gene and cell clusters
    - ordered as generated by consensus clustering algorithm; file names:
      * gene mean proximities.png
      * cell mean proximities.png
    - reordered by Ding’s algorithm; file names:
      * gene mean proximities reordered.png
      * cell mean proximities reordered.png
  + heat maps - clustered standardized binomial deviance null residuals; file names:
    - heat map - clip-1.png
    - heat map - percentiles.png

**Laplacian\_scores**

* purpose
  + calculate Laplacian scores [9] for filtered genes using mean random forest proximity matrix
* parameters
  + required
    - pandas dataframe: cell mean proximity array
    - pandas dataframe: standardized null residuals
* output value: list
  + pandas dataframe: Laplacian scores

**References**

1. Townes F W, Hicks S C , Aryee M J et al.: Feature selection and dimension reduction for single-cell RNA-Seq based on a multinomial model. *Genome Biol* 20, 295 (2019). <https://doi.org/10.118s13059-019-1861-6>
2. Breiman L, Cutler A: Random forests—Classification manual. *URL*  http://www.math.usu.edu/~adele/forests/
3. Shi T, Horvath S: Unsupervised learning with random forest predictors. *Journal of Computational and Graphical Statistics*2006, **15:**118-138.
4. Hastie T, Tibshirani R, Friedman J: The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition; section 14.2.4 “Unsupervised as Supervised Learning”
5. Ng A, Jordan M, Weiss Y: On spectral clustering: analysis and an algorithm. In T. Dietterich, S. Becker, and Z. Ghahramani (Eds.), *Advances in Neural Information Processing Systems* 14 (pp. 849 – 856). MIT Press, Cambridge (2002)
6. Meila M: The stability of a good clustering. <https://stat.washington.edu/sites/default/files/files/reports/2014/tr624.pdf>
7. Tibshirani R, Walther G, Hastie T: Estimating the number of clusters in a data set via the gap statistic. *Journal of the Royal Statistical Society*: Series B (Statistical Methodology). 63 (2): 411–423. doi:10.1111/1467-9868.00293. ISSN 1467-9868.
8. Ding C: Analysis of gene expression profiles: class discovery and leaf ordering. Proc. Conf. Research in *Comp.Mol.Bio* (RECOMB 2002), pp.127-136. April 2002, Washington, DC.
9. He X, Cai D, Niyogi P: Laplacian score for feature selection. In *Advances in neural information processing systems*. 2006: 507-514.