4-MAT USER MANUAL

Summary :

Introduction :

The 4-MAT program (standing for *4-Methods Analyzing Tool / 4-Methods Analysis of Transcriptome*) has been developed with the purpose of finding relevant associations between genes via transcriptomic data.
Initially conceived for associating genes of interest (from now on called *anchor genes*) who play a known role in given metabolic pathways and others *candidate genes* who are not yet known to have a role in those pathways, 4-MAT can also do a broader unfocused analysis and find associations between any couple of genes in a global pool.

Rules of use :

4-MAT is written in the Python language. Make sure a Python interpreter (like IDLE) is installed on the computer you're using to run the program as well as the following Python modules : time, collections, sys, copy, NumPy, matplotlib.pyplot, scikit-learn, pandas, SciPy, networkx, pickle, warnings.
NOTE : The underlined modules are installed by default with Python IDLE.

The command line must be written with the following arguments in this particular order :
- The *python3* key word, or any other key word used to declare a python script run.
- The program script proper.
- The parameters text file.
- At least one dataset csv file (unless you set the parameters file to only run the Consensus Step, in which case no dataset is required).

Ex : *python3 4-MAT_2025_09_18.py Parameters.txt Data_2-3-5-6_16428.csv*

WARNING : The GitHub depository is organized by file types (code files, data files, example result files) but for an optimal run, make sure to regroup all used files in a same folder.

When using multiple datasets, make sure all of them contain the same genes and the same number of genes (more details on the datasets' structure below).

When customizing the settings in the arguments file, make sure that all settings that are number-of-dataset dependent are correctly set according to the number of datasets you are using. Same goes for settings that are number-of-list-of-anchor-genes dependent.

The program is composed of 4 Analysis Steps plus a Consensus Step. You can customize the argument file to specify which steps you want to run or not. All steps that you'll run will produce one or several result files (details below).

All datasets must be structured in the following way :
- The first column must be called 'ID_REF' and contain the names of all genes to be analyzed. The order doesn't have to be the same in all datasets but all datasets must contain the entirety of the global gene pool, no more no less.
- The next columns can be called whatever you want. They must contain finite float values (no NaN or inf). The number of value columns can change from one dataset to another. As a general rule of thumb, the more columns there are, the more reliable the results will be.
- All datasets must use a coma separator (,) between it's columns and a dot (.) to indicate the decimal part of numerical values.

Initial Data and Settings :

By giving 4-MAT one or several datasets, each containing transcriptomic values for a same single gene pool, 4-MAT will apply up to 4 different analysis methods to find relevant associations between genes from the pool. If one or several lists of specific *anchor genes* are given, 4-MAT will create a consensus around those genes and sort all associations between them and all other *candidate genes* found by each method. If no such list of *anchor genes* is provided, 4-MAT will sort all associations between any couple of genes found by each method. The more methods a specific association is found by, the more relevant it ought to be.

The initial settings are the following :
- The **Global gene pool** is the list of all genes whose transcriptomic data are in the datasets you will provide. This list is used twice during the whole running of 4-MAT : once when sorting the anchor genes and once during the P-CEN method.

- The **Anchor gene lists** are the optional lists of specific genes whose associations you want the program to focus on. For each of these lists, 4-MAT will intersect it with the global gene pool and any gene present in the intersection will be labeled as an *anchor*.
- The **Anchor Genes labels** are optional names you can give to the *anchor genes*. By default, the program labels each anchor gene lists as "Anchor-1", "Anchor-2" and so on. If no list of *anchor genes* is provided, all genes will be labeled "Candidate".
- The **Anchor colors** are optional colors for the program to use when building the resulting networks for each method. By default, all genes, be they anchors or not, will be colored black. These colors are to be given in an RGB format (i.e. sets of 3 values between 0 and 1), and there has to be as many colors as you provided lists of anchor genes. If colors are given, only the *candidate genes* will be colored black.
- The **Step list** is the list of methods you want to activate when launching a 4-MAT run. You can ask for a run with no methods activated by explicitly setting this argument to 0.
- The **Consensus step** setting is to indicate if you want a consensus of results to be created.
- The **Result folder name** setting is an optional name for the folder in which the result files will be put upon creation. By default, the folder is named "Result_4-MAT_[date of creation]_[time of launch]. If customized, the program will check if a folder with the provided name exists and will create it if not. You can also write the key word 'None' if you wish for the result files to be put the same folder as the program has been launched from.
- The **Linkage dictionary** is the name of a preexisting result dictionary. By default, the program will create a new empty dictionary, initiate it's keys with all focused genes (i.e. *anchor genes* or all genes, depending on the providing of lists of the formers) and it's values with empty sub-dictionaries. For each activated method, the dictionary will fill each key's sub-dictionary value with the code name of the method as the sub-key and the list of associated genes the method has found for this key as the sub-value. If the name of a preexisting dictionary is provided, any activated method will overwrite the previous results found for this method.

The linkage dictionary's general structure looks like this :

- ❖ Anchor gene 1 :
  - ➢ Method A : [candidate 1.1a , candidate 1.2a , candidate 1.3a , ...]
  - ➢ Method B : [candidate 1.1b , candidate 1.2b , candidate 1.3b , ...]
  - ➢ Method C : [candidate 1.1c , candidate 1.2c , candidate 1.3c , ...]
  - ➢ Method D : [candidate 1.1d , candidate 1.2d , candidate 1.3d , ...]
- ❖ Anchor gene 2 :
  - ➢ Method A : [candidate 2.1a , candidate 2.2a , candidate 2.3a , ...]
  - ➢ Method B : [candidate 2.1b , candidate 2.2b , candidate 2.3b , ...]

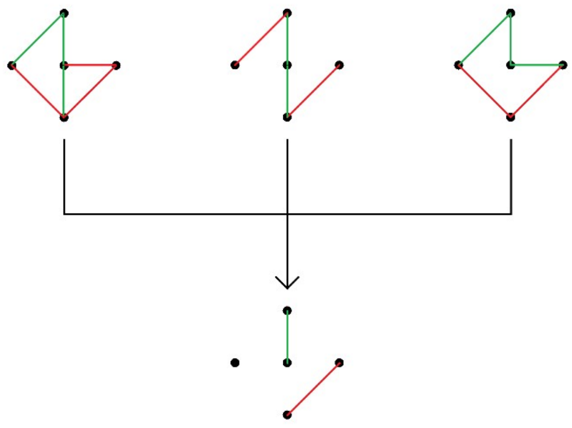
    - Method C : [candidate 2.1c , candidate 2.2c , candidate 2.3c , ...]
    - Method D : [candidate 2.1d , candidate 2.2d , candidate 2.3d , ...]
- Anchor gene 3 : ...

NOTE 1 : Some methods give more information than just the list of candidates found for each *anchor gene*. Those information are specified in the description of each method.

Method 1 – P-CEIN (*Pearson based Co-Expression Network*)

The *Pearson based CoExpression Intersected Network* method uses the Pearson's Correlation Coefficient (form now on abbreviated PCC) calculation to evaluate the association between genes.

NOTE : For reliable PCC values, this method only uses datasets with at least 3 timepoints (or more generally, 3 feature columns, not counting the 'ID_REF' first column). If you do provide several datasets and some of them have less than 3 timepoints, the program filters them out for the P-CEIN method only.



P-CEIN method illustration :
From 3 datasets with transcriptomic values for a same group of 5 genes, 3 networks are calculated, linking two genes if they have a significant Pearson's Correlation Coefficient (green edges for positively correlated genes, red edges for negatively correlated genes).
By intersecting all initial networks, only couples of genes showing similarly signed correlations across all networks are kept linked in the method's final network.

This method's settings are the following :

- The **Anchor centered P-CEIN** boolean indicates if you want the method to focus on the correlation between the anchors and the rest of the global pool (other anchors included) or if you want it to calculate the PCC between all couples of genes from the global pool. WARNING : deactivating this setting highly raises the method's run time.
- The **Pearson principal threshold** is an optional threshold used when first evaluating the PCCs. It is by default set to 0.5 (in absolute value) for all datasets and can go up to 1 (perfect correlations only) and down to 0 (all correlations). You can either give a single threshold that will be used for all datasets or give as many thresholds as the number of datasets you provided.
- The **Pearson dynamic threshold** boolean is used after the initial evaluation. If activated, the average PCC and standard deviation

(respectively abbreviated Avg and Std) of all kept couples will be calculated and a new threshold will be set. Any couple with a PCC lower than this threshold will be filtered out.

- The **Dynamic threshold factor** is the multiplicative value used in the formula to calculate the dynamic threshold. It is by default set to 1.
- The **Minimum neighborhood P-CEIN** is an optional setting used to filter out any candidate gene with less than a certain number of anchors linked to it. It is by default set to keep all candidates. If you want to activate this filter, you must give a minimum number of neighbors for each anchor list you have provided.
- The **Research time announcement P-CEIN** is a quality of life setting that writes on the console the run time for each analyzed gene.
- The **New Network** boolean indicates if you want to calculate the PCCs of your genes from scratch or if you want to load a preexisting network created by a previous run of the method. Be sure not to change the file's name between it's creation and it's subsequent loading.

The P-CEIN method's execution is as following :

1) It either loads a preexisting network or calculates a new one. When calculating a new network, it first sorts for each provided dataset the value vectors of each gene based on the global gene pool order. If the **Anchor centered** setting is activated, P-CEIN calculates for each *anchor gene* it's correlation to all the other genes. Otherwise, it calculates for each gene it's correlation to all the remaining genes of the pool (because the PCC between two vectors is symmetrical, it is pointless to go through the entire list for each next gene). For each couple of genes, P-CEIN calculates the PCC of the couple in each of the provided datasets. The couple is then saved in a network if it clears two conditions : it's PCCs are all higher than the **principal threshold** in absolute value, and they are all of the same sign. The edge representing the couple in the network is given a weight equal to the average of the couple's PCCs and a color based on the PCC's sign.
2) The newly built network is saved in a file and the general data of each gene are saved as well.
3) If the **Pearson dynamic threshold** is activated, the average weight of all edges in the network is calculated as well as their standard deviation and a second threshold is calculated via the formula "P = Avg + **factor***Std". All edges with a weight lower than this new threshold are filtered out. If after this filtering, any gene ends up with no neighbors, it is deleted from the network as well. The resulting network and it's general data are then saved.
4) If the **Minimum neighborhood P-CEIN** is customized, the program will go through all *candidate genes* and delete any who doesn't have enough *anchor genes* among it's neighbors. When several lists of *anchor genes* are provided, a *candidate gene* only needs to respect the quota of neighbors of one list to be kept in the network. It is only deleted if it

doesn't respect the quota for all anchor lists. The resulting network and it's general data are then saved.

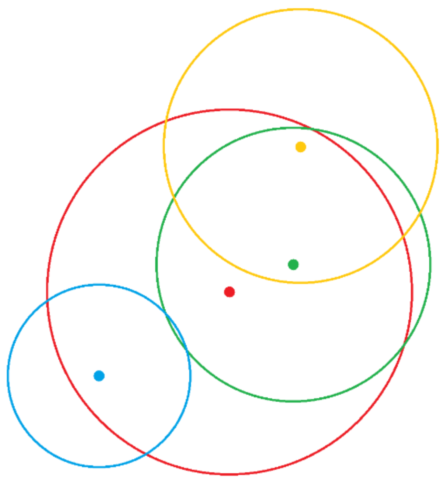
5) Finally, the last state of the network is used to fill the result **Linkage dictionary**. For each gene referenced in the dictionary at the launching or the program, a list of it's neighbors along the values of their respective PCCs (in real value) is add to the sub-dictionary of the referenced gene.

The linkage dictionary's specific structure for the P-CEIN method looks like this :

- Anchor gene 1 :
  - P-CEIN : [(candidate 1.1 , average PCC 1.1) , (candidate 1.2 , average PCC 1.2) , ...]
- Anchor gene 2 :
  - P-CEIN : [(candidate 2.1 , average PCC 2.1) , (candidate 2.2 , average PCC 2.2) , ...]

<u>Method 2 – KNN-RBH (*K-Nearest Neighbors enhanced with Reciprocal Best Hit*)</u>

The *K-Nearest Neighbors enhanced with Reciprocal Best Hit* method combines the ideas of the K-Nearest Neighbors algorithm (form now on abbreviated KNN) and the Reciprocal Best Hit concept (from now on abbreviated RBH) to evaluate the association between genes.



<u>KNN-RBH method illustration :</u>
In a dataset with transcriptomic values, a K-Nearest Neighbors algorithm determines a neighborhood distance threshold for each gene.
Couples of genes that are inside each other's neighborhood distance are then associated (here, that would mean the "red-green" couple and the "green-yellow" couple).
If several datasets are provided, couples need to be associated in all datasets to be kept in the method's final result.

This method's settings are the following :

- The **KNN version** allows to choose between two KNN calculations : Version *KNN_1* is the classic KNN algorithm that looks for the same number of k neighbors for all genes. Version *KNN_2* is a dynamic variant of the algorithm where a number of k neighbors is chosen for each gene separately.

- The **RBH sub-step** boolean indicates if you want the calculated neighborhoods to be reciprocal or not. If activated, any couple of genes where one considers the other a neighbor but the other way around is not true will not be kept as an association.
- The **Number of neighbors** is an optional value only used with Version *KNN_1*. It is the value of k neighbors to look for each gene. The default value is 2.
- The **Threshold factor** is the multiplicative value only used with Version *KNN_2* to calculate the dynamic k value. It is by default set to 1.

The KNN-RBH method's execution is as following :

1) It calculates each gene's neighborhood according the chosen KNN version. In version 1, all genes end up with the same number of k neighbors given by the **Number of neighbors** parameter. In version 2, the program calculates for each gene the Euclidean distance between it and all other genes, get the average distance and the standard deviation, then determines a threshold distance using the formula "D = Avg - **factor***Std". A given gene's neighborhood consists then in all genes whose distances are shorter than the threshold distance.
2) Once all neighborhoods are calculated, if the **RBH sub-step** is activated, the program goes through each gene's neighborhood and looks if the current gene is also in the neighborhood of each of it's neighbors. If both genes consider each other neighbors, they stay in their respective neighborhood. If not, they are deleted from the respective neighborhood.

<u>NOTE :</u> Steps 1 and 2 run once for each provided dataset.

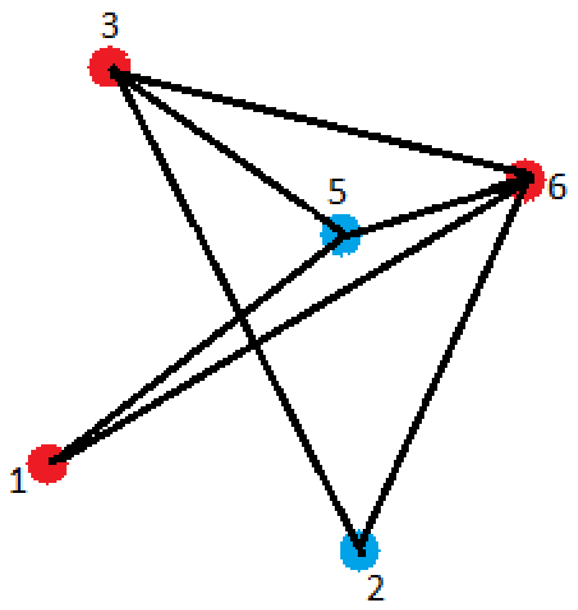
3) Once all datasets have been analyzed, KNN-RBH crosses for each gene it's neighborhoods from all datasets. A network is then built with an edge linking two genes that consider each other a nearest neighbor in all provided datasets. The edges are weighted with the average distance calculated between it's two genes in all datasets.
4) Once fully built, the network is saved and used to fill the **Linkage dictionary**. For each gene referenced in the dictionary at the launching or the program, a list of it's neighbors along the values of their respective average distances are added to the sub-dictionary of the referenced gene.

The linkage dictionary's specific structure for the KNN-RBH method looks like this :

- ❖ Anchor gene 1 :
  - ➢ KNN-RBH : [(candidate 1.1 , avg distance 1.1) , (candidate 1.2 , avg distance 1.2) , …]
- ❖ Anchor gene 2 :
  - ➢ KNN-RBH : [(candidate 2.1 , avg distance 2.1) , (candidate 2.2 , avg distance 2.2) , …]

Method 3 – NPC (*Network Properties Closeness*)

The *Network Properties Closeness* method uses network properties to evaluate the association between genes. This method's core execution has been developed by Gabriel Dominico, PhD student at the Federal University of Rio Grande do Sul in Brazil.



NPC method illustration :
In a dataset with transcriptomic values, a network is built, linking special genes (in red) to themselves and to candidate genes (in blue).
Those candidates are then sorted according to how much they are linked to the special genes (here, gene 5 would be sorted higher because it is linked to 3 special genes, one of which (gene 6) is part of two different cliques (clique 3-5-6 and clique 1-5-6), while gene 2 is linked to 2 special genes and is only in one clique).

This method's settings are the following :

- The **Filter version** allows to choose between two versions of a test to filter the edges in the initial network that is used to analyze the properties : Version *Unilateral* will keep an edge if it's distance is within the threshold of at least one of it's node. Version *Bilateral* will keep it only if only if it's distance is within both nodes' threshold.
- The **Distance threshold factor** is the multiplicative value used to calculate the nodes' distance thresholds. It is by default set to 1.
- The **Minimum neighborhood NPC** is used to filter out any *candidate gene* with less than a certain number of *anchors* linked to it. It can either be a manually chosen value or the word 'Dynamic'. In the later case, the program will calculate the average number of *anchor* neighbors per *candidate* and the standard deviation to determine a dynamic value for the filter using the formula "N = Integer(Avg + Std)".
- The **Minimum redundancy** is an optional value that indicates the minimum number of datasets a candidate must be found in for it to be considered relevant from the method's point of view. By default, candidates are kept only if they are found in all provided datasets for at least one *anchor gene* list.

NOTE : This method is the only one among all 4-MAT methods that explicitly needs *anchor genes* lists. It cannot run in the context of an unfocused analysis.

The NPC method's execution is as following :

1) It normalizes the dataset's values and calculates the Euclidean pairwise distances between all genes and sort them in ascending order while remembering the indices of origin.
2) It then build an initial network with each edge weighting the distance between the two genes it connects. This network differentiates between *anchor genes* and *candidate genes*.
3) Each edge is then submitted to a first filter for either keeping it in or deleting it from the network. This filter is based on calculating a distance threshold for each connected gene by use of the formula "D = Avg - **factor***Std" and see if the edge's distance checks at least one (**Unilateral version**) or the two (**Bilateral version**) thresholds.
4) Each *candidate gene* is next submitted to a second filter based on the **Minimum neighborhood NPC** setting. If a *candidate* counts among it's connections less *anchors genes* than setting's value, it is removed from the network.
5) Each remaining *candidate gene* is then ranked based on network properties such as (but not limited to) it's number of cliques, the size of it's largest clique and number of time it's connected to an *anchor gene* for each clique.

NOTE : Steps 2 to 5 run one time for each *anchor genes* list provided. If multiple *anchor genes* lists are provided, keep in mind that for each run, *anchor genes* from another list than the one actively used in the current run will be considered *candidate genes* from the method's point of view. This inconvenience is corrected later in the program and as such, is of no direct consequence.

6) Once all datasets and all *anchor genes* lists have been analyzed, NPC crosses the results and dresses for each of the method's candidates the list of *anchor genes* it has been found associated to, filtering out all candidates found in less than the **Minimum redundancy** threshold. All kept candidates (and their associations) are then used to build a final network where edges between two genes are more or less wide based on the redundancy of the corresponding associations.
7) Once fully built, the network is saved and used to fill the **Linkage dictionary**. For each gene referenced in the dictionary at the launching or the program, sub-lists of it's neighbors are add to the sub-dictionary of the referenced gene. Each sub-list contains candidates with the same redundancy. There are as many sub-lists as there are provided datasets.

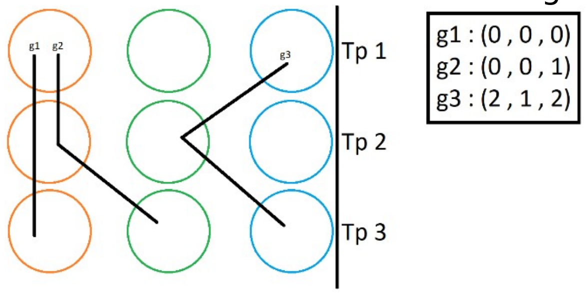
The linkage dictionary's specific structure for the NPC method looks like this :

❖ Anchor gene 1 :

    - NPC : [[candidate 1.1 , candidate 1.2] , [candidate 1.3] , ...]
- Anchor gene 2 :
    - NPC : [[candidate 2.1] , [candidate 2.2] , [candidate 2.3] , ...]

Method 4 – Cluster Path

The *Cluster Path* method uses several K-Means clustering instances to evaluate the association between genes.



Cluster Path illustration :
In a dataset with transcriptomic values, each timepoint is submitted to a K-Means algorithm to sort the genes in k clusters (here, k=3). Each gene is associated to a vector representing which cluster the gene is in at which timepoint.
Couples of genes are associated by the method if they more or less follow the same path (here, genes g1 and g2 could be associated if the parameters allow 1 difference).

This method's setting are the following :
- The **Number of clusters** is used to indicate how many clusters must be calculated by the K-Means algorithm. It is by default set to 3.
- The **Maximum deviation** is an optional value used to indicate how many timepoints are allowed to show differences along a cluster path. It is by default set to 0 (meaning that two associated genes follow the exact same path of clusters). You can either give a single value to be used across all the provided datasets (if you don't care where the deviations can occur) or give as many values as the number of dataset you provided (each dataset will have to respect it's respective value).
- The **Anchor centered CP** boolean indicates if you want the method to focus on the deviant paths between the anchors and the rest of the global pool (other anchors included) or if you want it to calculate the deviant paths between all couples of genes from the global pool. WARNING : deactivating this setting highly raises the method's run time.
- The **Research time announcement CP** is a quality of life setting that writes on the console the run time for each analyzed gene.

The Cluster Path method's execution is as following :
1) For each timepoint of a dataset, a 1-dimension K-Means algorithm is used to sort the genes according to the **Number of clusters** parameter. The clusters are then arranged in ascending order of their centers' values. Each gene is then associated with a vector where each value represents one of the clusters, a so-called "cluster path".
2) Once all timepoints have been analyzed, the genes are regrouped by common "cluster path".

NOTE : Sub-steeps 1 and 2 run once for each provided datasets. The results from each run is saved for the following sub-steps.

3) Once all datasets have been analyzed, the genes are regrouped common "cluster paths" in all datasets, meaning that genes that stay together in a common path, regardless of dataset and even if the path differs from a dataset to another, are grouped together. These groups of genes are then saved in a file.
4) Next, the path of each gene (selected according to the **Anchor centered CP** parameter) is confronted to the path of each other gene and a new vector is built indicating in which timepoints the genes share a same cluster and in which timepoints they are separated (those are "deviant" timepoints). If the number and distribution of deviant timepoints respect the **Maximum deviation** parameter, the new "deviant path" is saved and is associated to the list of couples of genes that follow it. These "deviant paths" and their lists of couples are then saved in a file.
5) Once all groups are formed, a network is built with edges between genes that are in the same group. Once fully built, the network is saved and used to fill the **Linkage dictionary**. For each gene referenced in the dictionary at the launching or the program, a list of it's neighbors along the values of their respective path's identity ratio (i.e. the number of non-deviant timepoint in the path) is add to the sub-dictionary of the referenced gene.

The linkage dictionary's specific structure for the Cluster Path method looks like this :

- ❖ Anchor gene 1 :
  - ➢ ClusterPath : [(candidate 1.1 , ratio 1.1) , (candidate 1.2 , ratio 1.2) , …]
- ❖ Anchor gene 2 :
  - ➢ ClusterPath : [(candidate 2.1 , ratio 2.1) , (candidate 2.2 , ratio 2.2) , …]

Methods Consensus

When given a non-empty **Linkage dictionary**, the Consensus part of 4-MAT crosses the results from all methods for all genes acting as keys in the dictionary. For each couple of associated genes, a consensus level is determined based on how many methods the association has been found by. Three files are then written to sum up the final results :

- An anchor-based file, where for each *anchor gene* are sorted it's associated candidates by descending Consensus level order. This file's data are non-redundant, meaning that if a first *anchor* has a second *anchor* among it's candidate, the second *anchor* doesn't have the first *anchor* among it's. At the end of the file, the total number of associations of each Consensus level is written.

- A candidate-based file, where all associations are described. Each line informs about the two genes concerned by the association, the label of the *anchor gene* (only if *anchor genes* have been provided), the Consensus level, and for each method a boolean indicating if a given method has found the association.
- A custom graph file, firstly listing all genes along with a color corresponding to their nature (candidate or anchor), and secondly listing all edges (representing the associations) along with the Consensus level, and for each method, either a 'NO' if the method didn't find the association or the value of finding if the method did find it (values being the PCC for the P-CEIN method, the distance for the KNN-RBH method, a number of datasets for the NPC method and the path identity ratio for the Cluster Path method).

Result files description :

During a run, 4-MAT will generate several result files. With the exception of two, all result files are exclusive to one of the methods or to the consensus step and once created are not modified further in the code.

The file called ***Result_0_dictionary.pickle*** is the default name of the *Linkage dictionary* (described in the **Initial data and settings** section). It is either created at the beginning of the program's run or provided by customizing it's setting. One of the two files who are used several times during the program's run, it receives the results of each called method at the end of their respective run and is read during the consensus step. If the *Linkage dictionary* has been provided and already contains result for a method that is run once again, the new result will replace the previous one.

P-CEIN files :
All P-CEIN method result files have their names starting with ***Result_1[x]_P-CEIN*** and can be sorted in two ways :
- A sorting by file type. It can either be .csv tabs or .txt graphs.
- A sorting by checkpoints. The [x] in the template name above can either be the character 'a', 'b' or 'c' and correspond to checkpoints after the 3rd, 4th and 5th steps of the method.

The files called ***Result_1a_P-CEIN_data.csv, Result_1b_P-CEIN_dynamic_data.csv*** and ***Result_1c_P-CEIN_relevant_neighborhood_data.csv*** contain the general data for all genes retained after the corresponding checkpoints. For each gene is indicated it's :
- name
- label (either a kind of *anchor gene* or a *candidate*)
- number of associated *anchor genes* from all kinds
- number of total associations

- number of positive associations
- number of negative associations
- average absolute PCC value
- standard deviation of absolute PCC value
- maximum absolute PCC value
- minimum absolute PCC value

The files called ***Result_1a_P-CEIN_graph.txt***, ***Result_1b_P-CEIN_ dynamic_data.txt*** and ***Result_1c_P-CEIN_ relevant_neighborhood_data.txt*** allow to visualize the results as graphs. All lines starting with a '>' represent a gene to which is associated a color in rgb format. All other lines represent a couple of associated genes and indicate the couple's PCC absolute value, sign and color.

<u>NOTE :</u> Only the ***Result_1a_P-CEIN*** files are always created. The ***1b*** and ***1c*** files are created only if the **Pearson dynamic threshold** setting and the **Minimum neighborhood P-CEIN** setting are respectively activated.

<u>KNN-RBH file :</u>
The file called ***Result_2_KNN-RBH_graph.txt*** allows to visualize the results as a graph. All lines starting with a '>' represent a gene to which is associated a color in rgb format. All other lines represent a couple of associated genes and indicate the couple's Euclidean distance.

<u>NPC files :</u>
All NPC method result files have their names starting with ***Result_3[x]_NPC*** and are sorted by checkpoints. The [x] in the template name above can either be the character 'a', 'b', 'c' or 't' and correspond to checkpoints after the 5th and 6th steps of the method.

The files called ***Result_3a_NPC_[Label]_for_Dataset_[x]_min[n].txt*** are the ranking tabs from each loop of steps 2 to 5. The [Label] indicates which *anchor genes* list has been used, the [x] indicates which dataset has been analyzed and the [n] indicates the value of the **Minimum neighborhood NPC** setting. For each kept *candidate gene* is indicated it's :
- number of unique associated *anchor genes*
- score (a value that represents how many times the *candidate* is associated to the *anchors*. It is calculated by counting for each clique the number of associated *anchor genes* and then sum all counts. It may differ from the number of unique associated *anchor genes* because a given associated *anchor* can be found in more than one clique and be counted more than once).
- number of cliques
- size of maximal clique
- minimum distance
- average distance

- maximal distance
- list of unique associated *anchor genes*

The file called ***Result_3t_NPC_CandidateLists_file.txt*** is a auxiliary file where all ***Result_3a_NPC_[Label]*** file names are sorted according to their [Label]. It has no direct value to the user but is read by the program in order to create the next file.

The file called ***Result_3b_NPC_CandidateGenes.txt*** contains the compiled results from all ***Result_3a_NPC*** files. For all *candidate genes* that pass the **Minimum redundancy** filter for at least one kind of *anchor* label, is indicated it's :

- name
- list of numbers of unique associated *anchor genes* (one number per dataset, one list per *anchor* label)
- total number of unique associated *anchor genes* across all datasets and all *anchor* labels
- lists of unique associated *anchor genes* sorted by association redundancy (1st list = *anchor genes* associated in only one dataset, 2nd list = *anchor genes* associated in two datasets, and so on...). If a given list contains *anchor* genes of different labels, anchors of a same label are grouped together and separated from the others. The word 'None' indicates an empty list for the given redundancy.

The file called ***Result_3c_NPC_graph.txt*** allows to visualize the results as a graph. All lines starting with a '>' represent a gene to which is associated a color in rgb format. All other lines represent a couple of associated genes and indicate the couple's association redundancy (i.e. the number of datasets in which the association has been found).

<u>Cluster Path files :</u>
All Cluster Path method result files have their names starting with ***Result_4[x]_ClusterPath*** and are sorted by checkpoints. The [x] in the template name above can either be the character 'a', 'b' or 'c' and correspond to checkpoints after the 3rd, 4th and 5th steps of the method.

The file called ***Result_4a_ClusterPath_PathList.txt*** contains the recap of all encountered paths. All lines starting with a '>' indicates :

- a path (or a collection of sub-paths if several datasets have been provided. The sub-paths are arranged in the same order as their respective dataset has been provided)
- the total number of genes that follow that path
- the numbers of *anchor genes* from each *anchor* list that follow that path

All other lines are the lists of genes that follow the path detailed by the previous line.

The file called ***Result_4b_ClusterPath_DeviantPathsList.txt*** contains the recap of all encountered paths that respect the **Maximum deviation** parameter (including all paths from in the previous file that regroup two or more genes). All lines starting with a '>' indicates :

- a path (or a collection of sub-paths if several datasets have been provided. The sub-paths are arranged in the same order as their respective dataset has been provided)
- the identity ratio of that path
- the total number of couples that follow that path

All other lines are the lists of couples of genes that follow the path detailed by the previous line.

The file called ***Result_4c_ClusterPath_graph.txt*** allows to visualize the results as a graph. All lines starting with a '>' represent a gene to which is associated a color in rgb format. All other lines represent a couple of associated genes and indicate the path the couple has been found in.

Consensus files :
All Consensus result files have their names starting with ***Result_END***.

The file called ***Result_END_consensus.csv*** contain the general data for all *anchor genes*. For each gene is indicated it's :

- name
- *anchor* label
- number of total unique associated *candidate genes* across all methods
- number of unique associated *candidate genes* found by exactly one method
- number of unique associated *candidate genes* found by exactly two methods
- number of unique associated *candidate genes* found by exactly three methods
- number of unique associated *candidate genes* found by exactly four method
- number of unique associated *candidate genes* found by the P-CEIN method (further sorted by the sign of the PCC value [i.e. positive or negative])
- number of unique associated *candidate genes* found by the KNN-RBH method
- number of unique associated *candidate genes* found by the NPC method (further sorted by their dataset redundancy)
- number of unique associated *candidate genes* found by the Cluster Path method

NOTE : In the case where you run 4-MAT in two separate instances with different methods each time (for example, running KNN-RBH and NPC first and then running P-CEIN and Cluster Path second), the order of the data will match the order in which the methods have been run for the first time (in the same example, you'd have the data in the order of KNN-RBH, NPC, P-CEIN and Cluster Path).

The file called ***Result_END_GenesOfInterest_LIstByAnchors.txt*** indicates for each *anchor gene*, the lists of associated *candidates* sorted by consensus level (i.e. the number of methods the associations have been found by). At the bottom of the file is written the total number of association for each consensus level.

The file called ***Result_END_GenesOfInterest_LIstByCandidates.txt*** indicates the consensus vectors for all associations. For each association, a line indicates :

- the *candidate gene*
- the *anchor gene* (the associations are sorted by the *anchors*)
- the label of the *anchor gene* (*)
- a 'Yes'/'No' tag for if the association links two anchors (*)(**)
- the consensus level
- the consensus vector : for each method, a '1' means that the method has found the association and a '0' means it hasn't. P-CEIN is the only method with two values, differentiating the positive and negative possibility of the method's result.

(*) : the 3rd and 4th columns only appear if at least one *anchor genes* list has been provided.
(**) : associations tagged as 'Yes' in the 4th column exist in 2 copies in the tab, one for each anchor serving a candidate for it's associated anchor.

The file called ***Result_END_graph.txt*** allows to visualize the results as a graph. All lines starting with a '>' represent a gene to which is associated a color in rgb format. All other lines represent a couple of associated genes and indicate the consensus level as well as the values from each method. If a method hasn't found the association, the corresponding value is 'NO'. If it has, the value is :

- the real PCC value for the P-CEIN method
- the distance value for the KNN-RBH method
- the dataset redundancy for the NPC method
- a 'YES' for the Cluster Path method

The file called ***RunLog.txt*** is the second file used several times during the program's run. It records the run's progression.