The NEST is designed to predict the gene essentiality based on protein interaction network and gene expression or CRISPR screen results. However, you can replace the network and expression matrix with any forms of data, as long as their identifiers match each other. If you find "NEST" useful, please cite us (http://NEST.dfci.harvard.edu/publication/). If you find any bugs or have any questions, welcome to contact us (peng.jiang.software@gmail.com).

Installation:
The installation is very easy on Unix like system, such as Linux and Mac OS. Within the folder, please type the followings:
./configure
make
make install

You may need "sudo make install" in the third step. If you don't have super user privilege, you can install in a local folder by changing the configure step:
./configure --prefix=path_to_local_folder

For example, I do "./configure --prefix=/Users/peng" on my laptop.

Usage:
As an example, after getting into "data" folder (cd data), please type the followings:
NEST -i string.9606 -m CUTLL1.GSI_wash -o output

All input and output files are tab delimited. The usage is like this:
NEST -i network -m matrix -o output

Format of input and output:
Each line of "network" represents an interaction like this "ID_A ID_B confidence" or "IDA IDB" without the optional confidence fields.

The "matrix" could be gene expression, CRISPR screen scores ("CRISPR.K562"), or any matrix with identifiers matched with network. The column names are sample names and row names are gene names (or any names aligned with network identifiers). Please don't put any extra column names such as "ID". The number of columns should be the same as the number of data values.

The "output" is a data matrix in the same format as input "matrix". The values are predicted gene essentiality scores.

CRISPR/shRNA screen analysis:
In genome-wide pooled screen, each gene may have several CRISPR gRNAs (or shRNAs) available. The input matrix of NEST requires each gene to have one unique value in each condition. Thus, we provide a simple tool "merge_matrix" to summarize gene level scores among several gRNAs (or shRNAs).

Usage:
As an example, after getting into "data" folder (cd data), please type the followings:
merge_matrix -i CRISPR.K562.sgRNA -o CRISPR.K562.sgRNA.merge

The general usage is like this:
merge_matrix -i input -o output [Options]

Options:

-m Merge method. Default: median
median: median of all values
mean: mean of all values
top3: mean of top three values by absolute
second: second best values by absolute

-c Minimum number threshold. Default: 3

The input matrix could be gRNA/shRNA fold change in the screen, or any matrix with identifiers matched with network. The column names are sample names with one extra column as gRNA index (or sequence). As an example, please see "CRISPR.K562.sgRNA" in data folder.

The option "-m" controls how to merge several gRNA (or shRNA) values for the same gene. There are many methods developed for this task. However, according to our experience, the default "median" will give very reasonable result. The option "-c" controls the minimum number of gRNAs (or shRNAs) required for each gene. If the number of available guides is lower than this threshold, the gene will be skipped in final result.

Advanced Usage: Calculating statistical significance.
For predicted gene essentiality, you can compute statistical significance (Z-score and *P*-value) by permutation test. First of all, you need to permute the input network by stub-rewiring method. Then, each predicted score is compared with the average value calculated with random networks to derive *Z*-score and *P*-value.

As an example, please get into "data" folder and type the followings:

stub_rewire -i string.9606 -o random/string.9606.random
NEST_fast -i random/string.9606.random -m CRISPR.K562 -o output
NEST_summarize output

For the first program "stub_rewire", the usage is like this:

Usage: stub_rewire -i input -o output [OPTIONS]

Options:

-n Number of randomizations. Default: 1000
-r Retry count. Default 10
-c Connected threshold. Default: 0.98
-m Compact mode: 1 (yes) or 0 (no). Default: 1 (yes)

The input network is treated as unweighted and the confidence score will be ignored if available, because the stub-rewiring method only preserves the unweighted degree of each node. The output includes a bunch of binary files including the original network, random networks and identifier name map. All self-interactions and duplicated interactions will be removed.

The option "-n" sets the number of random networks, which is 1000 by default. Since we don't allow self-interaction and duplicated interactions, the stub-rewiring process is not guaranteed to finish. The option "-r" sets the number of retries if the stub-rewiring process fails. The option "-c" sets the minimum fraction of edges that should be successfully rewired.

The "-m" controls whether the output files are written in binary format, which is compact in space (Appendix A for detail). If you override the default value 1 and set it as 0, ASCII files will be generated, which are readable for human, but not usable by program "NEST_fast".

For the second program "NEST_fast", the usage is like this:

Usage: NEST_fast -i network -m matrix -o output [OPTIONS]

Options:

-n Number of randomizations. Default: 1000
-w Write intermediate results: 1 (yes) or 0 (no). Default: 0 (no)

The "network" is the prefix of random networks generated by "stub_rewire". The "matrix" is your gene expression or CRISPR screen value. All of these inputs are the same as "NEST". The "output" is the prefix of a set of files, which contains the predicted gene essentiality scores, network degrees, *Z*-scores and *P*-values (Appendix B for detail). If you want to join these output files together for ease of view, you can do "NEST_summarize output".

The option "-n" controls the number of randomizations used, which should be the same as the value for "stub_rewire". The "-w" controls whether the predicted scores for random networks are written. You can set this flag as 1 if you want the values from random networks.

Appendix A: format of binary network output.

For the names of all network nodes, we assign them continuous integer indices starting from 0. The each name index, we use two bytes (16 bits) to represent it. Thus, the largest possible index is 2^16 – 1 = 65535. If you only need to represent genes in human or mouse, this upper limit is adequate. However, if you want a larger range, you can contact me, I can easily modify the program for you. For each interaction, we write down two bytes index for each partner in output.

Appendix B: calculation of Z-score and *P*-value for predicted gene essentiality.

For each random network, the gene essentiality scores are predicted. Z-score of a predicted value is computed as: (real value – average random value)/(stderr of random values). The *P*-value is calculated as the fraction of random values that are greater than or equal to the real value if the Z-score is possible; or the fraction of random values that are less than or equal to the real value if the Z-score is negative.