Manual of AP-PLN

Table of Contents

System Requirements

Running Automatic Pipeline to build Phenotypic Linkage Network (AP-PLN) requires at least Python version > 2.7.6 and R version > 3.1.2. The following Python package must be installed: numpy (numpy version 1.9.1). The following R packages must be installed: earth (> 4.3.3), igraph (1.1.2) and biomaRt (2.30.0). As no compilation is required, the pipeline can be used on any computer, where Python and R are installed and it is therefore available for Windows, Linux, and MAC OS machines.
License: AP-PLN is an open source and distributed under the GNU General Public License v3.0 (http://www.gnu.org).

Download and Installation

AP-PLN is available from http://csandorfr.github.io/AP-PLN/. Download the ZIP file or the TAR ball file, unzip/extract the download, and save it to your favorite directory for your applications. To uncompress the TAR ball file, type the following command:

```
tar xvzf xxxAP-PLNxxx.tar.gz
```

A directory AP-PLN is created in your favorite applications directory. It contains a subdirectory including all scripts: src/. Next, move into the directory AP-PLN (cd AP-PLN).

AP-PLN requires that you download and extract in your AP-PLN directory the following TAR ball file including pipeline data: data.tar.gz

We also provide two examples of the set that you need to download and to extract in your AP-PLN directory:
example.tar.gz
example_t2d.tar.gz

Finally, you will need to configure an environment variable, named $AP_PLN_HOME by typing the following command: export AP_PLN_HOME=directory of your AP_PLN directory

Overview

A fundamental analysis in AP-PLN allowing the user to design his own phenotypic benchmark consists of three steps (**Figure 1**):

- Semantic similarity scores are computed between gene pairs from a specified list of Human Phenotype Ontology (HPO) [1] or Mouse Phenotype Ontology (MPO) [2] annotations (**module 1 pipeline_compute_phen_sem_sim.py**).
- Different functional datasets considered by the user to build the final PLN are re-scaled on a unique phenotypic benchmark (**module 2 pipeline_evaluation_rescore.py**)
- The different re-scaled functional datasets are combined to construct the final phenotypic linkage network (PLN) (**module 3 pipeline_integration.py**).

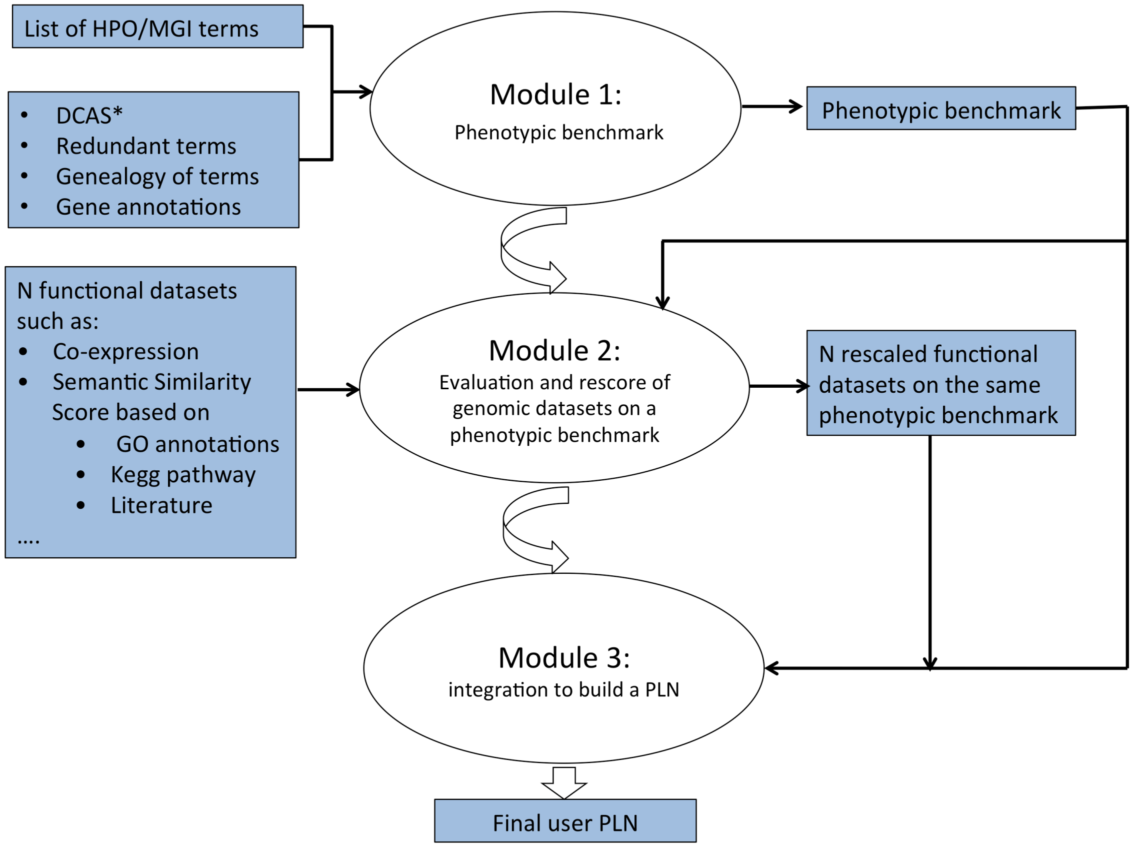
List of HPO/MGI terms
DCAS*
Redundant terms
Genealogy of terms
Gene annotations
Module 1:
Phenotypic benchmark
Phenotypic benchmark
N functional datasets such as:
Co-expression
Semantic Similarity Score based on
GO annotations
Kegg pathway
Literature
....
Module 2:
Evaluation and rescore of genomic datasets on a phenotypic benchmark
N rescaled functional datasets on the same phenotypic benchmark
Module 3:
integration to build a PLN
Final user PLN

Figure 1: Flowchart of the pipeline to design automatically a phenotypic linkage network

Quick Start

In your $AP_PLN_HOME/src/quick-start, you can run a python script that allows:

(1) to combine two functional datasets:
  - coexpr_gse3594 (Co-expression dataset based on the microarray expression profile (GSE5394))
  - sem_goabp (Semantic similarity scores based on Gene Ontology Biological Process annotations of human and mouse genes)

(2) to compare these two functional and the final PLN in term of gene coverage and accuracy
(3) to visualize the final PLN for a genes set
(4) to evaluate the functional clustering of a gene set

In the directory: $AP_PLN_HOME/example by using the following command:

```
python run_all_pipeline.py $AP_PLN_HOME/example/semantic_sim_gene_mgi_all.scale
$AP_PLN_HOME/example/list_file_data $AP_PLN_HOME/example $AP_PLN_HOME/example
/well/webber/users/cynthias/CW016_SANDOR_AP_PLN/AP-PLN/example/list_gene quick_example
```

To speed up this test, the phenotypic benchmark used to re-scale each functional dataset was pre-computed and available in: $AP_PLN_HOME/example: semantic_sim_gene_mgi_all.scale

Module1: design phenotypic benchmark

Python script

pipeline_compute_phen_sem_sim.py

Directory

$AP_PLN_HOME /src/module1_phen_sem_sim

Description

This python script enables the user designing his own phenotypic benchmark by computing semantic similarity scores between gene pairs from his list of HPO or MPO reference phenotype annotations.

Details

The module 1 pipeline_compute_phen_sem_sim.py consists in the next steps:

- To re-annotate the genes with only the MPO or HPO phenotype annotations and their child terms provided by the user.
- To estimate a measure of information content as described by Honti et al. [3] for each phenotypic annotation reflecting the specificity of terms
- To calculate the Resnik's similarity measure [4] between terms organized in a hierarchical ontology, defining the semantic similarity between any two terms t1 and t2 as the average IC of their disjunct common ancestor terms by using GraSM approach [5].
- To measure the functional relatedness of two genes by comparing their annotations with the maximum (max) and best match average (bma) methods [6].

Options

1 < *File providing the list of MGI or HPO relevant phenotype annotations* >

2 < *Directory of output files (output directory)* >

3 < *Suffix of output file* >

Example with MPO terms

```
python pipeline_compute_phen_sem_sim.py $AP_PLN_HOME/example_t2d/list_t2d_mgi_term.txt
$AP_PLN_HOME/example_t2d t2d_mgi
```

Example with HPO terms

```
python pipeline_compute_phen_sem_sim.py $AP_PLN_HOME/example_t2d/list_t2d_hpo_term.txt
$AP_PLN_HOME/example_t2d t2d_hpo
```

Computational performances

Table 1: Computational performances for two examples

Dataset	Number of terms	Number of Pairs Genes	Time
MPO	3	5277664	48 mn
HPO	8	830591	45 mn

Tested on high specification computer, 3.6-GB RAM and two 3.16-GHz Intel Core2 Duo CPUs. HS: 148-GB RAM and 24 2.67-GHz Intel Xeon CPUs

Input files

- File providing the list of MPO or HPO relevant phenotype annotations. The user must provide the MPO/HPO phenotype annotations according to their MP/HP accession number (see http://www.informatics.jax.org/searches/MP_form.shtml and http://human-phenotype-ontology.github.io/tools.html).

Output files

- <*m1_suffix output file*>.log a log file that summarizes the parameters used for the specific analysis run, and that lists the different steps
- < *suffix output file*>.scale, tab delimited file of the phenotypic similarity score, with the following columns: (1) gene1, (2) gene2 and (3) semantic similarity score. The semantic similarity score is scaled between zero and one and the gene pairs are sorted in ascending order according to the phenotypic semantic score.

Module 2: evaluation and re-score of each dataset on a phenotypic benchmark

Python script

pipeline_evaluation_rescore.py

Directory

$AP_PLN_HOME/src/module2_evaluation_rescore

Description

The module2 pipeline_evaluation_rescore.py consists of two main steps:

- To evaluate each dataset on a phenotypic benchmark by estimating the ability of a particular functional dataset to identify genes more likely to influence the same phenotype. The relation between a score derived from the functional dataset and the semantic phenotypic similarity measure computed with the module is examined. The gene pairs are sorted according to data-specific scores (supposed proportional to the strength of functional information) in descending order. The ordered pairs are divided into bins of x gene pairs and the median (or mean) of the phenotypic semantic similarity scores between gene pairs is calculated and plotted for each bin. The degree of phenotypic similarity expected for random gene pairs is equal to the median/mean of all gene pairs (yrandom). A plot is created and used to determine what strength of functional relation (linkage) within each functional dataset is informative to predict phenotypic similarity and to rescale the informative part of this last on the phenotypic benchmark.

- To re-score each dataset on a phenotypic benchmark: from (1) a threshold is derived based on low uninformative linkages from a given dataset and informative bins that are defined as those above the overall median of the semantic phenotypic similarity measure. To determine this threshold, a multivariate adaptive regression splines (MARS) analysis models the relationship between the score derived from the functional dataset (x) and the phenotypic semantic similarity score (y), and the x-intercept of this MARS with yrandom. Above this threshold, we fit a different MARS curves to re-score the links so that any data-specific scores characterising the gene pairs are replaced with the semantic similarity of phenotypes that they correspond to according to our MARS function.

Options

1 < *Phenotypic benchmark (Module 1)* >

2 < *File with the functional datasets to be evaluated &rescaled* >

3 < *Directory of each functional dataset file* >

4 < *Directory of output files (output directory)* >

5 < *Bin size: Number of gene pairs per bin* >

6 < *0/1* > *Use either the mean or the median to estimate:*

- *the phenotypic similarity value associated with random gene pair*

- *to compute the phenotypic similarity score per bin*

7 < *Suffix of output file* >

Example

```
python pipeline_evaluation_rescore.py  $AP_PLN_HOME/example/semantic_sim_gene_mgi_all.scale
$AP_PLN_HOME/example/list_file_data  $AP_PLN_HOME/example  $AP_PLN_HOME/example 500
0 quick_example
```

Computational performances

Table 2: Computational performances of two functional datasets

Dataset	**Numbers of Pairs Genes**	**Time**
Semantic Similarity (GO BP)	5967478	2 mn 15 s
Co-expression (GSE3594)	2779725	1 mn 10 s

Tested on high specification computer, 3.6-GB RAM and two 3.16-GHz Intel Core2 Duo CPUs. HS: 148-GB RAM and 24 2.67-GHz Intel Xeon CPUs

Input files

The input files are: (1) phenotypic similarity score (phenotypic benchmark) and (2) list of functional datasets. (1) and each dataset listed in (2) are tab-delimited files; each row referring to a unique pair of genes; columns 1 and 2 are gene pairs and column 3 is the phenotypic similarity score or score derived from the functional dataset. We used here the ensemble gene ID. Furthermore, there are two prerequisites regarding the phenotypic similarity score or score derived from the functional dataset: (1) the scores must be scaled between zero and one (e.g. by using the function scale of R); (2) the gene pairs in each functional dataset must be in descending order according to their phenotypic semantic score or functional related score.

Output files

All output files will be printed in the output directory provided by the user (option 4).
The output files are:

- < *m2_suffix output file.log* > a log file that summarizes the parameters used for the specific analysis run and that lists the different steps performed
- < *functional dataset. suffix output file.eval.png* > is a plot used to determine what strength of functional relation (linkage) within each functional dataset is informative to predict phenotypic similarity and to rescale the informative part of this last on the phenotypic benchmark. The x-axis and y-axis correspond to the score derived from the functional dataset and the phenotypic similarity score, respectively. The horizontal dotted gray line shows the phenotypic semantic similarity score of random gene pairs, while a vertical blue line represents the threshold from which low uninformative linkages derived from a given dataset are identified. The curve represents the MARS model used to rescore the data (example **Figure 2**).
- < *functional dataset. suffix output file.rescore* > file with functional dataset rescored on the phenotypic benchmark. The format is identical to the input files.
- < *list_data_rescore.suffix output file* > list of the rescored functional datasets

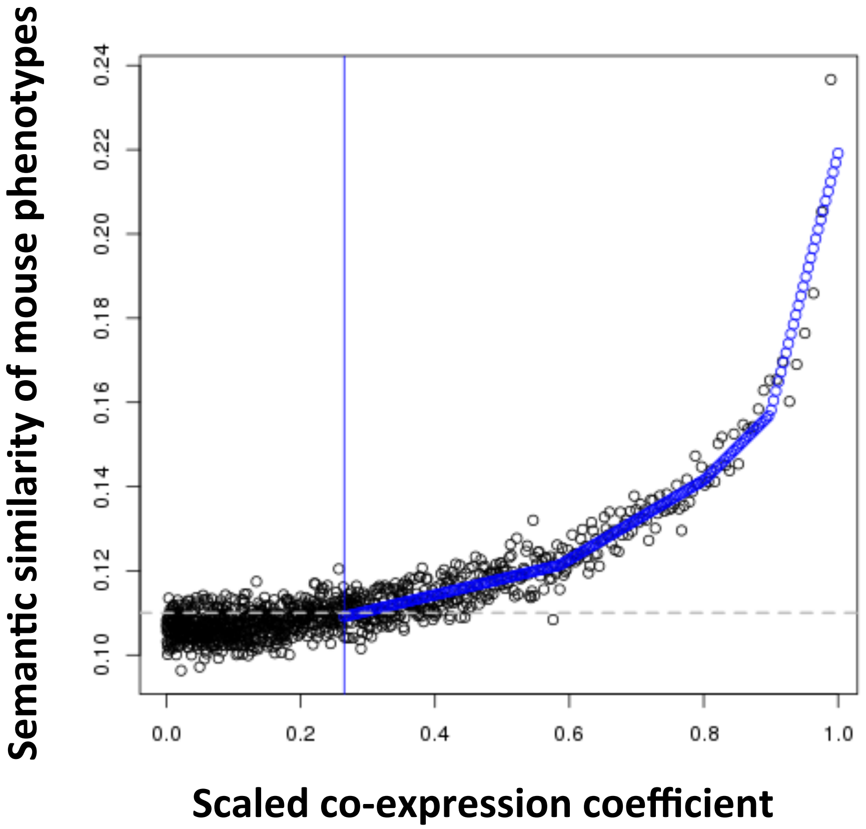

Figure 2: Example of a co-expression dataset of how each functional dataset is evaluated and rescored on a phenotypic benchmark. The depicted co-expression data (GSE3594) were ordered by the Pearson's correlation coefficient and divided into bins of 500 gene pairs (black dots). The blue multivariate adaptive regression splines curve represents the (re)scoring function that associates co-expression correlation coefficients with the corresponding semantic similarity measured with mouse phenotype annotations. The median of the semantic similarity values of 500 gene pairs has been calculated for each bin.

Module 3: Integration of re-scored functional dataset into single phenotypic linkage genes network

Python script

pipeline_integration.py

Directory

$AP_PLN_HOME/src/module3_integration

Description

The module3 pipeline_integration.py combines multiple functional datasets re-scored on a phenotypic benchmark. In the case where different genomic datasets suggest a functional link between the same gene pairs, the functional measures of the different dataset are summed by penalizing the less reliable data according to a formula proposed by Lee et al. [7]:

$$WS = L_o + \sum_{i=1}^{n} \frac{L_i}{D \times i}$$

, where L represents a re-scored functional measure from a single data set, L_0 being the largest functional measure among all the functional datasets between the given two genes, *i* is the index of the remaining links ordered by their weights for the gene pair and *D* is a free parameter. The value of *D* is optimized, by using the integrated dataset with a *D* parameter that is the best linear predictor for a phenotypic semantic similarity measure.

Options

1 *< Phenotypic benchmark (Module 1)>*

2 *< List of functional datasets rescaled on a phenotypic benchmark (Module 2) >*

3 *< Directory of each rescaled functional dataset file >*

4 *< Directory of output files (output directory) >*

5 *< Bin size: Number of gene pairs peer bin >*

6 *<0/1> Use either mean or median to estimate the phenotypic similarity score per bin*

7 *<Suffix of output file>*

Example

```
python pipeline_integration.py $AP_PLN_HOME/example/semantic_sim_gene_mgi_all.scale
$AP_PLN_HOME/example/list_file_rescore.quick_example $AP_PLN_HOME/example
$AP_PLN_HOME/example 500 0 quick_example
```

Computational performances

Table 3: Computational performances to build PLN with two rescored functional datasets

Dataset	Numbers of Pairs Genes	Max D	Time
Semantic Similarity (GO BP)	5967478	6	5 mn 15 s
Co-expression (GSE3594)	2779725		

Tested on high specification computer, 3.6-GB RAM and two 3.16-GHz Intel Core2 Duo CPUs. HS: 148-GB RAM and 24 2.67-GHz Intel Xeon CPUs

Input files

The input files are: (1) phenotypic similarity score (phenotypic benchmark) and (2) list of each functional datasets rescored on a phenotypic benchmark. (1) and each dataset in (2) are tab-delimited files; each row referring to a unique genes pair; columns 1 and 2 are gene pairs and column 3 is the phenotypic similarity score or score derived from the functional dataset. We used here the ensemble gene ID. Furthermore, there are two prerequisites regarding the phenotypic similarity score or score derived from the functional dataset: (1) the scores must be scaled between zero and one (e.g. by using the function scale of R); (2) the gene pairs in each functional dataset must be in descending order according to their phenotypic semantic score or functional related score.

Output files

- < *m3_suffix output file.log* > a log file that summarizes the parameters used for the specific analysis run, and that lists the different steps
- <*wl_suffix of output file.final.scale.ord*> Final phenotypic linkage network (PLN). It is a tab-delimited file; each row referring to a unique gene pair; columns (1) and (2) are gene pairs and column (3) is the functional weighted link measure. The gene pairs are in descending order according to their final functional weighted link measure.
- *best_parameter_d* reports the *D* parameter used.
- *the best_parameter_d.png* plot shows the evaluation of integrated measure with different *D* parameter.
- < *list_data_rescore_wl.suffix output file* > file with the list of functional rescaled datasets.

Additional tools

Comparison of accuracy and gene pairs coverage of multiple re-scaled functional datasets

The script $AP_PLN_HOME/src/additional_tools/comparison_dataset/pipeline_evaluation.py compares visually different functional rescaled datasets in terms of relative accuracy to predict phenotypic benchmark and gene pairs coverage.

This script takes seven arguments: (1) phenotypic benchmark file; (2) file with list of functional re-scaled datasets; (3) data directory; (4) output directory; (5) number of gene pairs by bin; (6) use mean or median (0 or 1); (7) prefix of output file

Example

```
python pipeline_evaluation.py $AP_PLN_HOME/example/semantic_sim_gene_mgi_all.scale
$AP_PLN_HOME/example/list_file_rescore_type_wl.quick_example $AP_PLN_HOME/example
$AP_PLN_HOME/example 2000 0 quick_example
```

The output file is a plot similar to the figure below:

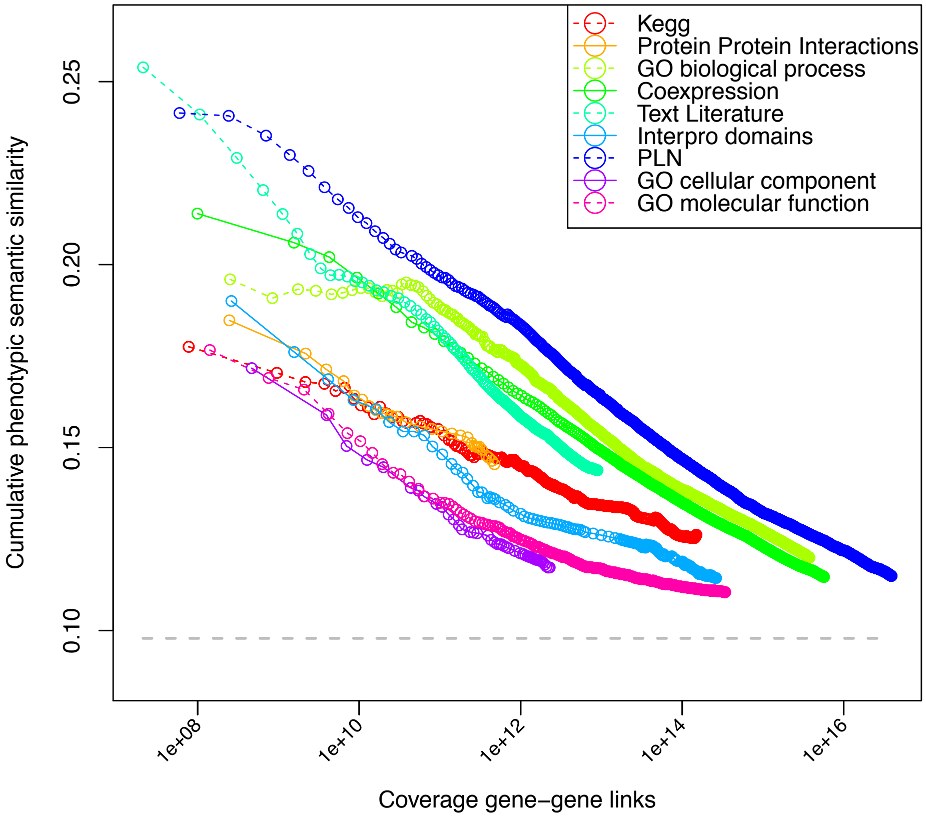

Figure 3: Comparison of information provided by different data types and by the final PLN. Different data types provide information of characteristic accuracy over different sets of genes. The Y-axis gives the semantic similarity of the phenotypes from the pairwise mouse model comparisons given the number of gene-gene links covered on the X-axis. The final PLN is depicted here in blue, was build

by using different type of functional dataset including (1) semantic similarity score base on Gene Ontology annotations, Kegg pathways annotations and pathway annotations of human genes from Reactome (2) protein-protein interaction dataset (3) Co-citations score (Text Literature) (4) protein domain annotations from InterPro (5) multiple expression dataset and described in details in (Honti, et al., 2014; Sandor, et al., 2017).

Network representation of PLN for a set of genes

The python script *$AP_PLN_HOME/src/additional_tools/make_network/ make_network_info.py* (1) extracts the functional associations within a PLN for a set of genes; (2) extracts the most contributing resources, among the integrated datasets to build a PLN, for any functional associations found; (3) represents the extracted subnetwork with links colored by the most contributing resource.

This script takes six arguments: (1) the list of genes as a one column (gene symbol annotation); (2) a phenotypic linkage network; (3) the list of functional rescored datasets; (4) directory of input files (5) directory of output files (6) the suffix of output files

The output files are: (1) the extracted subnetwork with links information; (2) network_representation_infoLinks.pdf, the plot of the links between input genes in the PLN with links colored by the most contributing resource (**Figure 4**); (3) net_igraph.Rdata, an igraph object.

Example

```
python make_network_info.py $AP_PLN_HOME/example/list_gene
$AP_PLN_HOME/example/wl_quick_example.final.scale.ord
$AP_PLN_HOME/example/list_file_rescore_type_wl_without_pln.quick_example
$AP_PLN_HOME/example $AP_PLN_HOME/example network_quick_example
```

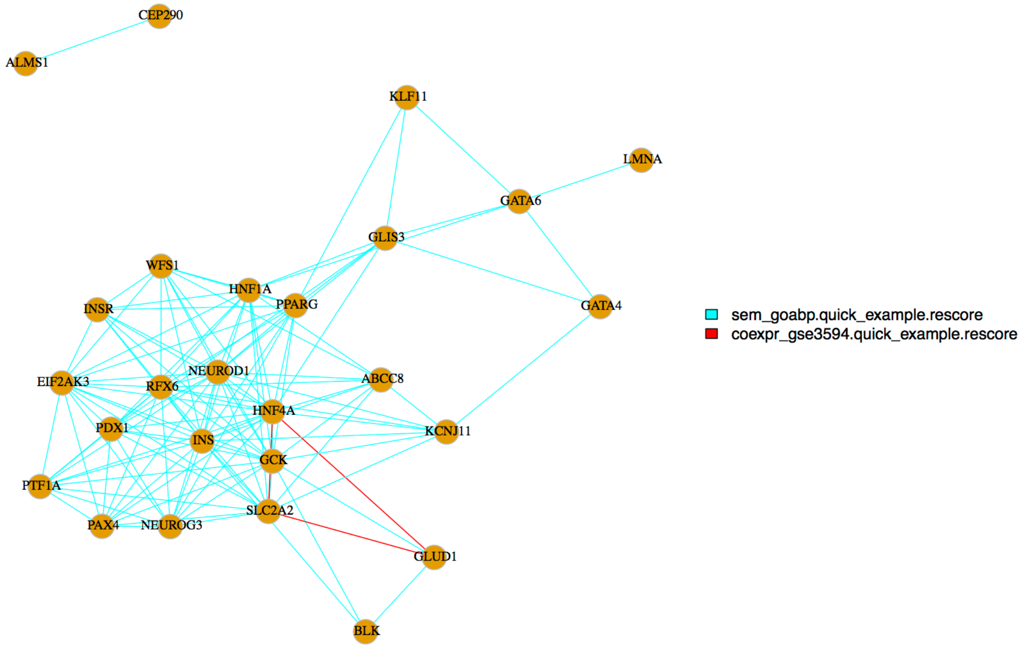

Figure 4: Example of network representation Network representation by using a generic PLN (built integrating the two functional datasets suggested in the quick_start) and genes associated with a list of monogenic and syndromic diabetes genes (S3 Table of Sandor et al. (2017) [8])

Functional clustering test of a gene-set

The python script *$AP_PLN_HOME/src/additional_tools/clustering_test/clustering_test.py* evaluates whether a given set of genes demonstrate unusually similar functionality. It examines the extent to which those genes cluster within the PLN. For this, it compares the sum of weighted links observed between these genes as compared to an equal number of random genes and computes an empirical p-value. As the coding-sequence (CDS) length and the network connectivity (degree) can bias functional network associations, the randomized genes are matched in CDS length and the degree to the original genes.

This script takes five arguments: (1) the list of genes as one column (gene symbol annotation); (2) a phenotypic linkage network; (3) directory of output files (4) the suffix of output files (5) the number of random sampling.

The output files are: (1) < *clustering_ suffix output.log*> the log file (2) < *suffix output_stat_clustering.pdf* > is plot representing the null distribution of sum of weighted links of random genes sets (black histogram) (**Figure 5**) and the sum of weighted link of tested set of genes (red line) (3) <*sim_suffixe_output_sum*> the sum of weighted link of tested set of genes (first row) and random genes sets (others row) (4) <*sim_suffixe_output_pval*> first column: number of random sampling, the sum of weighted link of tested set of genes and the empirical pval

Example:

```
python clustering_test.py /well/webber/users/cynthias/CW016_SANDOR_AP_PLN/AP-PLN/example/list_gene /well/webber/users/cynthias/TEST_PIPELINE/AP-PLN/example/wl_quick_example.final.scale.ord /well/webber/users/cynthias/TEST_PIPELINE/AP-PLN/example quick_example 1000
```

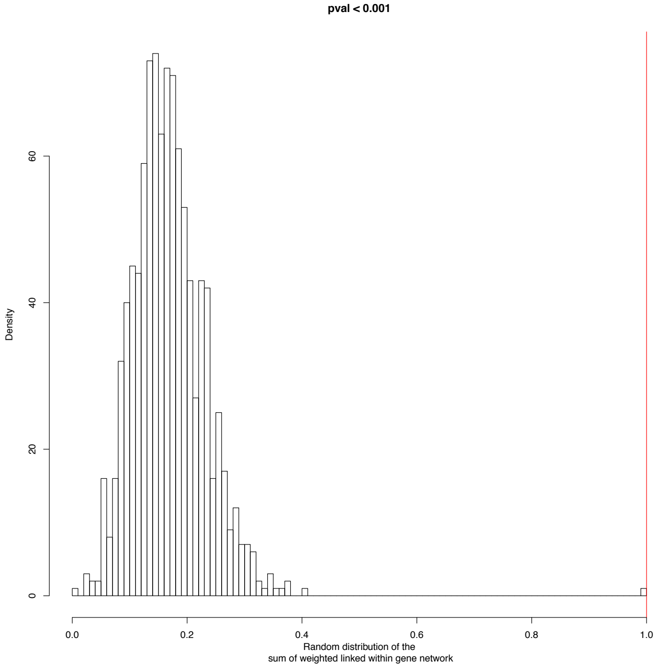

Figure 5: Testing for functional clustering of a gene-set

The black histogram represents the null distribution of the sum of weighted links of random genes sets (black histogram), while the red vertical line corresponds to the sum of the weighted link of the tested set of genes. We performed perform range standardization of sum of weighted links and the empirical p-value is given in the title.

References

1. Robinson PN, Kohler S, Bauer S, Seelow D, Horn D, Mundlos S. The Human Phenotype Ontology: a tool for annotating and analyzing human hereditary disease. Am J Hum Genet. 2008;83(5):610-5. doi: 10.1016/j.ajhg.2008.09.017. PubMed PMID: 18950739; PubMed Central PMCID: PMCPMC2668030.

2. Blake JA, Bult CJ, Eppig JT, Kadin JA, Richardson JE, Mouse Genome Database G. The Mouse Genome Database: integration of and access to knowledge about the laboratory mouse. Nucleic Acids Res. 2014;42(Database issue):D810-7. doi: 10.1093/nar/gkt1225. PubMed PMID: 24285300; PubMed Central PMCID: PMCPMC3964950.

3. Honti F, Meader S, Webber C. Unbiased functional clustering of gene variants with a phenotypic-linkage network. PLoS Comput Biol. 2014;10(8):e1003815. doi: 10.1371/journal.pcbi.1003815. PubMed PMID: 25166029; PubMed Central PMCID: PMCPMC4148192.

4. Philip R. Using information content to evaluate semantic similarity in a taxonomy. Proceedings of the 14th international joint conference on Artificial intelligence - Volume 1 %@ 1-55860-363-8, 978-1-558-60363-9. Montreal, Quebec, Canada: Morgan Kaufmann Publishers Inc.; 1995. p. 448-53.

5. Couto FM, Silva MJ, Coutinho PM. Measuring semantic similarity between Gene Ontology terms. Data & Knowledge Engineering. 2007;61(1):137-52. doi: 10.1016/j.datak.2006.05.003.

6. Pesquita C, Faria D, Bastos H, Ferreira AE, Falcao AO, Couto FM. Metrics for GO based protein semantic similarity: a systematic evaluation. BMC Bioinformatics. 2008;9 Suppl 5:S4. doi: 10.1186/1471-2105-9-S5-S4. PubMed PMID: 18460186; PubMed Central PMCID: PMCPMC2367622.

7. Lee I, Blom UM, Wang PI, Shim JE, Marcotte EM. Prioritizing candidate disease genes by network-based boosting of genome-wide association data. Genome Res. 2011;21(7):1109-21. doi: 10.1101/gr.118992.110. PubMed PMID: 21536720; PubMed Central PMCID: PMCPMC3129253.

8. Sandor C, Beer NL, Webber C. Diverse type 2 diabetes genetic risk factors functionally converge in a phenotype-focused gene network. PLoS Comput Biol. 2017;13(10):e1005816. doi: 10.1371/journal.pcbi.1005816. PubMed PMID: 29059180.