GENIE3-SG-sep and GENIE3-SG-joint: documentation

Author: Vân Anh Huynh-Thu, vahuynh@uliege.be

This is the documentation for the python implementations of GENIE3-SG-sep and GENIE3-SG-joint. These implementations are research prototypes and are provided "as is". No warranties or guarantees of any kind are given.

The GENIE3-SG-sep and GENIE3-SG-joint methods are described in the following book chapter:
Huynh-Thu V. A., Wehenkel L., and Geurts P. (2013) Gene regulatory network inference from systems genetics data using tree-based methods. In: A. de la Fuente (Ed.), *Gene Network Inference - Verification of Methods for Systems Genetics Data*, Springer, pp. 63-85.

1 Installation

To be able to run GENIE3-SG-sep and GENIE3-SG-joint, you have to install the **scikit-learn** module ($\geq$ version 0.16). Instructions for installing sciki-learn are provided here:
`http://scikit-learn.org/dev/install.html`

2 Load the GENIE3_sysgen package

```
from GENIE3_sysgen import *
```

3 Load example data

GENIE3-SG-sep and GENIE3-SG-joint are meant to be run on gene expression data and genotyping data, **where each gene has one genetic marker and the genetic markers can have two possible values only (0 or 1)**. Two files 'expr_data.txt' and 'gen_data.txt', containing examples of datasets, are provided for this tutorial. To load this data:

```
expr_data = loadtxt('expr_data.txt',skiprows=1)
gen_data = loadtxt('gen_data.txt',skiprows=1)
f = open('expr_data.txt')
gene_names = f.readline()
f.close()
gene_names = gene_names.rstrip('\n').split('\t')
```

- *expr_ data* is an array containing the expression data of 10 genes in 50 individuals. Expression data must be continuous values.
- *gen_ data* is an array containing the genotype data of these 10 genes in the same 50 individuals. Genotype data must be 0/1 values.
- *genes_ names* is a list containing the names of the genes.

4 Run GENIE3-SG-sep

Run GENIE3-SG-sep with its default parameters

The only mandatory input arguments to the function $GENIE3\_SG\_sep()$ are the expression matrix and the genotype matrix (note that the function name contains underscores instead of hyphens):

```
(VIM_expr,VIM_gen,VIM_sum,VIM_prod) = GENIE3_SG_sep(expr_data,gen_data)
```

$GENIE3\_SG\_sep()$ returns 4 arrays (VIM_expr, VIM_gen, VIM_sum, and VIM_prod), each containing scores for the putative regulatory links.

- $VIM\_expr(i,j)$ is the score of the expression of the i-th gene in the prediction of the expression of the j-th gene.
- $VIM\_gen(i,j)$ is the score of the genotype of the i-th gene in the prediction of the expression of the j-th gene.
- $VIM\_sum(i,j) = VIM\_expr(i,j) + VIM\_gen(i,j)$
- $VIM\_sum(i,j) = VIM\_expr(i,j) \times VIM\_gen(i,j)$

Restrict the candidate regulators to a subset of genes

```
# Genes that are used as candidate regulators
regulators = ['CD19', 'CDH17','RAD51','OSR2','TBX3']
(VIM_expr2,VIM_gen2,VIM_sum2,VIM_prod2) = ...
  GENIE3_SG_sep(expr_data,gen_data, ...
    gene_names=gene_names,regulators=regulators)
```

In the four returned arrays, the links that are directed from genes that are not candidate regulators have a score equal to 0.

Change the tree-based method and its settings

```
# Use Extra-Trees method
tree_method='ET'

# Number of randomly chosen candidate regulators at each node of a tree
K = 7

# Number of trees per ensemble
ntrees = 50

# Run the method with these settings
(VIM_expr3,VIM_gen3,VIM_sum3,VIM_prod3) = ...
  GENIE3_SG_sep(expr_data,gen_data, ...
    tree_method=tree_method,K=K,ntrees=ntrees)
```

Obtain more information

```
help(GENIE3_SG_sep)
```

5 Run GENIE3-SG-joint

Run GENIE3-SG-joint with its default parameters

The only mandatory input arguments to the function $GENIE3\_SG\_joint()$ are the expression matrix and the genotype matrix (note that the function name contains underscores instead of hyphens):

```
(VIM_expr,VIM_gen,VIM_sum,VIM_prod) = GENIE3_SG_joint(expr_data,gen_data)
```

$GENIE3\_SG\_joint()$ returns 4 arrays (VIM_expr, VIM_gen, VIM_sum, and VIM_prod), each containing scores for the putative regulatory links.

- $VIM\_expr(i,j)$ is the score of the expression of the i-th gene in the prediction of the expression of the j-th gene.
- $VIM\_gen(i,j)$ is the score of the genotype of the i-th gene in the prediction of the expression of the j-th gene.
- $VIM\_sum(i,j) = VIM\_expr(i,j) + VIM\_gen(i,j)$
- $VIM\_sum(i,j) = VIM\_expr(i,j) \times VIM\_gen(i,j)$

Run GENIE3-SG-joint with other parameter values

The parameters of the function $GENIE3\_SG\_joint()$ are the same as for the function $GENIE3\_SG\_sep()$ (see Section 4).

Obtain more information

```
help(GENIE3_SG_joint)
```

6 Write the predictions

Get the predicted ranking of all the regulatory links

```
get_link_list(VIM_prod)
```

The output will look like this:

```
## G1 G5 0.527481
## G5 G1 0.517994
## G6 G8 0.384952
## G8 G6 0.343328
## G9 G10 0.320162
## G2 G8 0.257154
## G9 G7 0.240072
## ...
```

Each line corresponds to a regulatory link. The first column shows the regulator, the second column shows the target gene, and the last column indicates the score of the link.
If the gene names are not provided, the i-th gene is named "Gi".
Note that the ranking that is obtained will be slightly different from one run to another. This is due to the intrinsic randomness of the Random Forest and Extra-Trees methods. The variance of the ranking can be decreased by increasing the number of trees per ensemble.

Important note on the interpretation of the scores: The weights of the links returned by $GENIE3\_SG\_sep()$ and $GENIE3\_SG\_joint()$ **do not have any statistical meaning** and only provide a way to rank the regulatory links. There is therefore no standard threshold value, and caution must be taken when choosing one.

Show the names of the genes

```
get_link_list(VIM_prod,gene_names=gene_names)
```

```
## TBX3 XRCC2 0.527481
## XRCC2 TBX3 0.517994
## CD93 CREB5 0.384952
## ...
```

Show only the links that are directed from the candidate regulators

```
get_link_list(VIM_prod,gene_names=gene_names,regulators=regulators)
```

Show the first 5 links only

```
get_link_list(VIM_prod,gene_names=gene_names,regulators=regulators, ...
  maxcount=5)
```

Write the predicted links in a file

```
get_link_list(VIM_prod,gene_names=gene_names, ...
  regulators=regulators,file_name='ranking.txt')
```

Obtain more information

```
help(get_link_list)
```