dynGENIE3: documentation

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

This is the documentation for the MATLAB implementation of dynGENIE3. This implementation is a research prototype and is provided "as is". No warranties or guarantees of any kind are given.

The dynGENIE3 method is described in the following paper:
Huynh-Thu V. A. and Geurts P. (2017) dynGENIE3: dynamical GENIE3 for the inference of gene networks from time series expression data. *Scientific Reports* 8:3384.

1 Installation

- Add the directories "RT" and "dynGENIE3_MATLAB" into MATLAB's path:

```
path(path,'your_path/RT');
path(path,'your_path/dynGENIE3_MATLAB');
```

- To check that everything works, type the following in the MATLAB console:

```
rtexample
```

  You should get no error and a mean square error around 4.8.

- If you get an error of this type:

  ```
  ??? Undefined function or method "rtenslearn_c" for input
  arguments of type "struct"
  ```

  then you must compile the function RT/rtree-c/rtenslearn_c.c using the command mex provided with MATLAB

```
mex rtenslearn_c.c
```

  and copy the resulting mex file into the directory RT.

  Note that this command will not work if gcc is not installed on your computer.

For Windows users who encounter some difficulties, a tutorial is available from:
`http://gnumex.sourceforge.net/`

For Windows 64 bits, you may need to install:

1. Microsoft Windows SDK for Windows 7 and .NET Framework 4:
   `http://www.microsoft.com/download/en/details.aspx?displaylang=en&id=8279`
2. Visual C++ 2010 Express:
   `http://www.microsoft.com/visualstudio/en-us/products/2010-editions/visual-cpp-express`

2 Example data

A file 'example_data.mat' containing some data is provided for this tutorial. To load this data:

```
load example_data.mat
```

- `SS_data` is an array containing the expression levels of 10 genes in 100 steady-state conditions.
- `TS_data` is a cell array, each cell corresponding to a time series experiment (here 3 time series experiments). The number of time points (rows) does not need to be the same in all the experiments, but the number of genes (columns) must be the same.
- `time_points` is a 1x3 cell array, where the k-th cell is a vector containing the time points of the k-th time series experiment.
- `decay_rates` is a vector containing gene decay rates.
- `genes_names` is a cell array containing the names of the genes.

3 Run dynGENIE3

dynGENIE3 is meant to be run on time series data (with or without steady-state data).

Run dynGENIE3 with its default parameters

The following input arguments are mandatory to run the function `dynGENIE3()`:

- Time series expression data
- The corresponding time points

```
[VIM, alphas] = dynGENIE3(TS_data,time_points);
```

The algorithm outputs:

- A matrix `VIM` containing the scores of the putative regulatory links. `VIM(i,j)` is the weight of the link directed from the i-th gene to j-th gene.
- A vector `alphas` in which the i-th element is the decay rate of the i-th gene. By default, the decay rate α_i of gene i is estimated from its time series data, by assuming an exponential decay $e^{-\alpha_i t}$ between the highest and lowest observed expression values of gene i.

Set the values of the decay rates

The user can specify the gene decay rates, using either:

- A single positive number. In that case, the decay rates of all the genes are assumed to have the same value.
- A vector of positive numbers, in which the i-th element specifies the value of the decay rate of the i-th gene.
- `'from_data'`. In that case, the decay rate α_i of gene i is estimated from its time series data, by assuming an exponential decay $e^{-\alpha_i t}$ between the highest and lowest observed expression values of gene i.

```
[VIM2, alphas2] = dynGENIE3(TS_data,time_points,decay_rates);
```

In this case, the returned vector `alphas2` is the same as `decay_rates`.

Run dynGENIE3 on time series data and steady-state data

`dynGENIE3()` can be used to learn networks jointly from time series and steady-state data. The steady-state data must be contained in a matrix where the element (n, i) is the expression of the i-th gene in the n-th steady-state condition.

```
[VIM3, alphas3] = dynGENIE3(TS_data,time_points,'from_data',SS_data);
```

Restrict the candidate regulators to a subset of genes

```
% Indices of the genes that are used as candidate regulators
input_idx = [1 2 6 10];
[VIM4, alphas4] = dynGENIE3(TS_data,time_points,'from_data',[],input_idx);
```

In $VIM4$, 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 = 4;

% Number of trees per ensemble
ntrees = 50;

% Run the method with these settings
[VIM5, alphas5] = ...
  dynGENIE3(TS_data,time_points,'from_data',[],input_idx, ...
    tree_method,K,ntrees);
```

Obtain more information

```
help dynGENIE3
```

4 Write the predictions

Get the predicted ranking of all the regulatory links

```
get_link_list(VIM)
```

The output will look like this:

```
## G4 G3 0.182500
## G6 G10 0.169262
## G5 G6 0.167712
## G6 G5 0.165200
## G9 G2 0.160855
## ...
```

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 `dynGENIE3()` **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 only the links that are directed from the candidate regulators

```
get_link_list(VIM,input_idx);
```

Show the names of the genes

```
get_link_list(VIM,input_idx,gene_names)
```

```
## aas_b2836_14 abgR_b1339_15 0.169262
## aas_b2836_14 aaeX_b3242_12 0.165200
## aaaD_b4634_3 aaeX_b3242_12 0.159581
## aaeA_b3241_14 abgB_b1337_15 0.157879
## ...
```

Show the first 5 links only

```
get_link_list(VIM,input_idx,gene_names,5)
```

Write the predicted links in a file

```
get_link_list(VIM,input_idx,gene_names,5,'ranking.txt')
```

Obtain more information

```
help get_link_list
```