Using sRACIPE

Vivek Kohar

10/15/2018

Introduction

This document shows illustrates the use of sRACIPE to simulate any network topology and analyze the generated data. For simplicity, we will use the toggle switch with mutual inhibition and self activation of both genes as a test case.

Load the network (topology)

As a first step, set the directory to the desired folder where you want to keep the input topology file and the results. In the working directory ("man" in this example), create a folder "inputs" and place the topology (the network) file in that folder. The typical format of the topology file is a 3 column file where the first column is name of the source gene, second column is name of the target gene and last column is the interaction type (1 - activation, 2-inhibition). The first line should contain the header (Source Target Interaction).

```
rm(list = ls())
library(sRACIPE)
working_directory <- getwd()
topology_file <- "inputs/test.tpo"
topology <- sRACIPE_load_topology(topology_file = topology_file)
```

```
## Topology file successfully loaded
```

```
topology
```

```
## $number_gene
## [1] 2
##
## $filename
## [1] "test"
##
## $topology
##   Source Target Type
## 1      A      B    2
## 2      B      A    2
## 3      A      A    1
## 4      B      B    1
##
## $topology_filepath
## [1] "inputs/test.tpo"
```

Initialize the simulation parameters

A configuration file containing the parameters for network simulation can be placed in the inputs folder. If the configuration file is not available, default arguments are used. Alternatively, you can specify the individual parameters when calling the functions.

```
config_file = "inputs/sRACIPE.cfg"
configuration <- sRACIPE_load_configuration(config_file = config_file)
```

```
## Configuration file successfully loaded.
```

```
configuration
```

```
##   ANNEALING possible_interactions NUM_MODELS PARAMETER_RANGE MPR_MIN
## 1         1                     3       1000             100       1
##   MPR_MAX DNR_MIN DNR_MAX FCH_MIN FCH_MAX HCO_MIN HCO_MAX STEP_SIZE
## 1     100     0.1       1       1     100       1       6      0.01
##   SIM_TIME MEDIAN_RANGE INITIAL_CONDITIONS OUTPUT_PRECISION
## 1       50          100                  1                12
##   THRESHOLD_MODELS RK_TOLERANCE NOISE_LEVELS MAX_NOISE
## 1            50000         0.25           10        30
##   NOISE_SCALING_FACTOR standard_deviation_factor SHOT_NOISE_SCALING
## 1                  0.5                    0.5658                  0
##   GENE_NOISE_SCALING FILE_WRITING_INTERVAL CONSTANT_NOISE PRINT_START
## 1                  0                     1              1          50
##   PRINT_INTERVAL PARAMETERS_FILE READ_IC
## 1             10               0       0
```

Deterministic simulations using fourth order Runge-Kutta method.

Now we will show deterministic simulations of this network using the fourth order Runge-Kutta integration method. Calling this function will generate four output files in the "results" folder in the same directory. The files contain (a) the names of the genes in the network, (b) the parameters for all the models, (c) the initial conditions used for each model (d) the final gene expressions for all the genes.

```
output_file <- sRACIPE_RK_deterministic(topology_file = topology_file, NUM_MODELS = 2000, INITIAL_CONDI
```

Plot the results

The simulated gene expression patterns can be plotted directly using the plot functions and plots will be in a pdf file in the results folder. There are three plots, (a) Heatmap of gene expressions on the first and second PC components, (b) Hierarchical clustering of gene expressions, (c) t-SNE heatmap of gene expressions

```
plot_data_deterministic(output_file, plot_filename = topology$filename)
```

Plot the network

The network can be plotted in an interactive html file in the results folder.

```
plot_network(topology)
```

Stochastic simulations

Stochastic simulations without annealing (for multiple noise levels)

Now we will perform stochastic simulations of this network using the Euler-Maruyama integration method.

Calling "sRACIPE_stochastic"" function will generate four output files in the "results" folder in the same directory. The files contain (a) the names of the genes in the network, (b) the parameters for all the models, (c) the initial conditions used for each model (d) the final gene expressions for all the genes. Notice that the output files now have NOISE_LEVELS*number_gene columns.

```
output_file_stochastic_NoAnnealing <- sRACIPE_stochastic(topology_file = topology_file, MAX_NOISE = 30,
```

Plot the stochastic results

This will plot the heatmap of gene expressions on the first two principal components. PCs are calculated using the determinstic data (D=0) and stochastic data (D > 0) are projected on the same PCs

```
plot_data_stochastic(output_file_stochastic_NoAnnealing, plot_filename=topology$filename, topology_df=t
```

Stochastic simulations with annealing.

We can perform stochastic simulations with annealing in similar way. Typically large number of noise levels (~30) is needed for better convergence

```
output_file_stochastic_Annealing <- sRACIPE_stochastic(topology_file = topology_file, MAX_NOISE = 30, N
```

```
## Topology file successfully loaded
```

```
## Topology file loaded.
```

```
## [1] "Configuration file successfully loaded."
##   ANNEALING possible_interactions NUM_MODELS PARAMETER_RANGE MPR_MIN
## 1         0                     3       2000             100       1
##   MPR_MAX DNR_MIN DNR_MAX FCH_MIN FCH_MAX HCO_MIN HCO_MAX STEP_SIZE
## 1     100     0.1       1       1     100       1       6      0.02
##   SIM_TIME MEDIAN_RANGE INITIAL_CONDITIONS OUTPUT_PRECISION
## 1      100          100                  1                12
##   THRESHOLD_MODELS RK_TOLERANCE NOISE_LEVELS MAX_NOISE
## 1            50000         0.25           30        30
##   NOISE_SCALING_FACTOR standard_deviation_factor SHOT_NOISE_SCALING
## 1                  0.6                    0.5658                  0
##   GENE_NOISE_SCALING FILE_WRITING_INTERVAL CONSTANT_NOISE PRINT_START
## 1                  0                     1              1          50
##   PRINT_INTERVAL PARAMETERS_FILE READ_IC
## 1             10               0       0
## 2
```

```
## Generating gene thresholds
```

```
## generating thresholds for uniform distribution1...
```

```
## Running the stochastic simulations
```

```
##   ANNEALING possible_interactions NUM_MODELS PARAMETER_RANGE MPR_MIN
## 1         0                     3       2000             100       1
##   MPR_MAX DNR_MIN DNR_MAX FCH_MIN FCH_MAX HCO_MIN HCO_MAX STEP_SIZE
## 1     100     0.1       1       1     100       1       6      0.02
##   SIM_TIME MEDIAN_RANGE INITIAL_CONDITIONS OUTPUT_PRECISION
## 1      100          100                  1                12
##   THRESHOLD_MODELS RK_TOLERANCE NOISE_LEVELS MAX_NOISE
## 1            50000         0.25           30        30
```

```
##   NOISE_SCALING_FACTOR standard_deviation_factor SHOT_NOISE_SCALING
## 1                  0.6                    0.5658                  0
##   GENE_NOISE_SCALING FILE_WRITING_INTERVAL CONSTANT_NOISE PRINT_START
## 1                  0                     1              1          50
##   PRINT_INTERVAL PARAMETERS_FILE READ_IC
## 1             10               0       0
## Running time evolution simulations for 2 genes...
## Using same noise level for each gene
## Simulations completed successfully. Data files are in results folder.
```

Plot the stochastic results with annealing

This will plot the heatmap of gene expressions on the first two principal components. PCs are calculated using the determinstic data (D=0) and stochastic data (D > 0) are projected on the same PCs

```
plot_data_stochastic(output_file_stochastic_Annealing, plot_filename=topology$filename, topology_df=topo
```