**Construction, visualization, and analysis of biological network models in Dynetica**

Derek Eidum[1,a], Kanishk Asthana[1,a], Samir Unni[1], Michael Deng[1], Lingchong You[1,2,3*]

[1.] Department of Biomedical Engineering, Duke University

[2.] Center for Systems Biology, Duke University

[3.] Center for Genomic and Computational Biology, Duke University

[a.] These authors contributed equally

[*] Correspondence should be addressed to L.Y. (you@duke.edu)

## 1 Modular Systems in Dynetica

### 1.1 Introduction

Modular Systems are an extension of Dynetica 2.0 that allow the user to create and maintain models that are modular in organization. This ability to add and remove modules gives the user incredible power when creating models, as it allows previous models to be reused and extended. Moreover, many gene circuits are modular in organization and construction. Large networks tend to be composed of smaller subnetworks, which communicate with one another through a handful of interconnects.

Using Dynetica 2.0, the user can quickly and easily create models which can be used to study the properties of a new gene circuit constructed using well-known smaller circuits as modules. This modular systems feature, when used in combination with Dynetica 2.0's import/export features, allow the initial MATLAB or SBML code for a model to be generated much more quickly and robustly, as the user won't have to rewrite the entire code when the model is modified or extended.

### 1.2 Design Paradigm

Like the previous version of Dynetica, a model can have Reactions, Substances, Parameters and Expressions (RSPEs) at the highest level of organization. In this version in addition to RSPEs, the highest level of organization (henceforth referred to as the Super system) also contains Modules. Like the Super system, each module can also have its own RSPEs. These RSPEs remain globally

accessible, i.e. individual RSPEs are visible to all the other RSPEs in the system, independent of whether they are present in the super system or in a module; for example a reaction can have substances inside a module as reactants and substances outside the module as products: the system behaves as if the modules don't exist. Moreover, the user can build models of arbitrary complexity by importing or exporting modules to and from Dynetica 2.0 (at the moment, the ability to import modules into a system is restricted to the system type "Modular System"). In addition, an existing model can be extended by merging its contents with an empty Modular System.

To aid in the creation and maintenance of modules, Dynetica 2.0 has introduced many new and intuitive features.

### 1.3 Creating and Editing Modules

Modules can be created from the main Dynetica 2.0 window by clicking the new Entity dropdown menu at the bottom of the screen and clicking on "GeneralModule". This should open a dialog box asking the user to name the new Module.

Once created the module interface looks similar to the interface of the main system. New entities can be created within the module by using the entity creation dropdown menu at the bottom of the window. Moreover, the structure of the module can be visualized using the module tree on the left. In addition to providing visual representation of the module structure, the module tree can be used to access each element in the module.

RSPEs can also be easily added to the Super system from a module and added to a module from the super system or from another module, using intuitive gestures. The drag and drop gesture common in windows file systems can be used to drag and drop RSPEs into a module using the main system window. Moreover, by clicking and dragging an RSPE within a module to the bottom of the module window transfers the RSPE to the main system. Drag and drop gesture can also be used to

transfer entities from popped-out modules to other modules that have not been popped out (explained below).

### 1.4 Module Visualization

A module can be visualized easily by double clicking the module in the main system. Moreover, by right clicking a module and then clicking on Pop Out, the contents of a module are immediately displayed in the main system window. These contents can be hidden again by clicking on "Pop In all Modules" at the bottom of the main window. Moreover, modules can be popped in and out from their respective windows.

The module window also provides a few intuitive features to help the user understand and visualize the model structure. These features include the ability to hide and display connections to the super system and the ability to visualize the immediate connection partners for RSPE's in the module. In addition, Dynetica 2.0 displays the kinetics or value of an RSPE when you mouse over it.

### 1.5 Importing, Exporting, and Merging Modules

To create a module that the user wants to store as an independent system, a new system type of "Reactive System" can be created from the File menu. This system can then be manipulated in a manner similar to a "Modular System" except no new modules can be created in this system.

To import the above system or any older systems as a module, the saved .dyn file for that system can simply be either merged with or imported as a module when "Merge" or "Import as Module" buttons are clicked at the top of the main window. Therefore by merging or importing a system a module, models of arbitrary complexity can be easily made.

### 1.6 Manipulating the System Tree

The main system tree displays the complete model and its structure. It contains the nodes Parameters, Reactions, Substances, Expressions and Modules. Expanding each parent node allows the user to view and access the elements inside that node.

By clicking the Modules node in the system tree and expanding a particular module node shows the RSPEs within that module. Moreover right clicking each node shows several options, including the ability to Pop In and Pop Out modules. The structure of this tree changes dynamically as RSPE's are moved from and to modules.

## 2 Parameter Searching in Dynetica

### 2.1 Introduction

The newly-implemented parameter search tools in Dynetica 2.0 allow the user to specify a desired behavior for the system and find a set of parameters for which the system closely matches this behavior. To do this, the user specifies metrics that he/she would like to optimize, and the target values for each metric. When specifying targets for multiple metrics, the user can assign different weights to each target, and targets with higher weights will exhibit a greater influence on the outcome of the parameter search. In addition, the parameter search can also be conducted to optimize a dose response of a simulation over a range of parameter of substance values.

The parameter search is done using a genetic algorithm (Holland, 1992), where the population consists of individual parameter vectors, and each individual's fitness is dependent on how well the output simulation matches with the user-defined behavior. To begin, the user specifies the target objectives, corresponding weights, and a maximum number of iterations for the algorithm. The population is then initialized, and the model simulation is run for each individual parameter vector, with a fitness score is assigned to the simulation output based on the user-defined objectives

and weights. Half the population is removed, with removal probability dependent on fitness score, and the population is restored through random mating between the remaining parameter vectors. This process is continued until the user-stopping criteria or the maximum iterations is reached.

**2.2 Specifying Target Objectives**

The parameter search can be initiated from the main Dynetica 2.0 window by clicking "Simulation" on the top toolbar, and clicking "Genetic Parameter Search." A dose response sensitivity analysis can be conducted by instead clicking on "Dose Response Parameter Search." A dialog box should then open, allowing the user to specify the target objectives for the parameter search.

For the Genetic Parameter Search dialog box, the user can select metrics, their target values (the combination of a metric and its target value hereafter referred to as an objective), and their weights, as well as which parameters to vary during the simulation. The user-selected metrics can be maximized, minimized, or targeted to a specific value. In addition, the must specify the population size, maximum number of iterations, and scoring threshold for the genetic algorithm. The scoring threshold represents the minimum necessary score for the termination of the algorithm. When the algorithm is run, each parameter vector is assigned a score from 0-100, and a scoring threshold of 0.05 means the algorithm will stop once a parameter vector of score 95 or higher is found.

One especially powerful objective is maximizing the correlation between a substance concentration and the output of an expression variable. By specifying an arbitrary function as the expression variable, the time course of the substance can be matched to nearly any shape or function.

In addition, the user can specify the method to initialize the population of parameter vectors. The default is to generate the initial population through random perturbations. Each parameter value is generated from a log-normal distribution with normal mean equal to the natural logarithm of the initial parameter value and normal standard deviation equal to the natural logarithm of 10. If,

however, the user chooses the "Randomize Initial Population" option, each parameter value is generated from a uniform distribution bounded by the minimum and maximum values of that parameter. In addition, the user-defined parameter vector is always included as an individual in the initial population.

**2.3 Genetic Algorithm Workflow**

Each iteration of the genetic algorithm consists of three main steps: simulation, culling, and rebuilding. In the simulation step, a model simulation is conducted for each individual parameter vector, and the output of the simulation is compared to the user-specified target objectives and scored. In the culling step, half the population of parameter vectors are removed, with probability of removal dependent on score. In the rebuilding step, the population is restored to its initial size by mating, mutation, and invaders (Holland, 1992).

For each objective, a score of 0-100 is assigned based on how well the simulation output matches the target value. The final score for each parameter vector is calculated as the weighted average of each of these objective scores, with weights supplied by the user in the initialization of the algorithm.

The culling of the population is dependent on score: lower scoring parameter vectors are more likely to be removed from the population. The probability of survival is proportional to the squared value of the score. In addition, the highest scoring parameter vector of the population is guaranteed to survive to the next iteration of the genetic algorithm so that the maximum score achieved does not decrease.

After culling, the population is rebuilt with mating and random invaders. For every iteration of the algorithm, there is a 2% chance a random invader will enter the population. This random invader is generated analogously to the randomization of the initial population—the values of each parameter are generated from a uniform distribution bounded by the minimum and maximum value

of the parameter. The remainder of the restored population is generated through mating of the parameter vectors surviving from the previous generation, with mating probability proportional to the square of the score. The parameter values of the child vector are calculated as a weighted average of the values of the mother and father vectors, with the respective weights generated from a uniform distribution on [0,1). In addition, the parameter values of the child vector are subject to mutation. For each parameter, there is a 10% chance the parameter value will be mutated, with the mutated value generated from a log-normal distribution with normal mean equal to the child's initial parameter value (calculated from weighted average of parent vector values), and normal standard deviation equal to one twenty-fifth of natural logarithm of 10.

**2.4 Dose Response Parameter Search**

The methods behind the dose response parameter search capabilities in Dynetica 2.0 are analogous to those outlined above for the basic genetic parameter search. There are, however, some differences in the initialization and operation of the algorithm.

In addition to specifying the target objectives and parameters to be varied, the user must also specify which variable to vary for the dose response. The user can select the range of variation, the number of intermediate points, and whether the points are linearly or logarithmically spaced. The target objectives for the dose response parameter search can be to make the dose response curve maximally linear, maximally or minimally sensitive, or to maximize the distance of the peak or first extrema. Like the basic genetic parameter search, multiple objectives can be chosen and their respective weights specified.

The genetic algorithm functions in the same three steps of simulation, culling, and rebuilding for the dose response parameter search as well. However, for the simulation step, multiple simulations are conducted for each individual parameter vector, varying the user-chosen variable for each simulation. Each resulting dose response curve from the output of the multiple simulations is

then scored 0-100 for each objective, the final score being a weighted average of the scores for each objective. The population culling and rebuilding steps proceed in the same manner as the basic genetic parameter search.

**References**

HOLLAND, J. H. 1992. Adaptation in natural and artificial systems an introductory analysis with applications to biology, control, and artificial intelligence. *Complex adaptive systems*. 1st MIT Press ed. Cambridge, Mass.: MIT Press,.