Dynetica User's Guide (version 1.2beta)

June 2005

Peter Blais and Lingchong You

Department of Biomedical Engineering

Duke University

Send comments, suggestions, and bug reports to you (at) duke.edu

- A Simplified Phage T7 Model (Back to top)

Appendix

Introduction

Dynetica is a modeling interface designed for the easy construction, visualization, and analysis of kinetic models. It is specifically intended for use in biological systems and adapts many of its tools and commands to reflect this. In particular, stochastic model analysis through the Gillespie algorithm is accounted for, while systems exhibiting Michaelis-Menten kinetics can be reproduced by a single function that eliminates the need to write out every single reaction occurring. The net result is a program that enables an accurate assessment of system behavior with a minimal amount of mathematical drudgery.

Operation of the interface is relatively intuitive; following through the provided examples should be enough of a crash course to give one a suitable proficiency to reproduce the kinetic reactions and visual models desired.

Dynetica is written in Java. You need Java 1.3 or above to run the program. We recommend always using the latest version of Java. You should be able to get Java from Sun (http://www.sun.com) for free.

Getting started

Installation (Back to top)

Dynetica v.1.2 beta can be downloaded from http://www.duke.edu/~you/Dynetica_page.htm. Follow the links therein and get a zipped file called DyneticaInstall.zip. Unzip this file to a directory, say DyneticaDirectory. In DyneticaDirectory there will be two items, one is a jar file: Dynetica.jar, the other is a subdirectory called "examples", which holds some example models built in Dynetica.

To run Dynetica, type either of the following commands in a command window (in Unix or Windows):

- java –classpath Dynetica.jar Dynetica // NOTE: if you're in the directory "DyneticaDirectory"
- java –classpath <DyneticaDirectory>/Dynetica.jar Dynetica //NOTE: if you're anywhere else

After typing this, the main (blank) window should pop onto your computer screen (Figure 1).

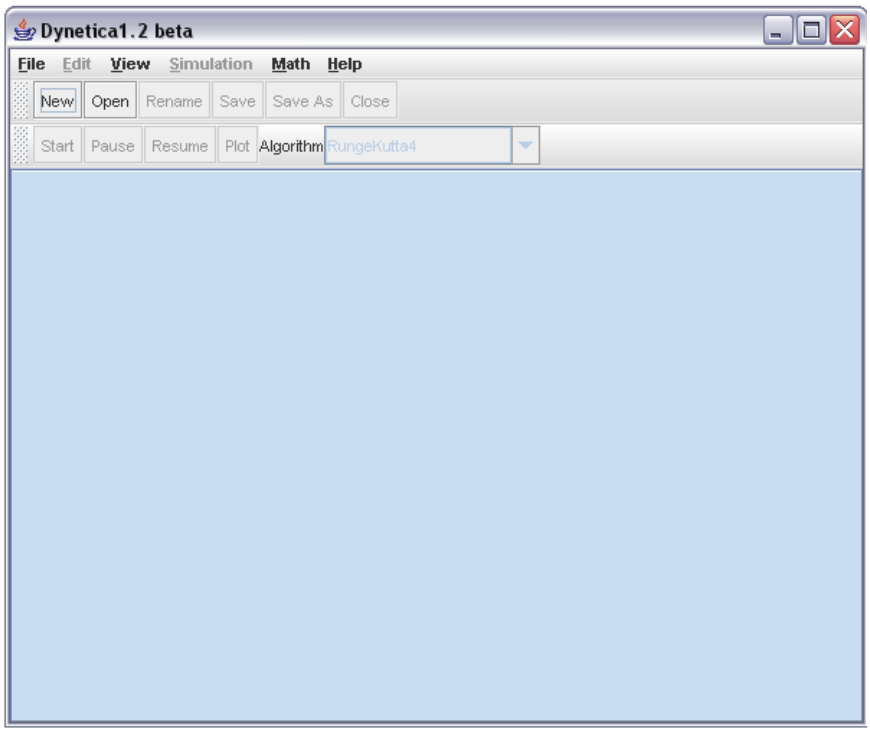

Figure 1. The main window without any models open

Note: Some of my colleagues have used Dynetica in a MacOS (OS X, I believe), althougth I haven't tried myself. Since I can't give you advice here, you will need to consult the manual for the Java virtual machine on how to run a java program in Mac.

⦚ Opening an existing model (Back to top)

To open an existing model, click on the **Open** button. A dialog window will appear. Using the dialog window to locate the directory, <DyneticaDirectory>/examples, something like Figure 2 should be displayed.

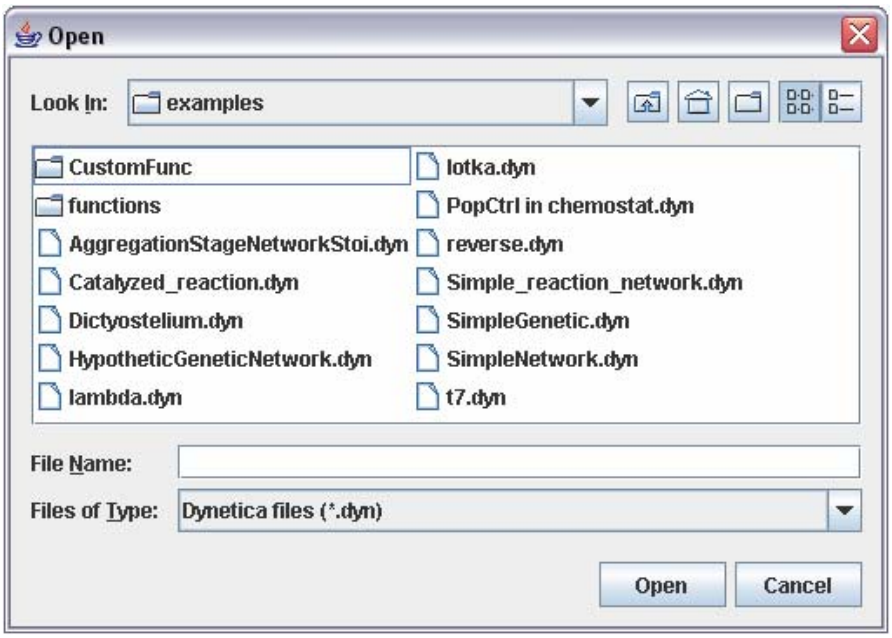

Figure 2. The dialogue window for opening an input file, within the "examples" directory in the Dynetica package.

Dynetica reads and creates .dyn files. If a file is opened in a directory other than the default one, Dynetica will remember this directory and set it as the new default. See the Example Models section of the manual for further instructions on opening and operating the files in the examples directory.

⦒ Creating a new model (Back to top)

Click on the button **New**, or select the menu item **New > Reactive System** in the **File** menu. The following window should pop out:

Figure 3. Dialogue window for creating a new system, named "Test" in this case.

Click OK and a blank system (bereft of substances or reactions) should come up (Figure 4). In this window, the left panel shows the tree-structure view of the network, and the right panel gives a graphic representation. There will be a list of parameters on the left panel; at the start only "Time" should exist. This parameter should never be modified or deleted: doing so may corrupt the input file.

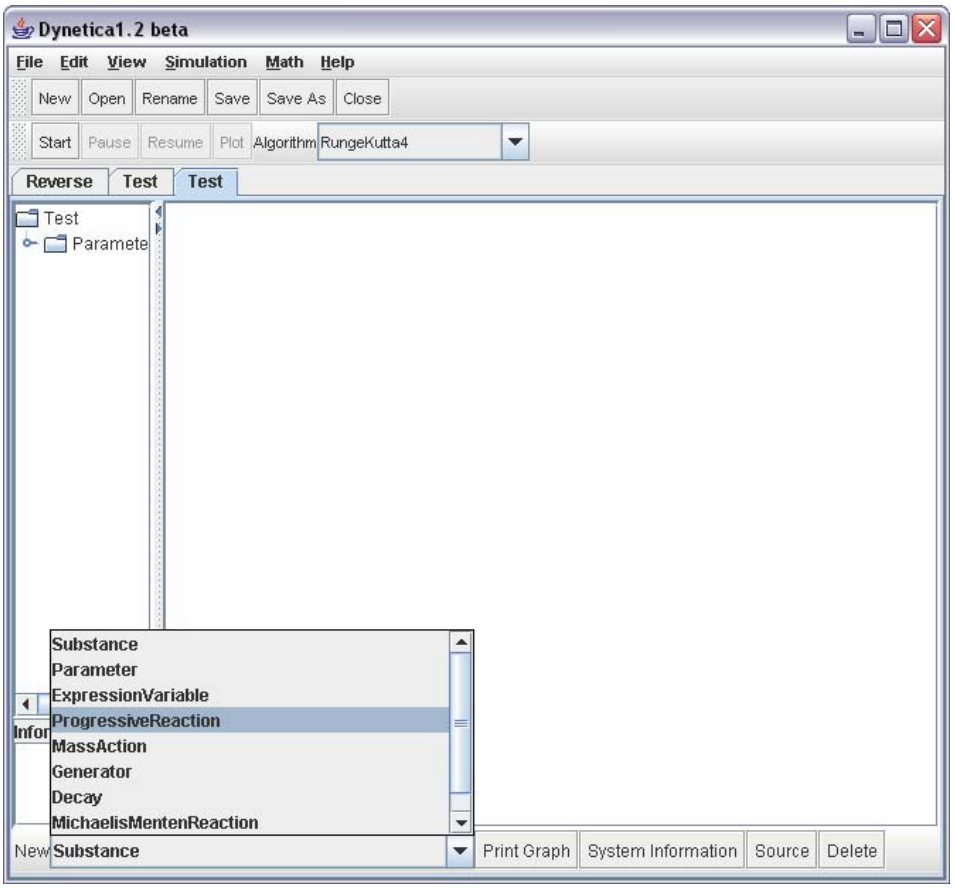

Figure 4. The blank test model. No substances or reactions are present in the graphical interface. Also, the combo-button at the bottom of the window has been pressed and the **ProgressiveReaction** option is highlighted.

In Dynetica, a substance stands for a reaction component: either some sort of reactant, product, or catalyst. To create a new substance, click on the combo-button at the bottom besides the "New:" label and select **Substance** (see Figure 4). After a name has been assigned to this substance, a blue node corresponding to it will appear on the main console. Note that this name cannot contain spaces or mathematical operators (+, *, /, ^, etc). Also, the editor window for the substance will also pop up, allowing you to specify the initial and boundary numbers of the substance. Biologically, this corresponds to molecule count but can also be representative of concentration, as Dynetica will work with decimals if the numbers provided are too small. The editor window can be closed by clicking on the upper-right corner of the box and reopened by double-clicking on the corresponding node (in the 1.2 version, this turns the node green).

To model a simple reaction that converts one substance to another, first create a second substance which will be the product of the first substance. Then, click on the same combo-button used to create the substance, but this time select the **ProgressiveReaction** option. Again, assign a name without spaces or mathematical operators, and once the editor window appears, type within the **Stoichiometry** field the

line "*First Substance Name -> Second Substance Name*", as if writing down the chemical reaction itself (see Figure 5). Next, in the **Kinetics** field, type in "*k* * [*First Substance Name*]", where *k* represents the rate constant parameter of the reaction. Once this is typed, *k* will appear on the lower half of the editor window. Its value may be entered on the **Parameters** list at the left-hand side of the main window, or it can be provided in the lower right box of the reaction editor window (Figure 5, again). Brackets or parenthesis may be omitted as long as no parameters and substances share the same names. For further demonstration, see the Examples section of the tutorial.

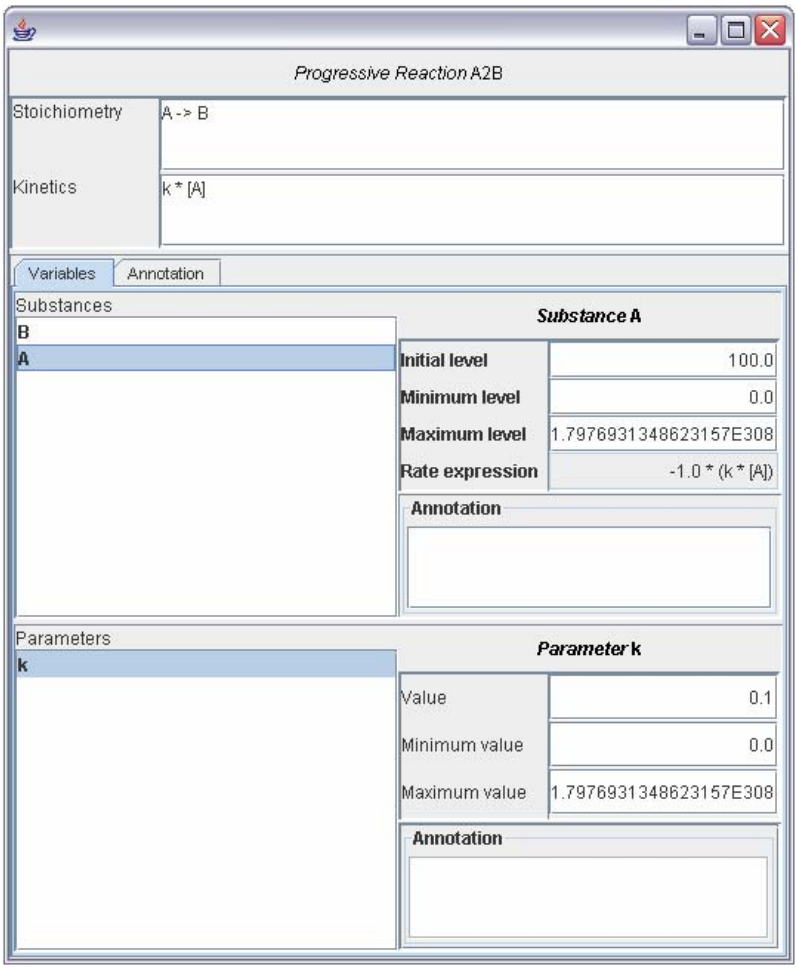

Figure 5. Editor window for the first-order reaction described above. Here, A and B represent the two interacting substances, with A forming B under first-order kinetics governed by rate constant k. As shown, A has an initial concentration of 100, and k has a value of 0.1. B has an initial concentration of 0.

The reaction in the graphical interface will appear as a boxed arrow. A green line indicates the production of the connected substance by the connected reaction, a red line represents the consumption of the connected substance by the connected reaction, and a gray dashed line indicates that the connected substance affects the kinetics of the connected reaction. Each component on the graphical interface can be moved around the space of the main window, and an ordered version of the model described in Figure 5

should look something like that of Figure 6. Parameters can be defined ahead of time using the same combo-button at the bottom, but since they are automatically generated when the reaction is established there is little point. In fact, substances can also be omitted before creating the reaction, and the program will automatically add them in. However, because the graphic layout of nodes and reactions on some platforms can be messed up by doing this, it is generally recommended that substances be created before the reactions they participate in. Such seeming trifles become significant pains when modeling larger reaction mechanisms.

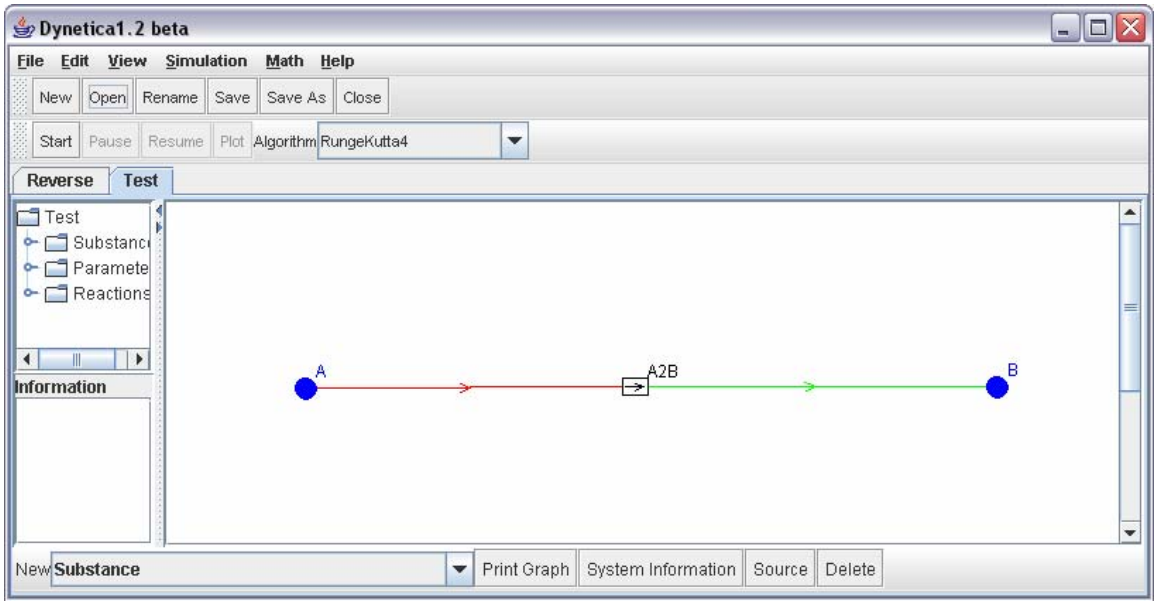

Figure 6. The first example model in Dynetica, created from the reaction editor window shown in Figure 5. The reaction consists of a first-order conversion of A into B.

Save the model by clicking on **Save** or using the **Save** or **Save As** buttons under the **File** menu. Repeat the creation of substances, reactions, and parameters as needed, until the desired reaction mechanism is complete. To print the completed graphical interface, use the **Print Graph** button at the bottom of the window.

Reactions in Dynetica (Back to top)

Dynetica classifies reactions as belonging to one of two types: progressive reactions or equilibrated reactions. A progressive reaction travels in only one direction: hence it is progressive. It is modeled using the **ProgressiveReaction** selection within the aforementioned bottom combo-button. As shown in Figure 5, two formulae define a progressive reaction in Dynetica, stoichiometry and kinetics (the establishment of these reaction conditions within the reaction editor window was detailed in the previous section).

In addition to the generalized form of the progressive reaction, several specialized reactions have been developed off the basic one to facilitate model construction. Each is

listed in the bottom combo-button. The **MassAction** selection will form a reaction in which the kinetics are based on the mass action law. The reaction specified by the **Generator** selection produces a product without the consumption of reactants at a rate directed by mass action kinetics. It is a simplified progressive reaction in that the stoichiometry need not be defined, only the product and the kinetics of the product generation. The **Decay** selection works in much the same way, except the substance specified acts as a reactant which is consumed in a degradation reaction. Again, only the reactant and kinetics are needed.

The **MichaelisMentenReaction** selection constructs a reaction that abides by Michaelis-Menten kinetics. Rather than writing out each reaction of enzyme binding and dissociation, this option models a single reaction to reproduce the system behavior of the entire substrate-enzyme interaction. The parameters V_{max} and K_m along with the names of the substrate, product, and enzyme must be defined. For systems where an enzyme binds to more than one substrate at a time, the **HillActivation** selection allows for the insertion of a Hill coefficient within the reaction kinetics of the Michaelis-Menten system. Thus, the **MichaelisMentenReaction** option is simply a **HillActivation** reaction where the Hill coefficient is equal to one.

All these reactions are simplifications of the original **ProgressiveReaction**, such that **ProgressiveReaction** may be used to produce the same effect as the other options, but not the other way around. For the **Generator** and **Decay** reactions where one side of the reaction is empty, simply leave that side of the chemical equation blank in the **Stoichiometry** field of the **ProgressiveReaction** editor window (see Figure 7).

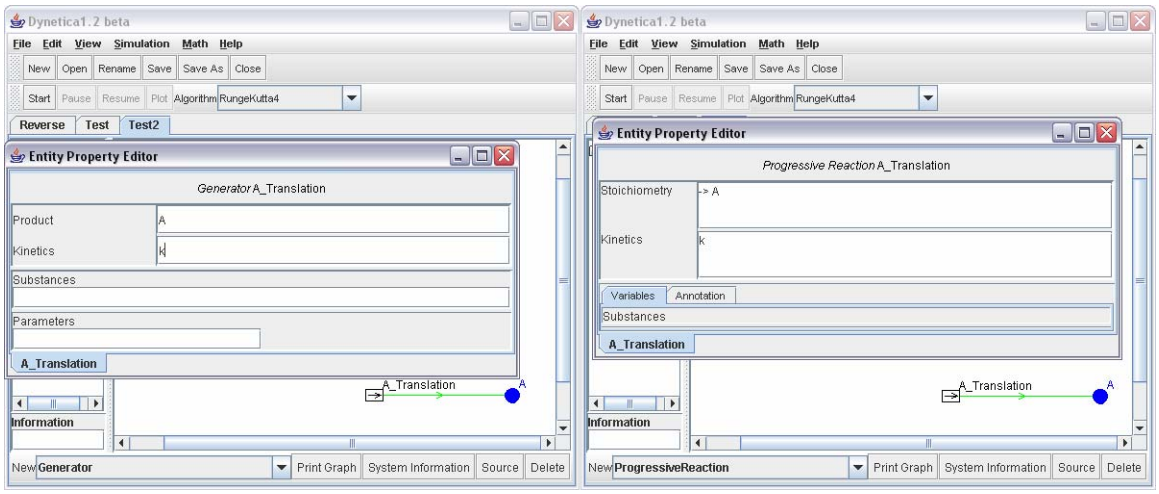

Figure 7: Modeling the formation of substance A, using either the **Generator** (left) or **ProgressiveReaction** (right) selections. Note how the stoichiometry is labeled for the progressive reaction case.

Currently, only one special type of equilibrated reaction has been implemented: the **EquilibratedMassAction**. As the name suggests, it creates an equilibrated reaction where the equilibrium constraint is specified following the mass action law. In this case, the stoichiometry of the reaction and the equilibrium constant must be provided. On the

graphical interface, an equilibrated reaction will appear as a boxed equals sign with red and green arrows indicating the direction of the reaction from reactants to products as dictated by the specified stoichiometry. Note that, because equilibrated reactions can occur in both directions, the graphical interface may not always correctly indicate the net direction of the equilibrated reaction. Another important point is that equilibrated reactions cannot be integrated into the Gillespie algorithm. When stochastic analysis is required, an equilibrated reaction should be replaced by a pair of reversible progressive reactions.

⦚ Kinetic expressions in Dynetica (Back to top)

Through the use of the expression parser in Dynetica, many circuit properties can be modeled which are not reproducible within the stoichiometric and kinetic framework of Dynetica's progressive reactions. They can account for time-varying input signals or provide an indicator of subsets of substance concentrations. In essence, expressions handle any mathematics that occur apart from the reaction chemistry of the system. To create a new expression, use the combo-button at the bottom of the page and select **ExpressionVariable**. The *Expressions* section of the appendix provides a list of the operations and functions understood by Dynetica within this structure. To see examples of the functionality of expressions, check the Lotka-Volterra model in the *Examples* section.

Running Simulations

⦚ Time course simulations (Back to top)

To run the model, click the **Start** button on the Simulation Bar at the top of the screen and then the **Plot** button next to it. These options along with most of the others mentioned below are available on the Simulation Menu of the Menu Bar as well—see the Menus Section. A simulation window containing the time-dependent behavior of substances in the system should appear (Figure 8). By default, **Time** is used as the x-axis, but you may select a state variable (substance level) as the x-axis. Only one variable may be used to define the x-axis, but multiple variables may be plotted as a function of it. Use the **Shift** and **Ctrl** keys while clicking on substances in the **Y values** list to get more than one substance to appear in the plot. The resolution of the plot always begins at a low value; increasing it by shifting the bottom bar right will make the plot more accurate by inserting more data points over the same interval, but will also adversely affect computational speed. Another feature of the simulation window is a list of the maxima and minima of substances for the x- and y-axes at the bottom left of the window. If multiple substances are selected, then the x- and y-axes ranges will consist of the union between the substances; thus, the maximum y-value of two substances will be the maximum y-value out of either of the two. Finally, the option exists for the plots to be displayed with logarithmic scales in either of the axes (or both if desired).

Deterministic simulation

By default, Dynetica uses a variable time step 4th order Runge-Kutta algorithm (labeled "RungeKuttaFehlberg"), which is implemented based on the description in *Numerical Recipies* (http://www.nr.com/). The classic Runge-Kutta 4th order algorithm (with a fixed time step rather than variable) is also available as a tool to model deterministic systems and should reproduce the results of its updated version. To access it, click on the top combo-button on the Simulation Bar beside the "Algorithm" label and select the **RungeKutta4** option.

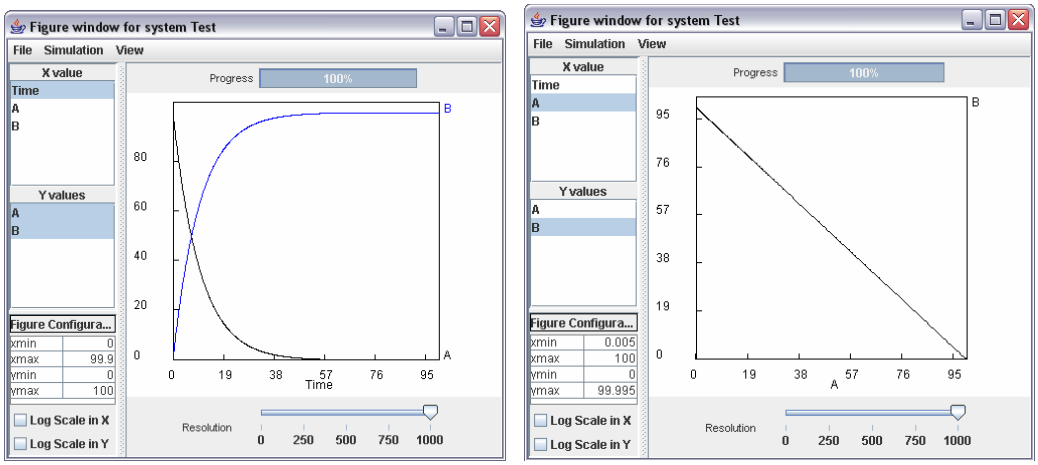

Figure 8. The time courses for the toy model in Figure 6. The left plot uses the default **Time** to define the x-axis, while the right plot uses the count of substrate A. This plot of B as a function of A is called a phase plane plot. The phase plane will look more interesting in a system that generates oscillations. For example, try the model described in lotka.dyn.

To switch algorithms or configure the simulation, either click on the desired algorithm within the top combo-button or use the menu bar to select the proper algorithm within the **Simulations > Algorithm** section. Each algorithm has a unique set of parameters that influence such things as the time-step of each data point and total duration of the simulation. Figure 9 shows the parameters corresponding to the default algorithm. The **Sampling step** sets the interval between each sample point, while the **Iterations** number gives the number of sample points produced. Naturally, the total time interval of the resultant plot becomes the product of these two values, so to increase the resolution of a simulation, decrease the sampling step while proportionally increasing the iterations to maintain a constant product. Also, the error tolerance for the variability of the steps can be manipulated: a smaller tolerance corresponds to higher accuracy (notice that this option does not exist for the classic algorithm).

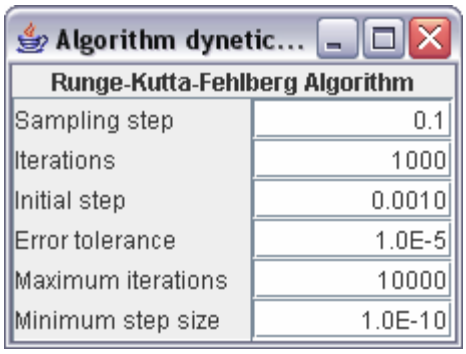

Figure 9. The configuration windowof the **RungeKuttaFehlberg** algorithm.

Stochastic simulation

In addition to the deterministic method of solving kinetic systems, Dynetica can also model system dynamics by considering each reaction as an individual step that has a certain probability of occurring during a sampled time interval. This type of analysis is relevant for systems involving low numbers of molecules of reactants and products. It is mathematically manifested via the Gillespie algorithm and termed stochastic because the presence of an equal probability any reaction firing off at any moment in time introduces a randomness to the simulation not possible within deterministic simulations. Because reactions in this simulation are fired off one at a time, computational time can become an issue in more complex models.

Three particular methods of performing the Gillespie algorithm are available in Dynetica for stochastic modeling. In the same combo button where the deterministic modeling options were located, click on one of the following: **DirectMethod**, **OptimizedDirectMethod**, or **FirstReactionMethod**. The Direct Method, the most common method for implementing the Gillespie algorithm, operates based on a random choosing of reactions with reaction probabilities that reset at every iteration of the time step. The Optimized Direct Method works the same way as the Direct Method but does not reset parts of the system that do not change. It will produce the same results as those of the Direct Method, but in cases of large networks it will be useful in decreasing the computation time. The First Reaction Method operates on an entirely different premise, creating variable time intervals between firings of each of the separate reactions (rather than variable probability intensities for the Direct Method) which are then overlapped onto the overall time sequence of the simulation.

Like their deterministic counterparts, the stochastic algorithms have modifiable parameters which can dictate the resolution and scale of the model. The **Sampling Time Step** and **Number of Iterations** have the same meaning as the sampling step and iterations on the deterministic model, respectively. The **Number of rounds** field defines how many times the stochastic simulation is to be run, with multiple system responses being saved to a data file as defined by the **Output** field. Simply click on the button to assign a data file to save the output to.

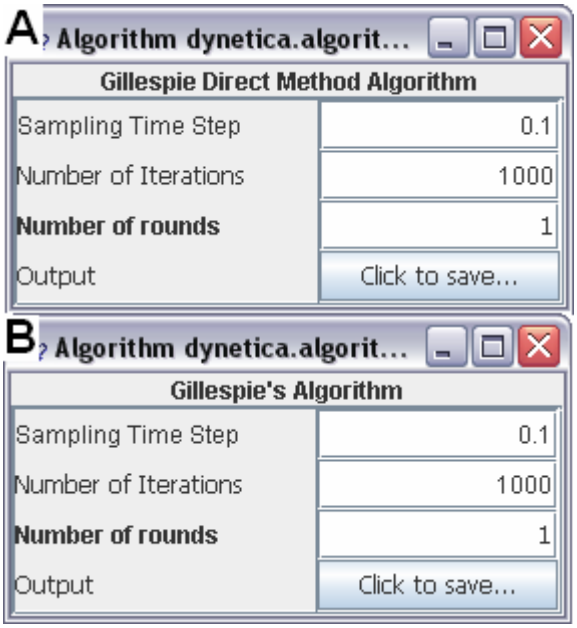

Figure 10. The configuration windows for (A) the **DirectMethod** and (B) the **FirstReactionMethod**.

See Fig. 15B for a stochastic model of the "reverse.dyn" system proposed in the *Examples* section. Note how they differ from behavior of the deterministic system only by the additional presence of noise. In other instances this may not necessarily be the case.

Basic Sensitivity Analysis (Back to top)

In addition to simulating the temporal evolution of a reaction network, Dynetica provides the basic functionality to explore how the network dynamics respond to perturbations, in terms of variations in parameter values or initial substance concentrations. This feature is desirable for simulating dosage curves and for identifying key system parameters that are important in determining the overall behavior of the system.

To access the sensitivity analysis window, click on the **Sensitivity Analysis** button in the **Simulation** menu. The window, seen on Figure 11, contains a variety of fields that define the parameter to be perturbed, the degree of perturbation, and the substances with which to monitor the response. Type the desired parameter under **Variable**, the range of values to be tested for this parameter under **Minimum Value** and **Maximum Value**, and the number of points within the range to test under **Number of Points**. Clicking the **Log Scale** button will make the distribution of data points between the maximum and minimum occur at logarithmic intervals. The sensitivity analysis program will run the simulation for each of the specified parameter values, and record the substance counts listed in the bottom right window at the time listed in the **Time** field. To denote which substances to record, highlight them on the left column at the bottom of the window and click "Add" to send them to the righthand column. Run the program by clicking the **Run** button, and plot the substance count as a function of parameter value by pressing **Plot**.

The plot for sensitivity analysis offers much the same features as that for the general model simulation: minima and maxima for each substance as well as logarithmic scaling of the axes. See Fig. 16 for a demonstration of sensitivity analysis of the "reverse.dyn" system in the Examples section.

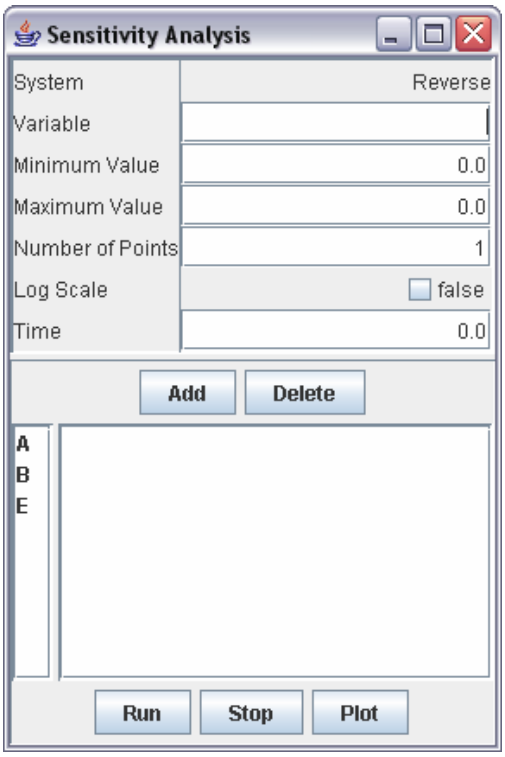

Figure 11. Sensitivity analysis configuration window, with default values.

Example models

(Note: the following models can be found in the examples directory.)

An Enzymatic Cycle Model (Back to top)

The first example of this tutorial has already been discussed. The kinetics of a simple A to B progressive reaction were examined, and Dynetica's tools were utilized to reproduce its operation on both the graphical interface and the simulation plot. The next example will be one order of complexity higher; it will consist of two progressive reactions opposing one another so that one reaction's products are the reactants of the other, and vice versa. This reaction mechanism, catalyzed by an enzyme in each direction, is commonly known in biology as an enzymatic cycle—however, in this case, the reaction kinetics are simplified by eliminating the enzyme in the reverse direction and modeling each reaction without Michaelis-Menten kinetics.

To access the model, click **Open**, either on the File Bar or under **File** in the menu, and from the default directory where Dynetica was installed, find the "reverse.dyn"

model under the "examples" directory. The graph of the model should look something like that on Figure 12 (note that the layout may be different but the substances and reactions are the same). Also, the reaction stoichiometry and chemical kinetics of this system are written here in Table 1 for convenience. These entries can be copied verbatim into the respective stoichiometry and kinetic fields for both reactions to derive a working model.

Table 1. The reactions in the simple reaction network shown in Figure 12.

Reaction name	Stoichiometry	Kinetics
R1	A → B	k_1 [A] [E]
R2	B → A	k_2 [B]

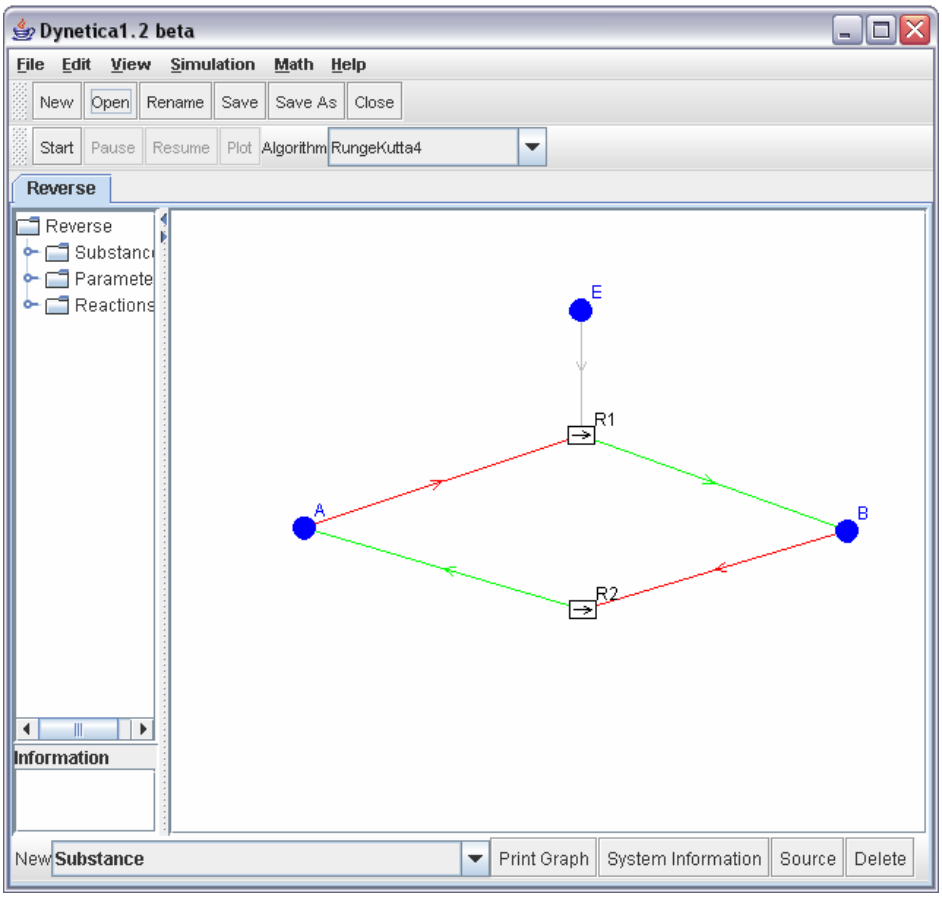

Figure 12. The main window with the system Reverse opened.

Click the nodes on the left hand side and inspect various entities in the model. Also, try to inspect the substances and reactions using the graphical interface (graph for short). For example, if double-clicking the node at A will bring up an entity editor (see below), where the attributes of this substance (such as initial level, minimum and maximum

levels) can be manipulated. The program will also automatically generate a rate expression viewable within editor window. Finally, substances can be annotated in the Annotation field.

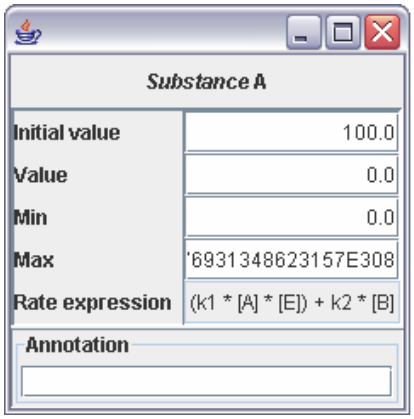

Figure 13. The entity editor window for substance A.

Double-clicking the node in the system tree will also bring up the entity editor window. Figure 14 shows the entity editor that pops up upon double-clicking the reaction node R1.

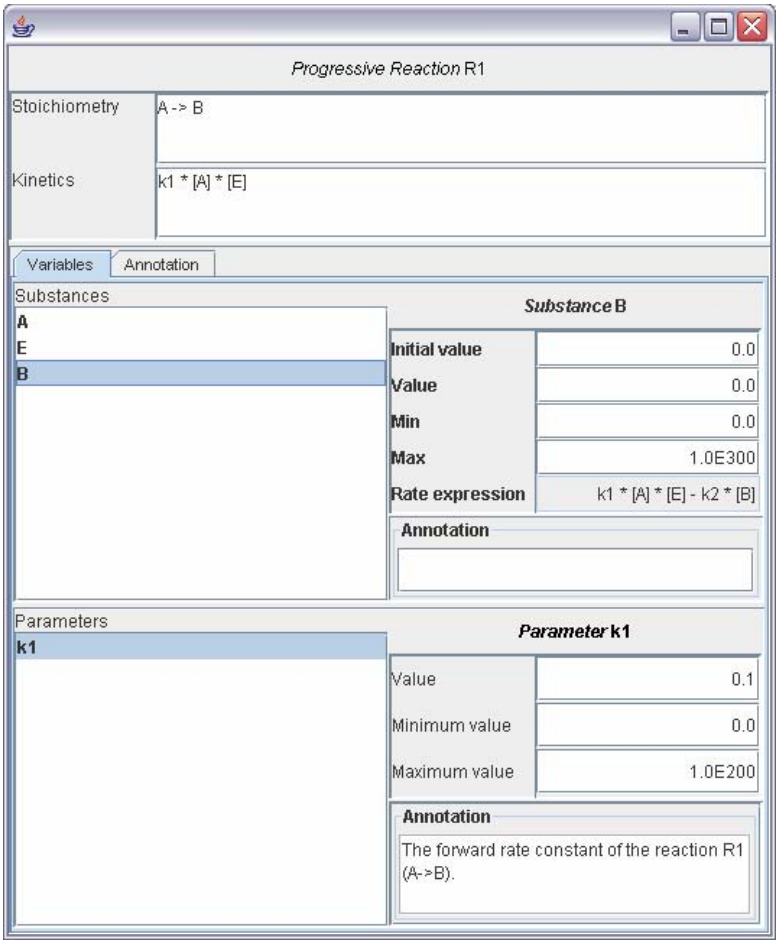

Figure 14. The editor window for reaction R1. A ProgressiveReaction is characterized by two basic attributes: its stoichiometry and its kinetics. Both can be typed in by following conventions in writing a reaction and a mathematical expression. Note that the substances and parameters associated with this reaction can also be edited within this window.

Run the simulation by pressing **Start** on the Simulation Bar, and then plot it with **Plot**. If all settings remain at default, the plot seen in Figure 15A should be seen (some settings in the number of iterations may need to be made to get the time-scale to match; to see how to do this, refer to the *Deterministic simulation* section). Shift-click all the substances under the **Y values** box to see all the time responses for all the substances simultaneously. Close the window, click on the algorithm combo-button (also on the Simulation Bar) and select the **DirectMethod** option. Leave the algorithm configuration as is, and press **Start** and **Plot** once again. This time, a plot like that on Figure 15B should appear (also with a possible variation in the time scale).

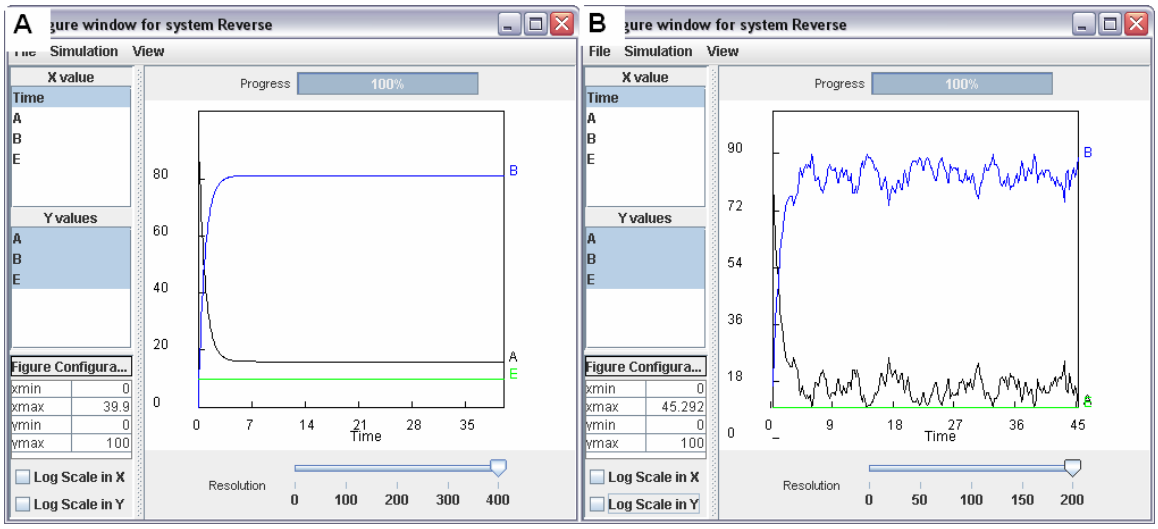

Figure 15. (A) Deterministic and (B) stochastic simulation results from the model "reverse.dyn".

Next, a sensitivity analysis of the system will be performed. Click on the **Sensitivity Analysis** button underneath the **Simulations** menu, and enter in the values shown on Figure 16A for each corresponding field. Doing so will set the variable to be varied as parameter k1. It will be varied from 0 to 1, with an interval of 0.1 (10 points will be calculated), and the system will be simulated until the time point at 50.0, at which point the molecule counts of all three state variables (A, B, and E) will be shown. By performing this test for a range of k1 values, a plot of each substance's dependence upon the k1 parameter will be produced (Figure 16B), lending a physical interpretation to the robustness of the system with respect to this parameter.

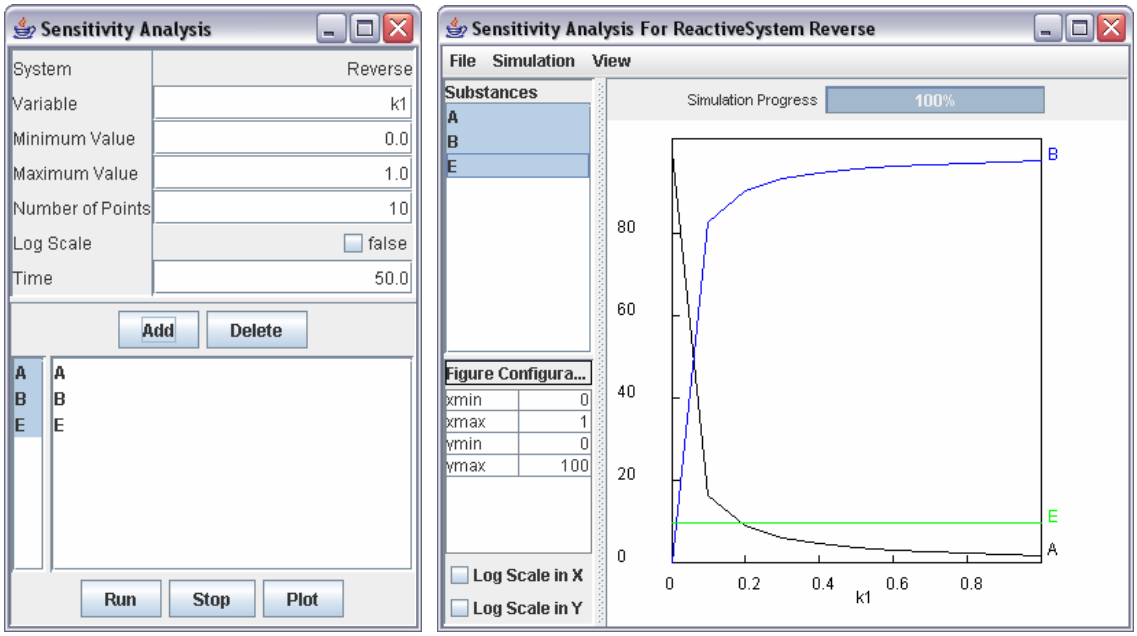

Figure 16. (A) The sensitivity analysis window with information specified. (B) The sensitivity analysis result for substances A, B, and E.

A Lotka-Volterra Model

Predator-prey kinetics can be modeled by this simple Lotka-Volterra model. Like the previous model, it can be found in the "examples" directory. The basic operation of each reaction is dictated in Figure 17. This is the first example to make use of an expression, C. Figure 18 demonstrates how to set C to equal the sum of the two species. When the parameters of growth, conversion, and decay are correctly calibrated, oscillations for both species in the system will be observed.

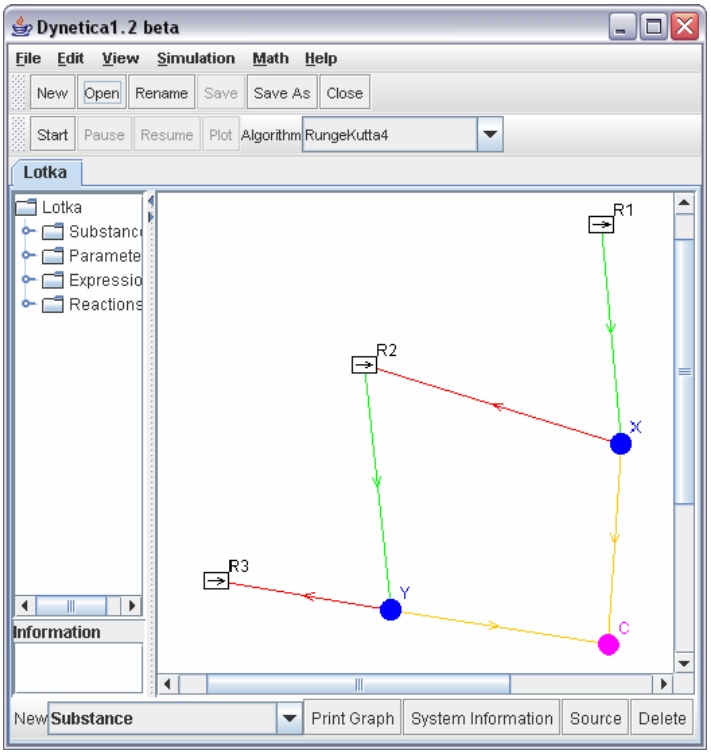

Figure 17: The Dynetica graphical representation of a Lotka-Volterra model, with X as prey and Y as predator. R_1 represents the generation of prey, R_2 the killing of X by Y (and coincident rescue of Y by X), and R_3 the death of Y. Expression C exists for graphical purposes only; it is the sum of both species at any given time.

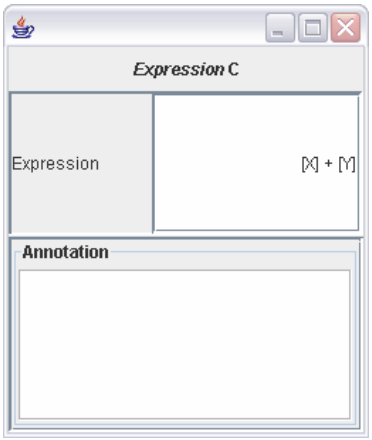

Figure 18: Editor window for expression C.

Running the simulation will produce a plot like that of Figure 19. Notable results that can be gleaned from this model include the observations that the predator population behavior lags behind that of the prey and that the total system population reaches a maximum after the prey population begins to decrease but before the predator population hits its maximum.

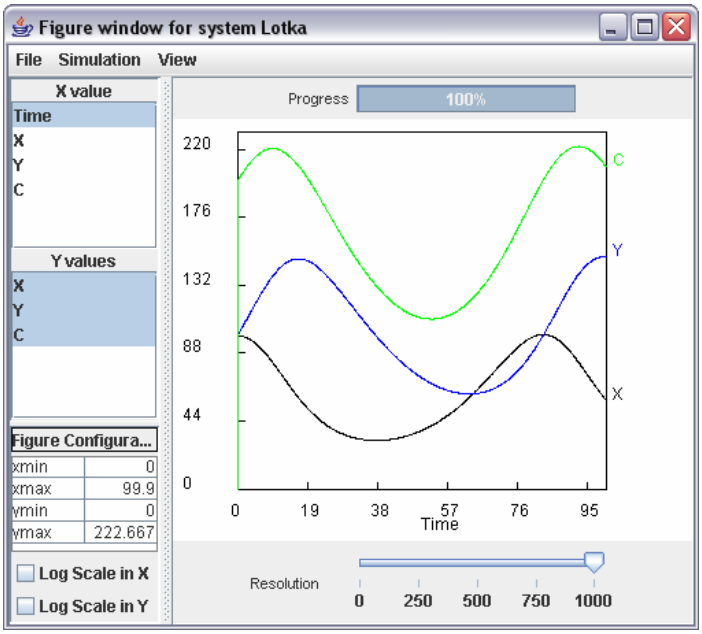

Figure 19: Simulation of system output for the predator (X), prey (Y), and total population (C).

If any of the parameters for the three reactions are unbalanced, the system will either fail to oscillate, decaying to the trivial steady-state for both species, or the oscillations will become distorted. To conceptualize this system dependence upon the parameters, the sensitivity analysis can be used for all three rate constants to reveal the ranges at which the system will oscillate (Figure 20). From the sensitivity analyses a better understanding is gained as to why such values for the parameters were originally chosen.

Finally, a limitation of the sensitivity analysis tool becomes evident when the plots of Figure 20 hint at exponential growth when in fact oscillations do exist, but are extremely distorted. Just because a substance concentration at one time point is growing exponentially does not mean the entire system is. To truly understand the nature of a model, its time-dependent behavior must be simulated using each of the new parameter values.

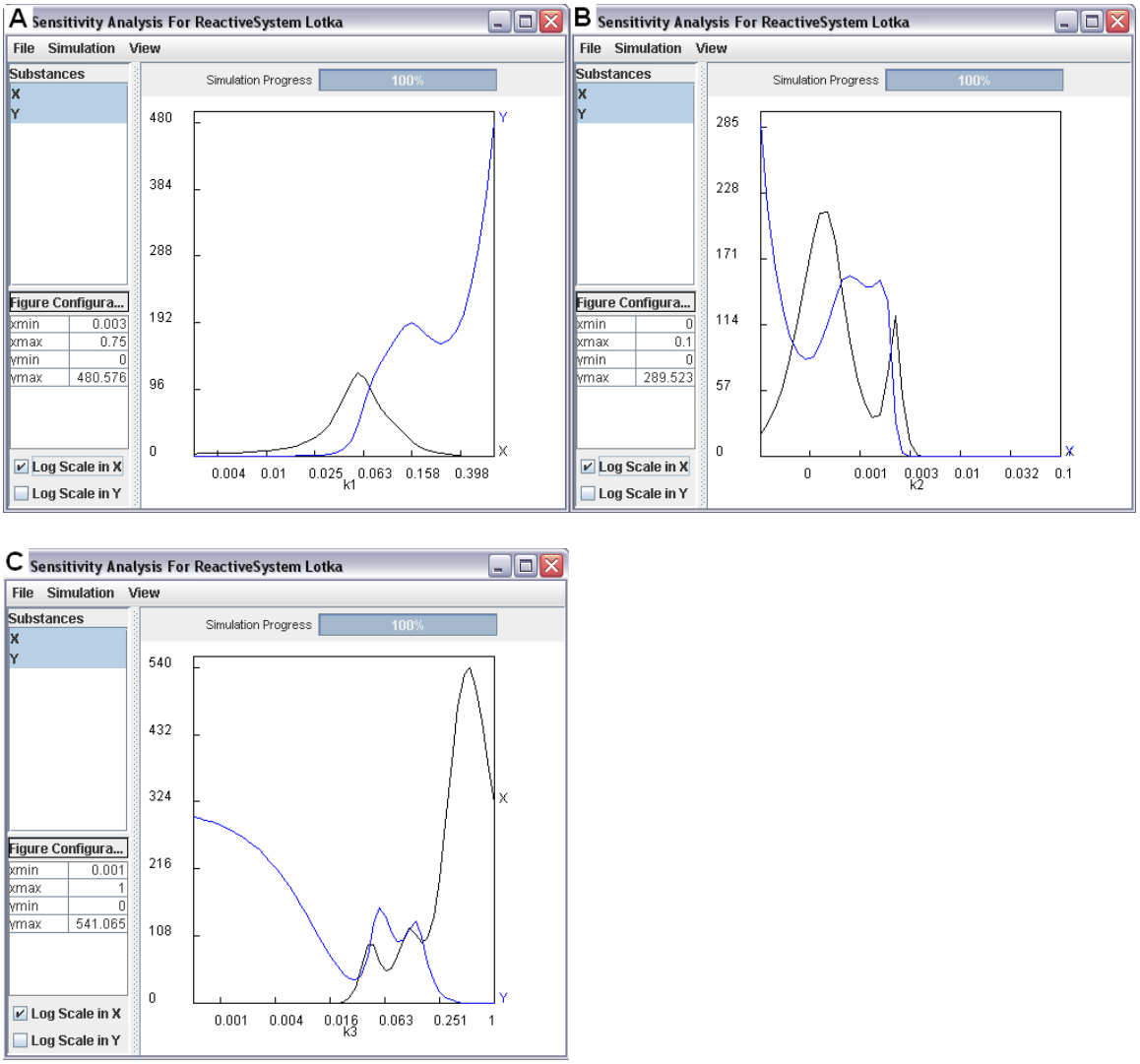

Figure 20: Sensitivity analysis for the three rate constants of the three reactions in the Lotka-Volterra system. (A) Rate constant for the generation reaction of prey X. When it is too low, the system will be trivial, whereas the predator will exhibit exponential growth (at that time point) when it is too high. The value settled upon in the Lotka-Volterra model was 0.1. (B) Rate constant for the conversion reaction of prey X into predator Y. When it is too low, the prey count will not decrease enough over time, so it will remain high. When it is too high, the trivial state ensues. The value settled upon in the model was 0.001. (C) Rate constant for the decay reaction of predator Y. When it is too low, predator concentration will not decrease fast enough and will remain high over time. When it is too high, prey concentration will blow up. The value settled upon was 0.06.

An Dictyostelium Aggregation Stage Network (Back to top)

This model is based on Laub and Loomis, 1998, Mol Biol Cell 9: 3521-32. However, the parameters used here have different values from the published parameters, some of which were in error (W. Loomis, personal communication). Reactions of this network are shown in Table 2, with Dynetica's version of this in Figure 23, while the interface and simulation of the model on Dynetica are given in Figures 21 and 22. With appropriate parameters, this model will generate stable oscillations.

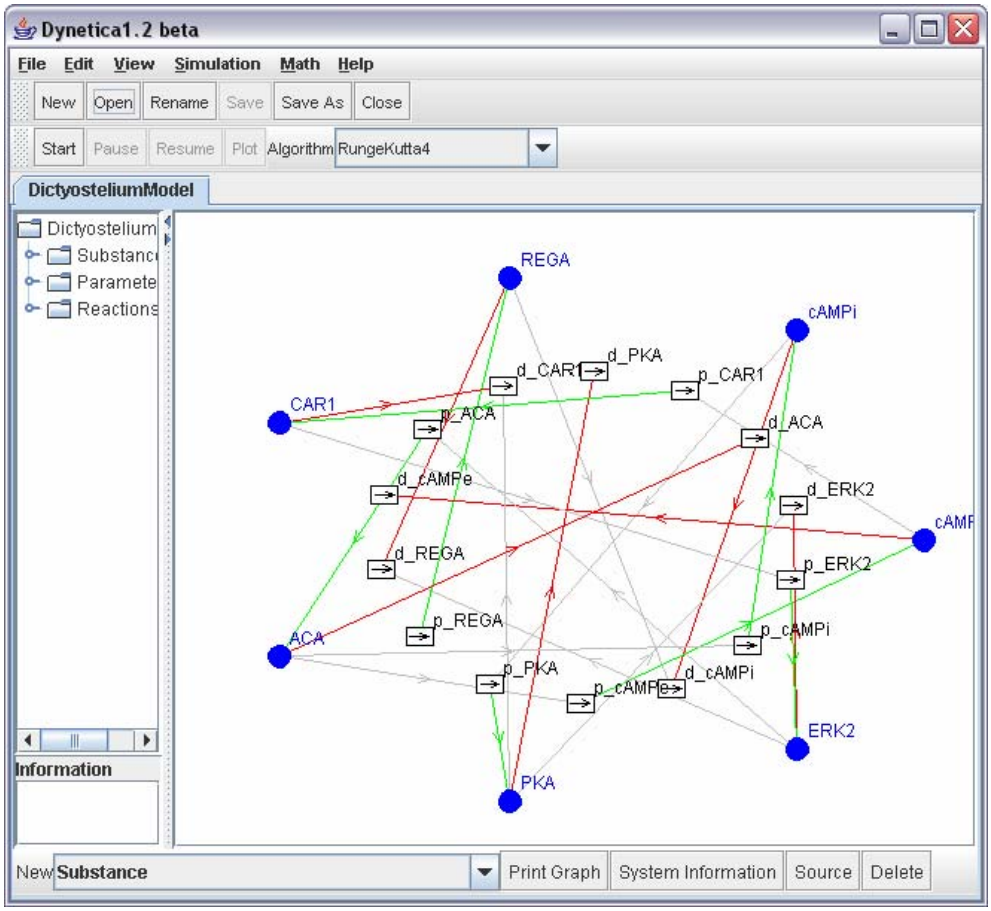

Figure 21. The graphical interface representation of the aggregation stage network model.

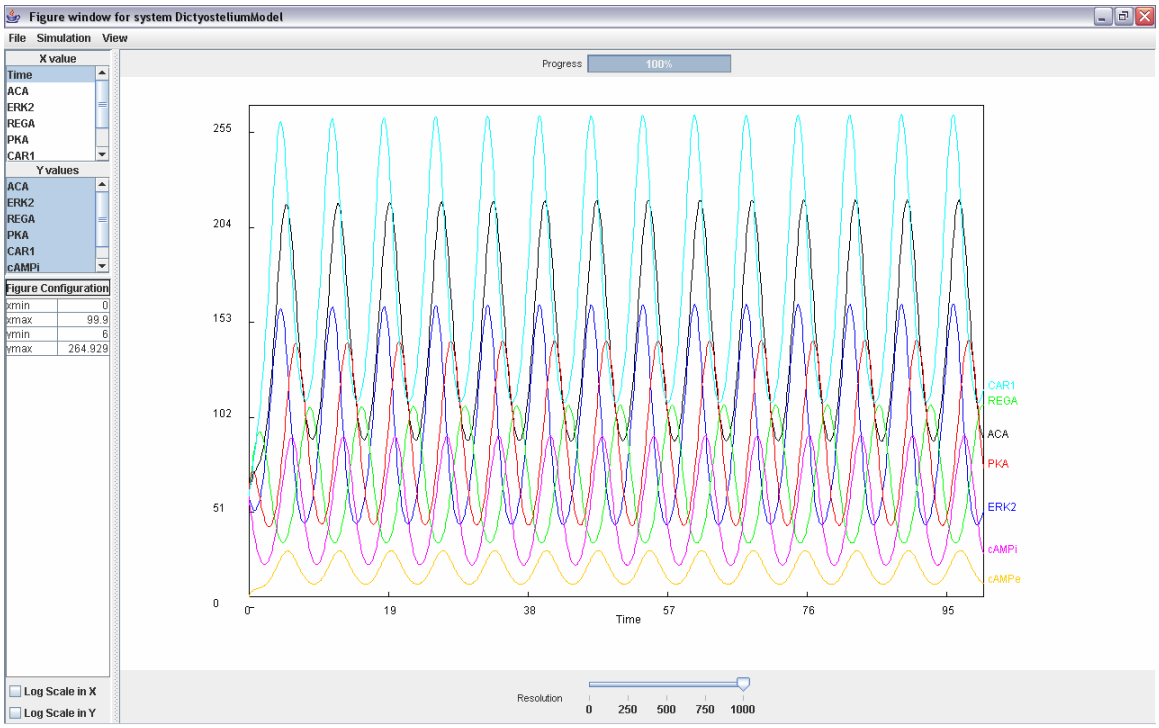

Figure 22. A typical simulation result for the aggregation stage network model demonstrating oscillations in the enzyme levels.

Table 2. The production reactions in the aggregation stage network [a]

Reaction	Stoichiometry	Kinetics	Notes
p_ACA	→ ACA	k_1 [ERK2]	activation of ACA by ERK2
d_ACA	ACA →	k_2 [ACA]	degradation of ACA
p_PKA	→ PKA	k_3 [cAMPi]	activation of PKA by cAMPi
d_PKA	PKA ⟆	k_4 [PKA]	degradation of PKA
p_ERK2	→ERK2	k_5 [CAR1]	activation of ERK2 by CAR1
d_ERK2	ERK2 →	k_6 [ERK2] [REGA]	degradation of ERK2 (catalyzed by REGA)
p_REGA	→REGA	k_7	constant production of REGA
d_REGA	REGA→	k_8 [REGA] [ERK2]	degradation of REGA (catalyzed by ERK2)
p_cAMPi	→ cAMPi	k_9 [ACA]	activation of cAMPi by ACA
d_cAMPi	cAMPi →	k_{10}[REGA][cAMPi]	degradation of cAMPi (catalyzed by REGA)
p_cAMPe	→ cAMPe	k_{11} [ACA]	activation of cAMPe by ACA
d_cAMPe	cAMPe→	k_{12} [cAMPe]	degradation of cAMPe
p_CAR1	→ CAR1	k_{13} [cAMPe]	activation of CAR1 by cAMPe
d_CAR1	CAR1→	k_{14} [CAR1][PKA]	degradation catalyzed by PKA

[a] Although recent studies have suggested a slightly revised reaction network (http://www.biology.ucsd.edu/labs/loomis/network/laubloomis.html), the published model suffices to illustrate the usage of Dynetica.

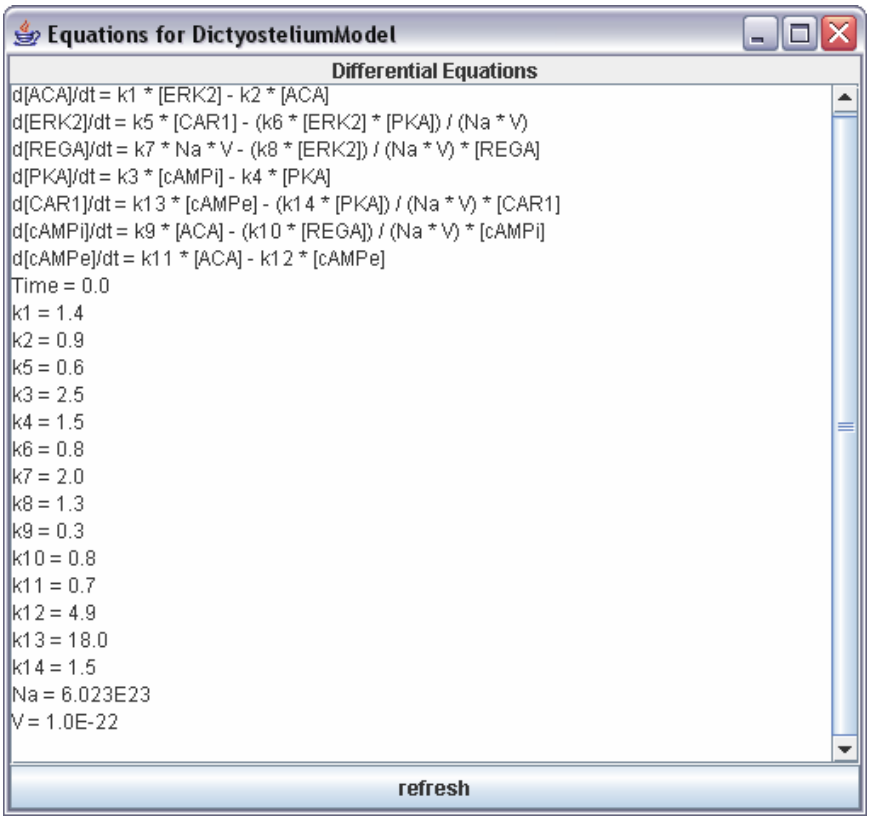

Figure 23. Dynetica's window displaying all rate equations and parameter assignments, accessible through the **Math >> Show Equations** option on the Menu Bar.

A Simplified Phage T7 Model (Back to top)

The model presented here is a simplified version of a model of phage T7 intracellular growth cycle. The major difference between the current model and the previous ones is that a simplified genome is used here (Figure 24). This simplified genome contains 20 essential T7 genes. The regulatory effect of promoters and transcription terminators is accounted for by specifying the relative transcription activity of each gene. As a result, RNA polymerases are allocated to different genes based on their relative transcription activities, whereas in the complete model RNA polymerases are allocated based on the relative strengths of promoters. The resulting T7 reaction network contains 91 reactions and 55 substances, excluding genes (Figure 25). In this network, the reactions describing expression of genes and degradation of gene products are automatically generated by Dynetica.

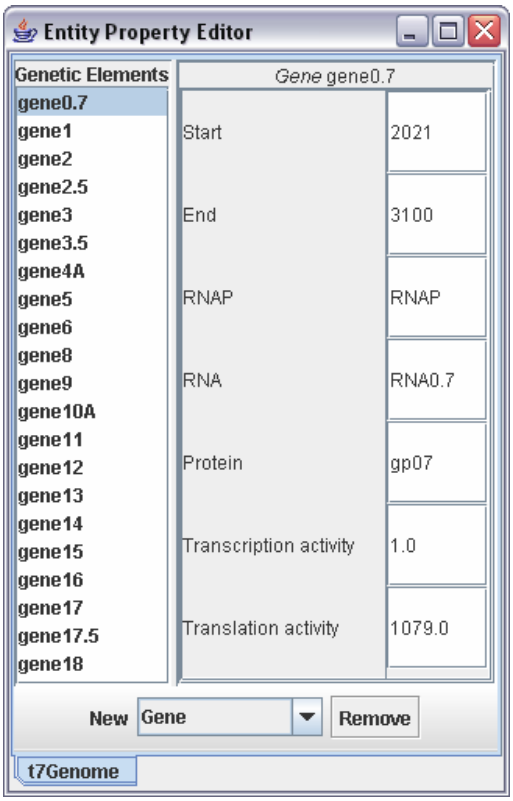

Figure 24. Simplified genome list defined within Dynetica for the T7 bacteriophage. The right column gives the properties of each respective gene.

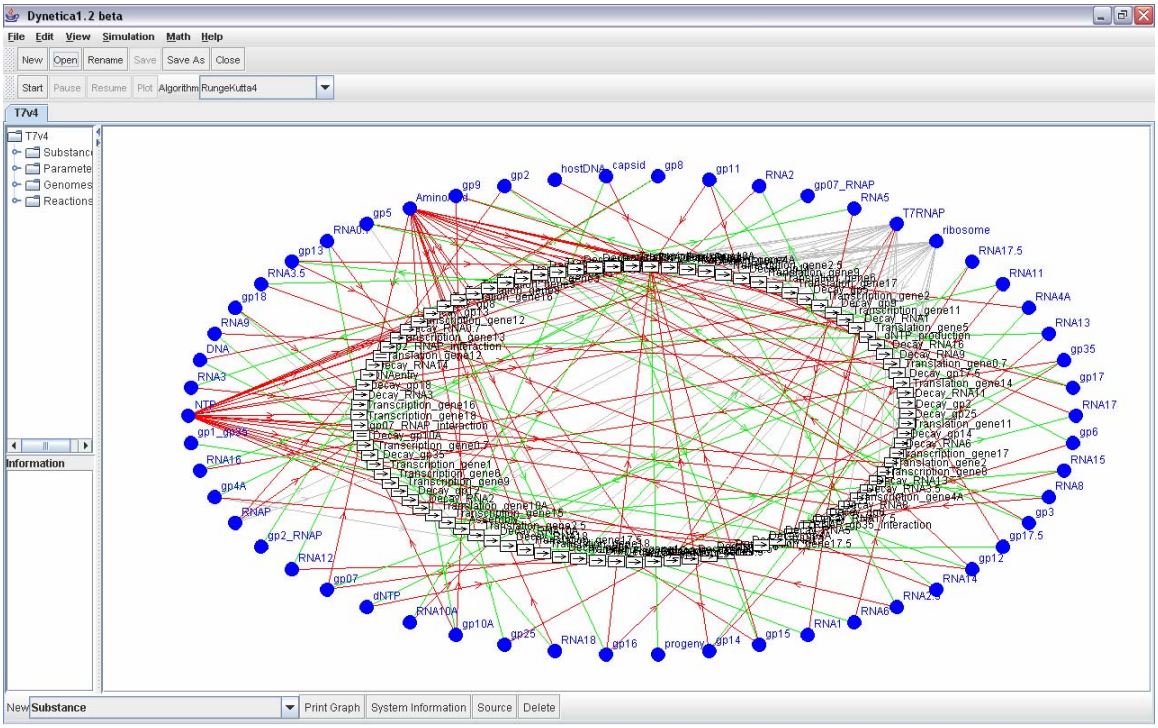

Figure 25. The graphic representation of the reaction network for the simplified phage T7 model. Most of the reactions are automatically generated based on the genetic information shown in Figure 19.

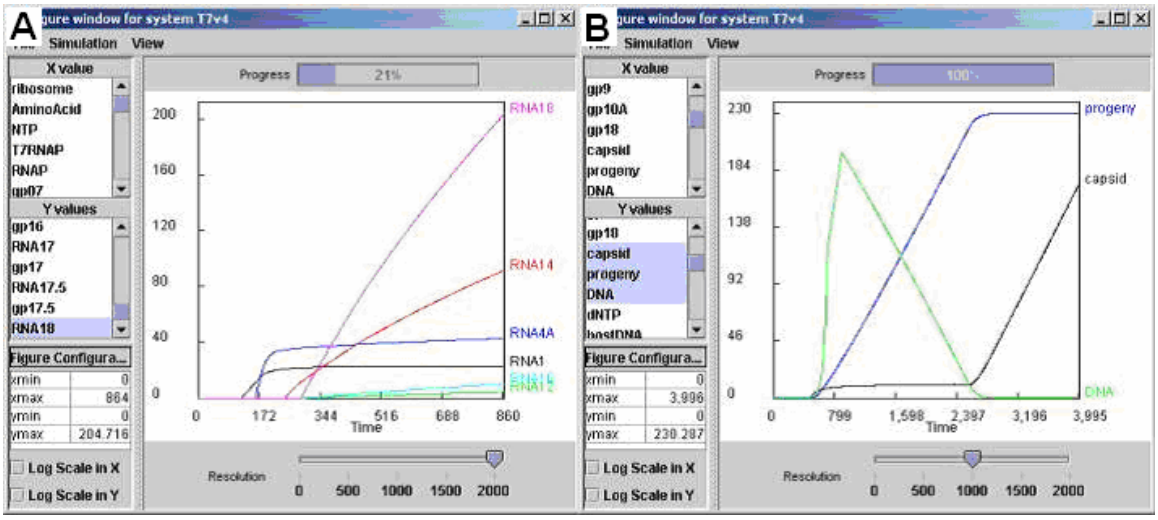

Figure 26. Typical simulation results from the simple T7 model. Time courses of (A) selected mRNAs (half way into simulation), and (B) three major phage components: the newly synthesized DNA, procapsids, and final phage progeny.

Appendix

The appendix is a listing of all functions, options, and tools offered by Dynetica that have not otherwise been discussed in previous sections. It covers the Menu Bar of both the Dynetica window and the plot window, the options available by right-clicking on the graphical interface window and the parameters window, the variety of operators and functions supported in the Expressions window, and offers some troubleshooting tips as well.

Dynetica Menu Bar

File:

New >> Reactive System- creates a new model of the typical format; has the same function as the **New** button on the File Bar.

New >> Genetic System- creates a new model accounting for all the interacting genes within the genome of a system organism. This section will be explained in a future update.

Open- opens the browser window to allow the selection of a previously saved model to be loaded into Dynetica; has the same function as the **Open** button on the File Bar.

Save, Save As- saves the current model as its previously saved file name, or in the case of the latter, allows for the creation of a new name for the model; have the same functions as the **Save** and **Save As** buttons on the .

Close- closes the model currently displayed on the graphical interface window; if it is the only window open then Dynetica will simply become blank.

Exit- exits Dynetica.

Edit:

Rename Current System- allows for the system to be given a new name; (note that this name is Dynetica's designation and not Windows': to edit the file name of the system simply use the **Save As** option).

Annotation- opens the System Information window which normally is displayed at the beginning when any model is opened. The annotation can be edited at any time

View:

Show File Bar- toggle switch which allows for the display or removal of the File Bar (the first bar below the menu bar).

Show Simulation Bar- toggle switch which allows for the display or removal of the Simulation Bar (the second bar below the menu bar).

Look & Feel- allows for the change of the window style into three different design templates (the **Mac** option works only with the Mac OS).

Simulation:

Algorithm- allows for definition of the algorithm to be used in running the model's simulations; performs the same function as the **Algorithm** combo button on the Simulation Bar. **Runge Katta** (deterministic) is the default algorithm.

Start- starts the simulation based upon the model shown in the graphical interface, substance and parameter values defined in the editor windows or on the lefthand window, and the algorithm selected by one of the two aforementioned methods.

Pause- pauses a model that has been initiated but not yet completed.

Resume- resumes a model that has been paused.

Plot- opens a window with a plot of the simulation that was run by pressing the **Start** button. Even if the model is half-done, this function will plot the time-courses of substances as they are computed.

Sensitivity Analysis- opens the Sensitivity Analysis that can be used to monitor system output given a range of values for a single parameter. For more on this function, see the **Basic Sensitivity Analysis** section of the manual under **Running Simulations.**

Math:

Show Equations- opens up a window displaying all rate equations and parameter assignments as defined by the reaction stoichiometrics of the model. This is a useful tool for easily deriving rate equations without going through the trouble of working out each equation to decompose the production and consumption factors for each substance. See Figure 23 under the Dictyostelium aggregation network model for an example.

Customized Expressions- a programming parser within Dynetica similar to the Macros function within Excel; allows for the programming of an expression that can be used in separate locations of a model without copying each computation for each desired instance.

Plot Menu Bar

File:

Save Selected Data- allows for the curves shown in the plot to be saved to an output file which can be read by Notepad or Excel. To change which data is selected, click on the lefthand portion of the plot window where the different substances are provided. To include more than one substance use the **Shift** and **Ctrl** keys on the keyboard.

Save All Data- saves all data (with the assumption that time is the x-axis for all data points and that all substances on the y-axis constitute the entirety of the data) to an output file readable by Notepad or Excel.

Simulation:

Start/Pause/Resume- starts, pauses, and resumes the model simulation, respectively. These options have the same function as the ones under the Menu Bar of the main window.

View:

Log Scale in X Axis- plots all time courses using a log-scale on the x-axis; can also be triggered by pressing the toggle-button on the lower lefthand side of the plot window.

Log Scale in Y Axis- plots all time courses using a log-scale on the y-axis; can also be triggered by pressing the toggle-button on the lower lefthand side of the plot window.

Select All- selects all substances (on the y-axis) to be displayed on the plot.

Automated Updating- a toggle switch that either allows the simulation plot to be displayed as it is formed (with a percentage sign above the plot indicating what portion of the simulation has been computed) or to be left blank until the entire time-course is computed.

Graphical Interface Options

Note: these functions are accessible by right-clicking anywhere on the graphical interface window. To target a substance or reaction with these functions, simply right-click on those nodes or arrows.

Refresh System- reloads the graphical interface, inserting arrow lines between nodes and reactions that may not have been present immediately after their creation.

Show Node Names- toggles between displaying the names of each substance/expression and masking it so that only the nodes themselves are seen without labels.

Show Arrows- toggles on and off the red, green, grey, and orange arrows that connect all substances, reactions, and expressions in the graphical interface. This does not affect the actual connectivity of the system stoichiometry.

Node Size- allows for the increase and decrease of size of the nodes on the graphical interface.

Line Width- allows for the increase and decrease of line thickness of the arrows on the graphical interface.

Arrow Size- allows for the increase and decrease of tail length of the arrows on the graphical interface.

Graph Size- expands or shrinks the working area within the graphical interface. If the working area exceeds beyond those capable of display by the entire

Dynetica window, scrollbars at the edges of the graphical interface will appear to allow proper navigation around the whole of the system model.

Entity Menu: Highlighting a node by left- or right-clicking on it will allow for the usage of the entity menu: the only real way to manipulate nodes as if they were files in an operating system or words in a document. Other than highlighting this node in the graphical space, the entity menu can also be accessed by highlighting the designated substance/expression on the lefthand parameters window.

Rename- allows for the label on the node to be changed to something else; this change will be reflected in all reactions and expressions that previously called forth this node.

Edit- brings up the editor window; the same thing as double-clicking on any substance, expression, or reaction.

Delete- deletes an entity.

Clone- Dynetica's version of *copy*, creates another substance, reaction, or expression that has the same properties as its precursor. The name of this new node is the same as that of the previous one with a "_clone" tag to the end. For expressions, nothing changes. For substances, the new substance has the same initial value but is not implemented in any of the pre-existing reactions of the system. For reactions, some bugs still exist, but the stoichiometric and kinetic definitions of the clone should be the same as the original.

Parameter Options

Note: these functions are accessible by right-clicking anywhere on the parameters window.

New >> Substances- creates a new substance with a random placement on the graphical interface window; has the same function as the **Substance** option on the combo button of the bottom bar.

New >> Parameters- creates a new parameter which can be called forth in reaction kinetics formulae or in expressions as needed; has the same function as the **Parameter** option on the combo button of the bottom bar.

New >> New Expression- creates a new expression with a corresponding node (like the substances) also placed randomly on the graphical interface window; has the same function as the **ExpressionVariable** option on the combo button of the bottom bar.

New >> Reactions- creates a new reaction with the corresponding boxed arrow on the graphical interface; has the same function as the **ProgressiveReaction** option on the combo button of the bottom bar.

New >> Genomes- creates a new genome with genes and transcription kinetics to be defined; this function is not available in the default model format, but only if **New >> GeneticSystem** is selected under the **File** menu in the Menu Bar.

Expression Functions

Dynetica supports a wide array of functions which can facilitate the formation of a signal input or the processing of the system output. Functions like *pulse*, *step*, or Expressions can be plotted like other substances as a function of time in this respect (see the Lotka-Volterra model for an example). The latest version has vastly increased the palette of functions to be used in this context, and all these are listed below in Table 3.

Table 3. The mathematical operations and functions that are supported by Dynetica

	Symbol/expression	**Function**
Basic operations	+	Addition
	-	Subtraction
	*	Multiplication
	/	Division
	^	Raises to the power of the number to the right (exponent)
Basic functions [a]	sin(a)	Returns the sine of a
	cos(a)	Returns the cosine of a
	tan(a)	Returns the tangent of a
	sqrt(a)	Returns the square root of a
	exp(a)	Returns the exponential of a
	log(a)	Returns the natural logarithm value of a
Special functions (primarily used in expressions) [a]	abs(a)	Returns the absolute value of a
	sum(a,b,c,…)	Returns the sum of all substances called within the parenthesis
	step(a, b)	Returns 1 if a > b, and 0 otherwise (creates a step function when used as a signal input generator)
	compare(a, b)	Returns 1 if a > b, 0 if a = b, and –1 if a < b
	pulse(a, x, b)	Returns 1 if a < x < b, 0 otherwise (creates a pulse when used as a signal input generator)
	pulses(t,a,b,c)	Returns 1 if (a + n*b) <= t < (a + n*b + c) where n is any integer, creates a periodic pulsing input
	randomn(a, b)	Returns a random value between a and b
	rand()	Returns a random value between 0 and 1

	normal()	Returns a random number generated in a normal (or Gaussian) distribution, with a mean of zero and a standard deviation of 1
	min(a, b, c, …)	Returns the minimum value from the list of arguments
	max(a, b, c,…)	Returns the maximum value from the list of arguments

[a] Each of the symbols (a, b, c and x) may represent a simple variable or a mathematical expression.

Other

Print Graph- prints a copy of the graphical interface of the model to the default printer; if the model is large, the view of the graphical interface will be expanded so that the entire model fits on one page.

System Information- opens the System Information window containing annotations for the model; has the same functionality as the **Annotations** button in the **Edit** menu on the Menu Bar.

Source- opens up the compiled source code for the model onto the window that previously had the graphical interface. To bring up the graphical interface once again, simply toggle the button (which should then read **Graph**).

Delete- seems to have no function at this point.

Troubleshooting

If your model is not working properly, first try shutting down Dynetica and reopening it. If multiple models were opened at once (seen by the multiple tabs just below the Simulation Bar), Dynetica may mistakenly take the parameters from one model and use them in the formulation of the second model's simulation. Other unknown problems are known to occur when multiple models are opened at once, so try to avoid running simulations while two or more are open.

If closing the model and the program do not work, try looking for notational errors in the model, either the improper naming of a substance or reaction (remember that only underscores can be used, but no mathematical operators like dashes, slashes, or stars) or the improper inscribing of reaction stoichiometrics/kinetics. If this proves too taxing, try opening up the model using wordpad to get a clearer indication of the substance/reaction parameters. Models that are too complicated (and especially ones modeled with stochastic algorithms) will greatly increase computation time and the amount of memory used, so if this is the problem either try simplifying the model or running the model on a computer with a greater computational speed.