KINSOLVER: A simulator for computing large ensembles of biochemical and gene regulatory networks.

Boanerges Aleman-Meza[1], Yihai Yu[1], H.-B. Schüttler[2], Jonathan Arnold[3], Thiab R. Taha[1]

[1] Department of Computer Science, University of Georgia, Athens, GA 30602

[2] Department of Physics and Astronomy, University of Georgia, Athens, GA 30602

[3] Department of Genetics, University of Georgia, Athens, GA 30602

Abstract. With the sequence of many genomes now available the major challenge of functional genomics is 'reassembling the pieces'. One functional view of a living system is as a chemical reaction network, and new genomics technologies like RNA and protein profiling are providing ways to measure the state of these networks. The goal here is being able to simulate an arbitrary ensemble of hypothesized biochemical and genetic regulatory networks to predict what a cell is doing, *i.e*. to compute life, so that these predictions may be compared with the observed state of the system. The simulator KINSOLVER will solve chemical reaction networks, satisfying standard multiplicative mass balance kinetics, of arbitrary size, topology, rate constants, and initial conditions by 5 standard methods (Euler, Modified Euler, Runge Kutta (RK), Adaptive RK-Fehlberg, and LSODES). The simulator includes a simple Web-browser interface for specifying and refining a target reaction network as well as visualization tools to represent the network's behavior. The simulator is verified as rapidly solving in seconds (with benchmarks relative to GEPASI) some classic biological circuits like the *lac* operon and *qa* gene cluster as well as a new circuit, the repressilator, with oscillatory behavior. The LSODES method uniformly outperformed the other methods with a relatively large error tolerance of 0.01 and with a small error tolerance of 1E-6. The simulator is written in a nearly platform independent manner to simulate large ensembles of models and has a Web-browser interface to interact with the simulator by using Java and C++ at the back-end.

Availability. Software can be downloaded from http://webster.cs.uga.edu/~boanerg/mams or http://gene.genetics.uga.edu/stc.

Contact. Please contact arnold@uga.edu.

Keywords: genetic networks, biological circuits, chemical reaction network, simulator, biochemical and gene regulatory network, repressilator, model ensemble

1. Introduction

A living system can be viewed as a chemical reaction network (Beadle and Tatum, 1941). One of the central problems of functional genomics then becomes the identification of biochemical and gene regulatory networks describing how a living system functions (Arnold *et al*., 2004). Solving this problem begins by focusing on a particular process like lactose metabolism (Jacob and Monod, 1961), identifying the regulatory control exercised by genes, finding the products of these genes, and determining their role in relevant biochemical pathways. The resulting model of lactose metabolism is then a network or "biological circuit", in which genes and their products enter as nodes in the circuit [see Figure 1]. The term biological circuit refers to a model in biochemistry and molecular biology, which includes efforts to account fully for the effects of concentrations of each species, the time evolution of biochemical events, and the accumulation of transient intermediates (Purich and Allison, 2000). Validating a circuit depends upon our ability to simulate a particular reaction network and to predict how the network responds to various experimental perturbations. Perturbations may include gene knockouts, change in a substrate like a carbon source, or the addition of a protein inhibitor like a drug. As the information about this circuit accumulates, the biological circuit is modified in the light of new experiments, and new predictions are made to refine the circuit (Wagner, 2001).

These biological circuits can be partially identified for a few well-studied paradigms like the *lac* operon (Jacob and Monod, 1961), *trp* operon (Yanofsky and Kolter, 1982; Yanofsky, 2001), *GAL* gene cluster (Johnston, 1987), *qa* gene cluster (Giles *et al*., 1985), cell cycle (Sveiczer *et al*, 2000), and biological clock (Lee *et al*., 2000). Such circuits can display a diversity of dynamical behavior including a transient response, switch-like behavior, and oscillations. As circuits are coupled into larger networks they begin to display emergent properties (Bhalla and Iyengar, 1999). As their behavior becomes increasingly complex, so does the task of predicting their behavior. A simulator of a broad class of models and the generation of new testable hypotheses is required to understand circuit behavior. As new technologies for measuring the global response of a circuit through RNA and protein profiling (DeRisi *et al*., 1997; Gygi *et al*., 1999) become available, we are faced with the ultimate challenge of computing the behavior of an entire reaction network, i.e. of computing life. The behavior of the circuit is measured globally, and its success or failure as a scientific hypothesis is judged in this wider genomic context (Tomita *et al*., 1999).

In order to refine and examine the behavior of a biological circuit in a genomic context, a general purpose simulator is needed to compute life. Our goal here is to present a simulator of an arbitrary reaction network satisfying multiplicative mass balance kinetics (Bhalla and Iyengar, 1999) with a simple interface for specifying and refining the target reaction network as well as with visualization tools to represent the circuit's predicted behavior. A second goal is to support the efficient computation of a whole ensemble of 10^4-10^5 related models to facilitate both the characterization of model families (Alves and Savageau, 2000) as well as the identification of biological circuits (Battogtokh *et al*., 2002). A model ensemble is then a probability distribution over the parameter space of possible models deemed consistent with the available data on the reaction network. This model ensemble is specified once a figure of merit for selecting reasonable models is selected, such as the likelihood function or posterior distribution (Battogtokh *et al*., 2002).

In this general model on which the simulator is based, the cell is viewed as well-stirred, although it is possible to embed cell compartmentalization and scaffolding into a simulation with the existing simulator described here. The simulator allows an arbitrary number of species into a reaction if higher-order kinetics need to be hypothesized. In principle, the number of reactions and participating species is only limited by the array dimensions set in the simulator. The model of a reaction network is deterministic, and each reaction has a forward and backward reaction rate. Once the initial concentrations of all species are provided and the reactions specified, the time derivative of each species' concentration can be computed. The simulator KINSOLVER solves recursively a system of coupled nonlinear differential equations for the trajectories of all species in the reaction network. As illustrated by some of the examples, this task of solving these differential equations is not always straightforward, and there is a premium on speed in the

computing of each model in a large model ensemble. The resulting solution curves are presented over the Web via Java back-end processes.

A number of software packages exist for simulation of biochemical reaction networks, such as METAMODEL (Cornish-Bowden and Hofmeyer, 1991); GEPASI (Mendes, 1993, 1997); SCAMP (Sauro, 1993); KINSIM (Barshop *et al*., 1983; Dang and Frieden, 1997), MIST (Ehlde and Zacchi, 1995), and E-CELL (Tomita et al., 1999). Like E-CELL, KINSOLVER here has been developed in C++ to simulate large reaction networks using a variety of numerical integration methods; 5 numerical solution methods (as opposed to 2 methods in E-CELL) are built in with an option to add others so that challenging reaction networks like the Oregonator (Murray, 1993) can still be simulated. Input can take the form of multiple models as might arise in model fitting. The Web interface of KINSOLVER is produced by Java code in order to allow Web-accessibility of the simulator and to promote its platform independence. E-CELL allows a stochastic component to the models (KINSOLVER does not), while KINSOLVER includes the possibility of reaction velocities with Michaelis-Menten form and cooperativity effects between activators or inhibitors. Like GEPASI (Mendes, 1997), KINSOLVER is not constrained by the size of the reaction network and has a menu-based interface, but it differs in the kinds of numerical integration tools available and in having a Web interface to give it platform independence. There are three features that separate KINSOLVER from most other simulators, its platform independence, flexibility in solution procedures, and its capability to simulate efficiently large (10^4 -10^5) ensembles of models.

Other modeling approaches for gene regulatory and biochemical networks are being pursued by others. These include Bayesian networks (Murphy and Mian, 1999; Friedman *et al*., 2000), neural networks (Weaver et al., 1999), and boolean networks (Huang, 1999; Shmulevich *et al*., 2001, 2002). The ultimate success of these competing approaches will be determined by their ability to predict successfully reaction network behavior. The advantage of the approach here is that it is well-rooted in chemistry and physics.

Here we present a description of the underlying model on which KINSOLVER is based. Then the implementation of the simulator is described. A simple example is used to illustrate how the simulator is used over the Web and to establish a notation for reaction networks. Some important real examples are used to evaluate the solution procedures of the underlying coupled nonlinear differential equations relative to GEPASI (Mendes, 1997) with a particular focus on simulating model ensembles. The paper concludes with limitations of KINSOLVER and some needed extensions to the modeling approach, which KINSOLVER implements.

2. Kinetics Model

Models that represent biochemical reaction networks including genes and their products allow us to predict what the cell is doing. The standard multiplicative mass balance kinetics leads to a specification of an underlying system of coupled differential equations that describes a particular reaction network (Bhalla and Iyengar, 1999). The specification of a reaction network model begins with the construction of a circuit diagram as in Figure 1, which captures the relationships of reactants, products, the reactions in which the species participate, and the relationships of the reactions. This biological circuit in Figure 1 is a kinetic model of DNA, RNA, and proteins involved in carbon metabolism (Bhalla and Iyengar, 1999; Weng *et al*., 1999). The example in Figure 1 is one of two early paradigms of eukaryotic gene regulation and represents how the model system *Neurospora crassa* utilizes quinic acid as a sole carbon source (Geever *et al.,* 1989). The *qa* gene cluster in *N. crassa* is a good example because of its relative simplicity and because its circuit structure is shared with many other gene regulatory systems.

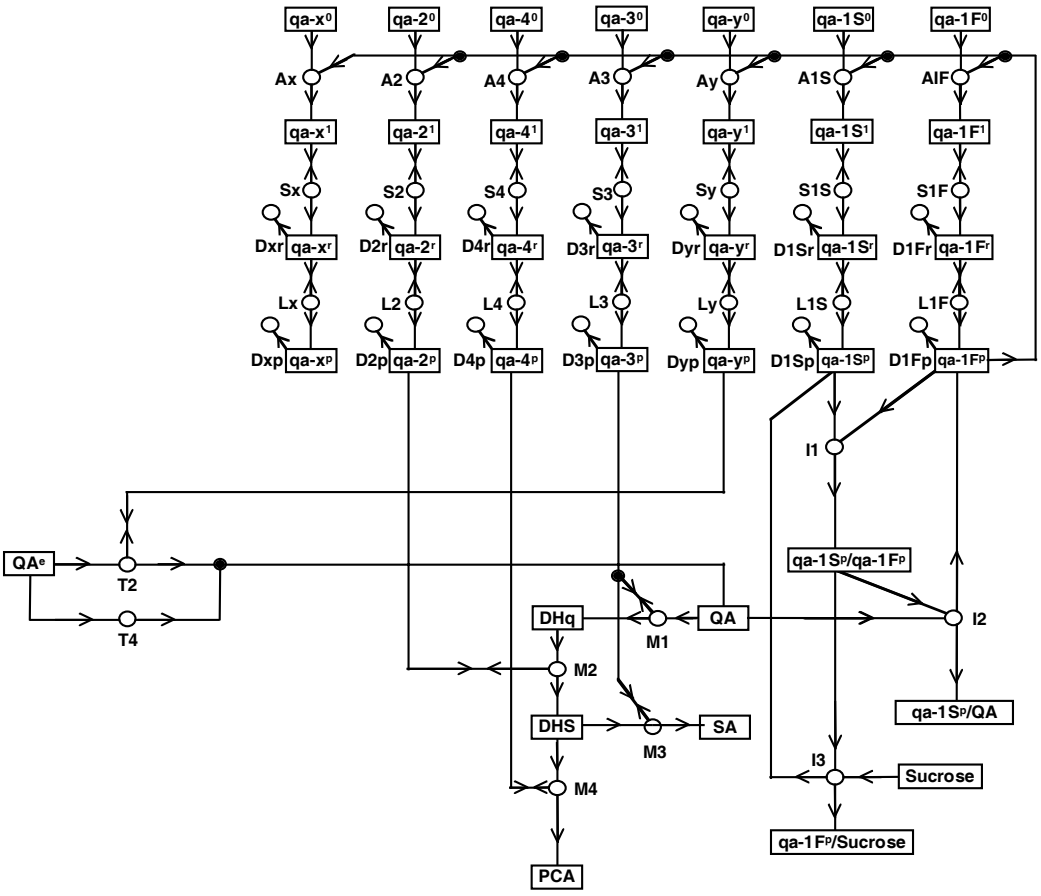

Figure 1. Kinetics model of quinic acid metabolism represented as a biological circuit. The figure is redrawn from Figure 1 of Kochut *et al.* (2003).

A kinetics model is a specification of reactions between hypothesized molecular participants. Almost all of the examples in this paper relate to combustion to produce energy (sometimes for a cell), and so to illustrate the simplest kind of combustion we begin with that of molecular Hydrogen and Oxygen in Figure 2. The reactants or products are represented as boxes. Reactants connect to other species by reactions represented with circles. Incoming arrows are used to indicate the reactants entering a reaction, and outgoing arrows are used to indicate the products of a reaction. The arrows also define the forward direction of the reaction. The presence of a double arrow is used to indicate that a species being pointed to appears on both the left hand side and right hand side of a reaction. A reaction can have arbitrary numbers of input and output species, that is, can involve higher-order kinetics.

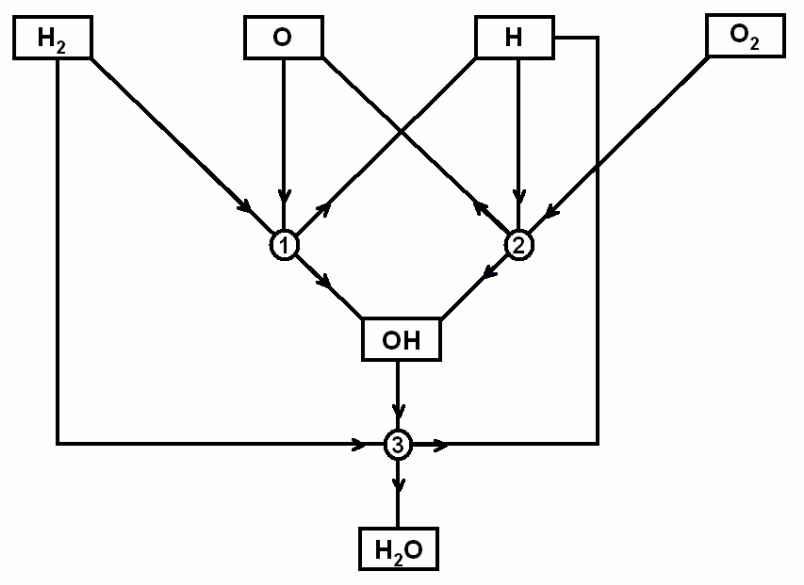

Figure 2. Water Model I.

For simplicity, we will explain the construction of a full mass balance kinetics within the context of the model depicted in Figure 2. This simple Hydrogen/Oxygen Combustion Model will be referred to as *Water Model I*. It consists of 3 reactions with 6 participating species (***H***, H_2, ***O***, O_2, ***OH***, H_2O).

Such a model can be easily expressed as a series of *chemical reactions* or *kinetics reactions*. These (reversible) chemical reactions are of the form $S_1 + S_2 \leftrightarrows S_3 + S_4$, where S_i are the participant species. As explained in (Salvadori and Schwarz, 1954), reversible reactions are chemical reactions in which one molecule of ***A*** is transformed into one molecule of ***C*** and may actually consist of two successive reactions in which at first one molecule of ***A*** is transformed into one molecule of ***B*** and then one molecule of ***B*** transformed into one of ***C***; moreover, the intermediate product ***B*** and the final product ***C*** may spontaneously revert to ***A***, and ***B***, respectively. This can be written in two reactions $A \leftrightarrows B$, and $B \leftrightarrows C$ where the symbol "$\leftrightarrows$" is used to indicated that the reaction is reversible with forward and backward reaction rate constants. Note that the backward reaction rate constants are often very small, and in some cases can be set to zero.

In order to write down the chemical reactions from Figure 2, each reaction (depicted with circles) is written independently. Take the reaction represented by circle 1 as an example. Arrows connecting reactant H_2 and reactant ***O*** to circle 1 denote that they are reactants. The outgoing arrows from reaction 1 towards ***H*** and ***OH*** indicate that these are the products. The arrow also defines the forward reaction. This is expressed as a chemical reaction $H_2 + O \leftrightarrows H + OH$. When the stoichiometric coefficients are other than one, this can be represented by multiple lines to or from a species or by associating a number with each arrow for a more concise representation.

Similarly, for reaction 2, ***H*** and O_2 are the reactants, and ***O*** and ***OH*** are the reaction products. This is written as $H + O_2 \leftrightarrows O + OH$.

For reaction 3, arrows from H_2 and ***OH*** denote that they are the input reactants, and the outgoing arrows to H_2O and ***H*** depict them as products. The chemical reaction is written as $H_2 + OH \leftrightarrows H_2O + H$.

The three reactions of the Figure 2 can be written as:

$$\begin{aligned} &\text{reaction}_1 && H_2 + O \leftrightarrows H + OH \\ &\text{reaction}_2 && H + O_2 \leftrightarrows O + OH \\ &\text{reaction}_3 && H_2 + OH \leftrightarrows H_2O + H \end{aligned} \qquad (1)$$

To obtain the system of differential equations from the reaction network, a participant species is viewed in terms of the changes it goes through in the series of reactions. For the case of a reaction like $A \rightarrow B$, the species A is converted into B at some rate k_1. This can be expressed as $\frac{d(A)}{dt} = -k_1 [A]$. The negative sign of k_1 indicates that A is consumed/reduced in the reaction. For the same reaction, the equation for species B can be expressed as $\frac{d(B)}{dt} = k_1 [A]$. The absence of a negative sign means that B is "produced" from A at the rate given by k_1.

For a *reversible* reaction, the input participants are consumed at a "forward rate" and produced at a "backward rate". For example, for the reversible reaction $H_2 + O \leftrightarrows H + OH$ of equations (1), the differential equation for H_2 can be expressed as $\frac{d(H_2)}{dt} = -k_{f1} [H_2] [O] + k_{b1} [H] [OH]$. The concentration of H_2 is reduced when it reacts with O at the forward rate constant k_{f1} as indicated by the negative sign. The final products H and OH revert back to H_2, and O at the rate given by the backward constant k_{b1}. This is expressed in the second term with a positive sign for the backward rate constant.

Signs are different when writing the differential equation for species that are *produced* in the reaction. Take again the reversible reaction $H_2 + O \leftrightarrows H + OH$. The species H is produced from the reaction of H_2 and O at a forward rate k_{f1}. The products H and OH revert back to H_2 and O at a backward rate k_{b1}. The positive sign for k_{f1}, and the negative sign for k_{b1} indicate this in the equation $\frac{d(H)}{dt} = +k_{f1} [H_2] [O] - k_{b1} [H] [OH]$.

Note that these equations can be further simplified as in (Rice and Do, 1995; Salvadori and Schwarz, 1954), but this mathematical simplification was not considered because of the symbolic computation involved.

The full multiplicative mass balance kinetics of *Water Model I* (1) is then obtained by collecting for each species all production and consumption rate terms from all the reactions in which the species participates, *i.e.*:

$$\begin{aligned} \frac{d(H_2)}{dt} &= - k_{f1} [H_2] [O] + k_{b1} [H] [OH] - k_{f3} [H_2] [OH] + k_{b3} [H_2O] [H] \\ \frac{d(O)}{dt} &= - k_{f1} [H_2] [O] + k_{b1} [H] [OH] + k_{f2} [H] [O_2] - k_{b2} [O] [OH] \\ \frac{d(O_2)}{dt} &= - k_{f2} [H] [O_2] + k_{b2} [O] [OH] \\ \frac{d(H)}{dt} &= + k_{f1} [H_2] [O] - k_{b1} [H] [OH] - k_{f2} [H] [O_2] + k_{b2} [O] [OH] + k_{f3} [H_2] [OH] - k_{b3} [H_2O] [H] \\ \frac{d(OH)}{dt} &= + k_{f1} [H_2] [O] - k_{b1} [H] [OH] + k_{f2} [H] [O_2] - k_{b2} [O] [OH] - k_{f3} [H_2] [OH] + k_{b3} [H_2O] [H] \\ \frac{d(H_2O)}{dt} &= + k_{f3} [H_2] [OH] - k_{b3} [H_2O] [H] \end{aligned} \qquad (2)$$

where k_{bi} and k_{fi} are the backward and forward reaction rate constants respectively, for reaction i.

3. Systems and Methods

The simulator receives an input file that is a representation of the chemical reactions instead of the system of ODE's. The participating species are assigned concentrations at an initial time t_0, and the system is solved for concentrations at a final time t_n. The parameters and initial conditions are all part of the input file, which has a fixed format. The simulator takes the input file (which can describe an ensemble of models to be simulated) and solves the differential equations (for each model listed) and generates an output file (with the output for each model simulated). The simulator or simulation engine KINSOLVER can be used as a standalone program (named *kin.c* or *kin.for*).

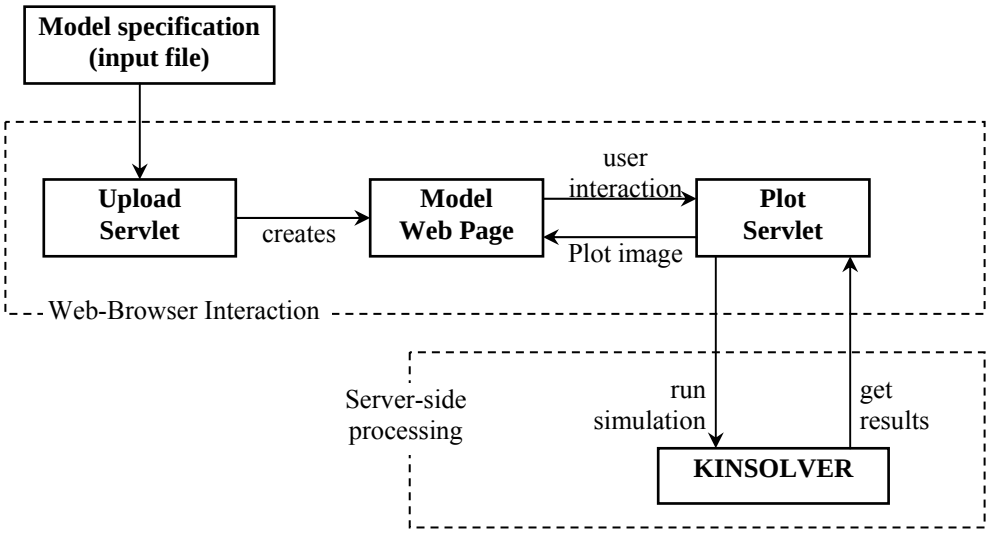

Figure 3. Simulator Process.

The simulator is written in both Fortran77 (H.-B. Schüttler) and C++ (Aleman-Meza) and available at http://gene.genetics.uga.edu under "Computing Life" and http://webster.cs.uga.edu/~boanerg/mams . The simulators have been executed on a variety of platforms including SUN Solaris UNIX for ULTRA SparcStations.

In addition, Aleman-Meza has constructed an input/output *Web-browser* interface to execute the simulator in a Web-based environment at http://gene.genetics.uga.edu/~aleman/kinnew on Solaris 8.0 on a SunFire V250 server. The simulator + interface permits a text file describing a kinetics model to be uploaded and edited as in Figure 4. The different parts of this process are illustrated in Figure 3. The interface invokes the simulator and produces trajectories of the species to be viewed over the Web as in Figure 5.

The plotting tool (based on GNUPLOT) is accessible by a Web Browser. After the researcher has defined a model in an input file `kin.i01`, he or she submits it through a Web page. The submission of the input file is processed by a Java program that generates a "workspace Web page for the model". At this Web page the model parameters can be modified, and the simulation results are displayed graphically on the Web browser, using Java Servlets (see Rossbach and Schreiber, 2000). Figure 4 gives an example of a "workspace" Web page for the model *Water Model I*. There are *buttons* that can be used to run the simulation and obtain a plot of each species vs. time. Running the simulation and producing a plot of the results is done by the *plot* button of the workspace web page of a model.

The "Computing Life" Web Page (http://gene.genetics.uga.edu/stc/) from the Genetics Department of the University of Georgia contains links to the plotting tool explained above. In Figure 5 there is a species vs. time (***OH*** vs. time) plot generated by the plotting tool from *Water Model I*.

A general deterministic kinetics model satisfying mass balance will be the input to the simulator, and its corresponding system of ODE's will be solved numerically. A numerical solution of a system of ODE's is a table of approximate values to the exact solution. Starting at the initial time t_0 with given initial values or initial concentrations of the species, the trajectory of the system is followed by evaluating $f(t_0, y_0)$–the slope at that point. This helps to predict the value of y_1 of the solution at time t_1, where $t_1 = t_0 + h$, and h = (final time – initial time) / (number of time steps). Note that the simulator does not receive as input the system of differential equations. It receives an input file in plain–text format which is a representation of the species, chemical reactions, and related parameter values.

The numerical methods used in the simulator to simulate a biological circuit are:

1. Euler Method (first order)
2. Modified Euler Method (second order)
3. Runge–Kutta Method of Order 4
4. Adaptive Runge–Kutta–Fehlberg Method
5. LSODES Method

The adaptive method automatically adjusts the integration step size through the numerical solution of the system of ODE's. Fehlberg (1969) developed the commonly known *Fehlberg method of order 4*, which is in the family of Runge–Kutta methods. The method also gives a global error bound for each solution. See (Hairer *et al*., 1987) for more details on numerical methods for solving ordinary differential equations. GEPASI (Mendes, 1993) in contrast uses a hybrid method (LSODA) that starts with a nonstiff method (i.*e*., Adams-Moulon) and switches to a stiff method (BDF). See the work of Petzold (1983) or Zwolak *et al*. (2001) for a description of LSODA. We have also implemented a new version of LSODA within KINSOLVER called LSODES (Petzold, 1983) to enable a clean comparison of methods under one system, namely KINSOLVER. The LSODA method is a version of LSODE (Hindmarsh 1980), but it treats the Jacobian matrix as either dense or a banded matrix, and the calculation for the switch can be very expensive. In contrast, the LSODES method (Hindmarsh 1983), which is the actual stiff solver we have incorporated into KINSOLVER, is another version of LSODE, and it treats the Jacobian matrix as sparse, and uses components of the Yale Sparse Package (Eisenstat *et al*., 1977, 1982)

4. Example

Here we illustrate the results of a given input file for the model depicted in Figure 2 as a simple representation of the three reactions in equation (1). An example of an input file with arbitrary reaction constants and arbitrary initial concentrations is given:

```
 data set Hydrogen Combustion I: 2 H_2 + O_2  ->  2 H_2O
   nspec   nreac
       6       3
           time0           time1   ntime  ntskip   jtime
           0.000        10.00000    1000     100       2
         namespec/xspec0   jfix
            H_2
                  6.000       0
            O_2
                  3.000       0
            O
                  0.000       0
            H
                  0.010       0
            OH
                  0.000       0
            H_2O
                  0.000       0
H_2+O:     rkfor           rkbak  nipart  nopart    jkin
          1.0000          0.0200       2       2       1
            H_2
            O
                             OH
                             H
O_2+H:     rkfor           rkbak  nipart  nopart    jkin
          1.0000          0.0200       2       2       1
            O_2
            H
                             OH
```

```
                          O
OH+H_2:    rkfor          rkbak  nipart  nopart    jkin
         0.5000         0.0100       2       2       1
           OH
           H_2
                          H_2O
                          H
```

The input file has to adhere to a specific structure. The file must be named "`kin.i01`". It may contain one data set or multiple data sets. Each data set must begin with a text line containing the character string, "`data set`". Each data set will simulate one time evolution of the kinetics equations for the reaction constants and initial species concentrations specified following the *`data set`* line.

Each data set consists of the following set of input parameters, to be entered in the format shown in the previously displayed sample input file. Note that numerical input values and text strings can not be entered on the same input line.

nspec number of participating species

nreac number of reactions

time0, **time1** start and stop times of kinetics time evolution

ntime number of time steps. The time step width $\boldsymbol{h}$ is computed as $\boldsymbol{h} = \frac{\boldsymbol{time_1 - time_0}}{\boldsymbol{ntime}}$

ntskip number of time steps to skip in the *dense* output file.

jtime integration method to use, (1) Euler, (2) Modified Euler, (3) Fourth order Runge–Kutta, (4) Adaptive Runge–Kutta–Fehlberg, (5) LSODES.

namespec (list) user supplied name of each participant species. All printable ASCII characters, except blanks are allowed as species names

xspec0 (list) Starting concentration of each participant species at time $\boldsymbol{t = time_0}$

jfix It is set to zero to let the concentration of the species evolve according to the kinetics rate equations. It is set to 1 to keep the initial concentration fixed at its starting value, this simulates an unlimited supply of the participant species

Reaction List

rkfor, **rkbak** forward and backward reaction rate constants for each reaction

nipart, **nopart** number of input and output participating molecular species for each reaction. For example: In the reaction $\boldsymbol{H_2 + O \leftrightarrows OH + H}$ the species $\boldsymbol{H_2}$ and $\boldsymbol{O}$ are input, hence ***nipart* = 2**, the species $\boldsymbol{OH}$ and $\boldsymbol{H}$ are output, hence ***nopart* = 2**.

In a reaction like $\boldsymbol{H + H_2 \leftrightarrows H + H + H}$, the species $\boldsymbol{H}$ and $\boldsymbol{H_2}$ are input, hence ***nipart* = 2**, the species $\boldsymbol{H}$ gets generated as output 3 times, hence ***nopart* = 3**.

jkin Set to 1. (In the Fortran77 version *kin.for,* setting jkin = 11 yields Generalized Michaelis-Menten kinetics and setting jkin = 22 yields cooperatively activated/inhibited transcription kinetics for a network as explained in the documentation).

Reaction participant lists Following the parameter line specifying ***rkfor***, ***rkbak***, etc., for that reaction, the names of the participating species must be entered with only one name per line such that the names of the input species are followed by the names of the output species. If more than one molecule of a certain species gets consumed or generated in a reaction, then the name of that species must be listed as many times (as input and/or as output) as it enters into the reaction. Example: In the reaction $\boldsymbol{H_2 + O \leftrightarrows OH + H}$ with specified ***nipart* = 2**, ***nopart* = 2**, and we list species names as follows:

```
H_2
O
OH
H
```

as the participating species names. The code will automatically recognize that the first 2 names listed ($\boldsymbol{H_2}$, $\boldsymbol{O}$) are input species and that the remaining 2 names ($\boldsymbol{OH}$, $\boldsymbol{H}$) are output species.

The input file is validated and (if valid) uploaded to a Web page as shown in Figure 4, where it can be modified. Model input files can be concatenated to simulate ensembles of models with *kin.c* or *kin.for*, resulting in concatenated output files.

There are three output files:

- File **`kin.o01`** contains a complete listing of all the input parameters used in the input file. The file also contains the initial/final species concentrations, final reaction concentrations (defined below), and time dependent species concentrations. The reaction concentration (***xreac***) records the net number of times a reaction event (***ireac***) has happened from the start of the kinetics evolution (***time0***) until some time point (***timei***). Here the number of reaction events is defined as the number of ***forward*** events minus the number of ***backward*** events. The reaction event number is measured, *e.g.,* in absolute number of events, absolute numbers of moles of events, or number of events per volume, depending on, and consistent with, the corresponding choice of units for the numbers of molecules of the species participating in the reaction.
- File **`kin.o02`** lists the concentrations of all of the participating species over time. Each of them is presented as tabular dense output with values for the iteration (***itime***), time (***timei***), and concentration (***xspec***). The ***ntskip*** parameter in the input file causes the simulator to skip over a certain number of integration time steps in listing the results in the output file.
- File **kin.o03** contains the time-dependent species concentrations in a format easy to use by plotting subroutines.

An example of a `kin.o02` output file follows:

```
#
#
# data set Hydrogen Combustion Modified
#  time-dep. species concentrations
#
#
#   ispec
#       1
# namespec:
#             H
#  itime         timei         xspec
       0    0.000000E+00    1.000000E+00
       1    1.000000E-03    9.999600E-01
       2    2.000000E-03    9.999200E-01
       .           .               .
       .           .               .
       .           .               .
     999    9.990000E-01    9.664881E-01
    1000    1.000000E+00    9.664597E-01


#
#   ispec
#       2
# namespec:
#             H2O
#  itime         timei         xspec
       0    0.000000E+00    0.000000E+00
       1    1.000000E-03    2.909266E-05
       2    2.000000E-03    3.731986E-05
       .           .               .
       .           .               .
       .           .               .
     999    9.990000E-01    3.075079E-05
    1000    1.000000E+00    3.074627E-05
```

The Java Servlet process then generates a plot of the species requested as shown in Figure 5.

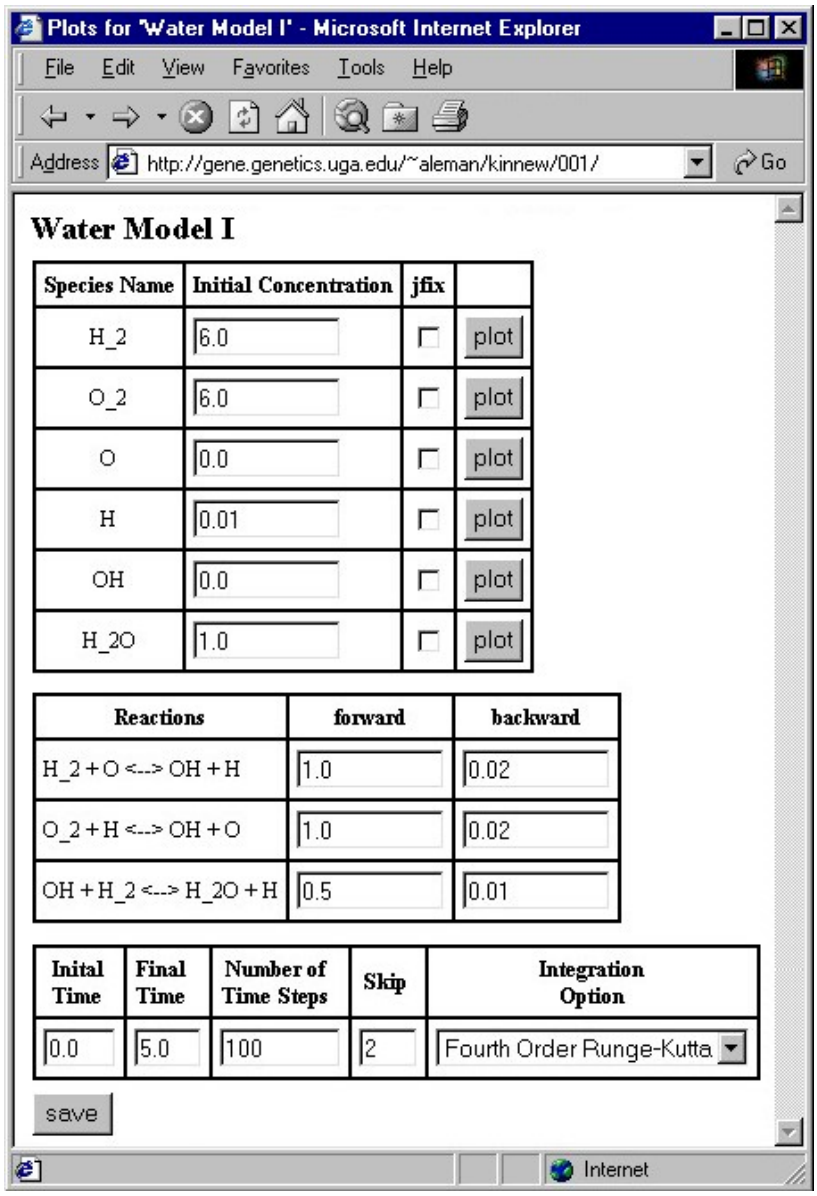

Figure 4. Workspace web page for *Water Model I*.

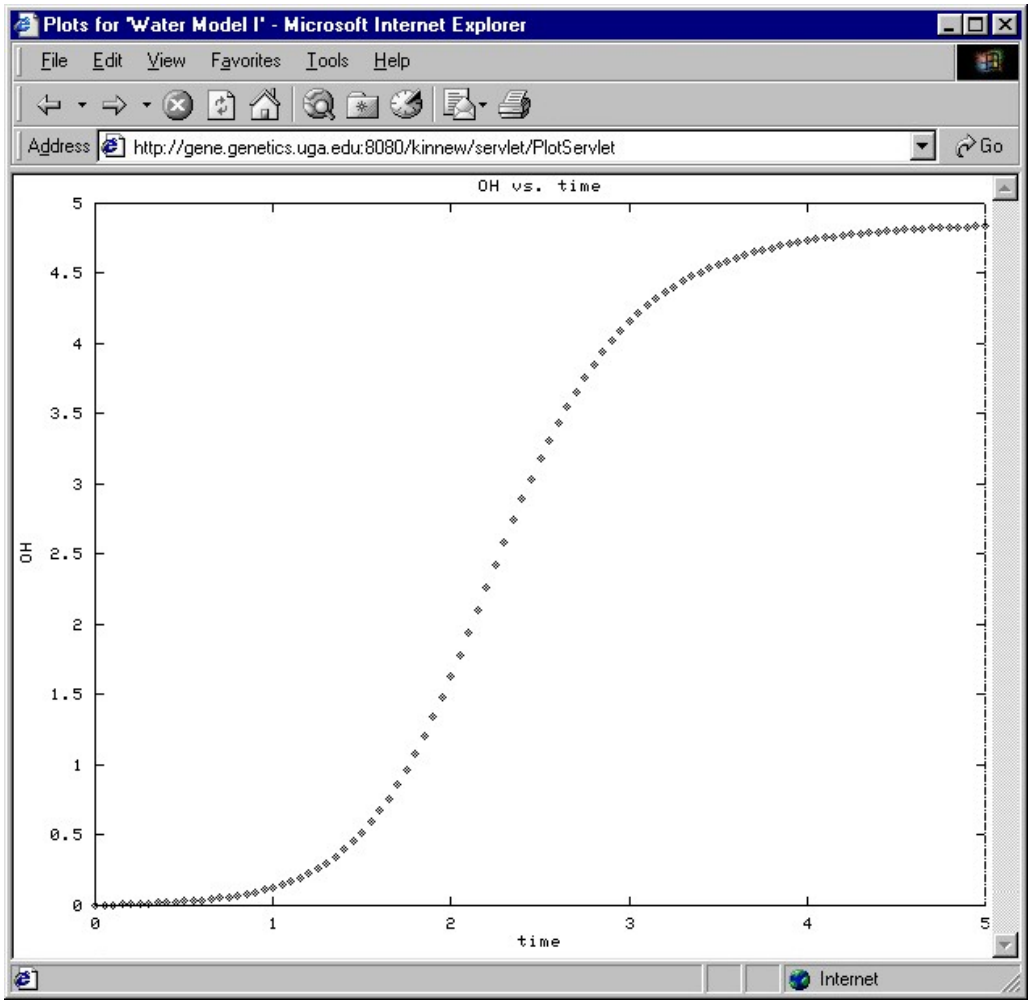

Figure 5. Species vs. time plot for ***OH*** *–Water Model I*.

5. Results

When trying to optimize the program to measure its efficiency, it was required to know how much *processor time* or *CPU time* was used at any given time point. The timing of the numerical results is in seconds on a SunFire V250 server running Solaris 8.0 (unless otherwise stated), and it was estimated by calculating an average over 8 executions.

For those interested in examining one or a few reaction networks, the goal is finding a solution of high precision (10^{-6} error). This is not our goal. Our goal is to obtain the behavior of a large (10^{4}-10^{5}) ensemble of models with biologically reasonable accuracy (10^{-2} error) to obtain general properties of families of reaction networks (Alves and Savageau, 2000) and to use new methods of ensemble identification of reaction networks (Battogtokh *et al*., 2002). The principal barrier to simulating a model ensemble is the time spent in computing the trajectory for each member of the ensemble. One means to reduce this barrier is to reduce the demand for high accuracy on each model simulation.

Our expectation is that higher order methods will reach the solution with fewer integration time steps than lower order methods (an expectation based on the historical preoccupation with high precision solutions). However, the higher order methods may do so at a higher computational cost because they require more function evaluations at every time step. As a benchmark our simulator KINSOLVER was compared with GEPASI (Mendes, 1993) on a PC with a Pentium III processor running Windows 98. The package

KINSOLVER was compiled with DJGPP (32-bit C/C++ development system for Intel PCs running DOS) freely available at http://www.delorie.com/djgpp/. The solutions of the two simulators in all cases (but one, the Oregonator) were identical.

The examples in Tables 1 through 4 are arranged in order of increasing stiffness of each biological model tested. A dimensionless stiffness index S defined to be the time needed to reach equilibrium, times the absolute value of the most negative eigenvalue of the Jacobian matrix (*i.e.*, associated with the right hand side of equation (2)) was calculated (Hindmarsh, 1983). This stiffness measure varies over the time course of the solution. The eigenvalues reveal the modes of the system. The most negative eigenvalue tells us the most dominant mode at this time point to limit the step size. If this value is too big in magnitude compared to the time span, in which we are interested, then it just takes too much time to get to that time point since the step size is very small. This is the rationale for the stiffness measure. If S is on the order of 1000 or higher, we consider this problem to be stiff, and if it's less than 10, we consider the problem to be nonstiff.

In Table 1 there is a summary of the number of integration time steps by different solution methods for a model of the *qa* gene cluster named "QA–Tr.B" with 39 participant species and 45 reactions. The stiffness index S varies from 400 to 2500, but is usually on the order of ~1000. The input file is available at http://gene.genetics.uga.edu/stc. The number of time steps shown yields a relative error less than 0.01 and less than 1E-6.

Relative error is calculated relative to the fourth–order formula of the Adaptive method using 20,000 time steps (100,000 time steps for the *lac* operon model discussed later).

Table 1. Times required for 6 differential equation solvers to simulate the *qa* gene cluster model (QA–Tr.B). Statistics are averages of 8 independent runs.

Method **Under Solaris:**	No. Derivative Function evaluation	Relative Error <.01		Relative Error < 1E-6	
		Time steps required	Time (seconds)	Time steps required	Time (seconds)
Euler	1,000	1,000	0.02	100,000,000	-
Modified Euler	1,600	800	0.04	100,000	4.57
Runge Kutta order 4	2,800	700	0.07	3,163	0.31
Adaptive RK–Fehl-berg	(4,068)	(678)	0.12	(678)	0.12
LSODES in C	-	37	0.01	229	0.0325
Under Windows 98:					
Adaptive RK–Fehlberg		-	-	(678)	0.11
Euler		1,000	0.01	-	-
*GEPASI**		N/A	0.05	N/A	0.07

*In GEPASI the inputted relative tolerance and absolute tolerance were set equal and given the value of 0.01 or 1E-6.

It can be seen in Table 1 that the number of time steps required for each method depends on its order. Nonhybrid higher order methods require fewer time steps, but they perform more computations per time step. Also generally the number of derivative evaluations required by the method helps to explain the time in seconds for the method with relative error less than 0.01. In the case of a targeted relative error of less than 0.01 the reduction in the required time steps in a higher order method other than LSODES is not enough to compensate for the computational time needed for finding the overall solution for a relative error < 0.01. The function evaluations are expensive to compute; therefore, the fastest method (excluding LSODES) is the one that computes the solution by doing fewer function evaluations per step, in this case

the Euler method. All five methods achieved less than 0.01 relative error. In contrast with the more traditional targeted relative error of 1E-6, the extra effort of a higher order method (other than LSODES) pays off in time. The hybrid LSODES method outperformed the other methods for a relative error of 0.01 or 1E-6.

The following table shows similar results for another model of the *qa* gene cluster named "QA–A" with 37 participant species and 43 reactions with input file on the same Web site. The stiffness measure S varied from 250 to 2000. The number of time steps required by each method to achieve a relative error less than 0.01 and 1E-6 is shown in Table 2.

Table 2. Times required for 6 differential equation solvers to simulate the *qa* gene cluster model (QA–A). Statistics are averages of 8 independent runs.

Method **Under Solaris:**	No. Derivative Function evaluation	Relative Error < .01		Relative Error < 1E-6	
		Time steps required	Time (seconds)	Time steps required	Time (seconds)
Euler	2,900	2,900	0.07	100,000,000	-
Modified Euler	2,200	1,100	0.04	100,000	4.35
Runge Kutta order 4	3,200	800	0.08	3,163	0.30
Adaptive RK–Fehl-berg	(4,416)	(736)	0.12	(736)	0.12
LSODES in C	-	39	0.011	180	0.029
Under Windows 98:					
Adaptive RK–Fehlberg		-	-	(736)	0.39
Euler		2,900	0.05	-	-
*GEPASI**		N/A	0.05	N/A	0.10

* In GEPASI the inputted relative tolerance and absolute tolerance were set equal and given the value of 0.01 or 1E-6.

In the above table more time steps are required than in Table 1. It can be seen that there is a bigger difference between the number of time steps by the Euler method vs. the modified Euler method at 0.01 relative error. The Euler method performs comparably to the Modified Euler Method because it computes fewer function evaluations per time step than the modified Euler method. In the case of a relative error of 0.01 again the conclusion reached is that a simpler method is faster if the LSODES method is not considered. If high precision (1E-6) is sought, then higher order methods (other than LSODES) are preferred. Under both scenarios for the relative error the hybrid LSODES method outperformed the other four methods, and its advantage over other methods has increased to at least 10X at 0.01 relative error with the higher stiffness of the problem. Relative to GEPASI, the Euler method is comparable for a relative error of 0.01, but the modified Euler method would have outperformed GEPASI for a relative error of 0.01. GEPASI performs a factor of 4 faster than Adaptive RK – Fehlberg Method in achieving the high precision (1E-6) solution.

In the following table are results for a model named "QA–Tr.A" with 37 species and 43 reactions with input file on the Web site http://gene.genetics.uga.edu/stc. The stiffness measure S varies from 1000 to 8000. The number of time steps in Table 3 shown achieved a relative error less than 0.01 and 1E-6.

Table 3. Times required for 6 differential equation solvers to simulate the *qa* gene

cluster model ("QA–Tr.A"). Statistics are averages of 8 independent runs.

Method **Under Solaris:**	No. Derivative Function evaluation	Relative Error < .01		Relative Error <1E-6	
		Time steps required	Time (seconds)	Time steps required	Time (seconds)
Euler	3,300	3,300	0.08	100,000,000	-
Modified Euler	6,000	3,000	0.13	100,000	4.35
Runge Kutta order 4	9,200	2,300	0.22	3,163	0.30
Adaptive RK–Fehl-berg	(12,078)	(2013)	0.32	(2013)	0.32
LSODES in C	-	23	0.02	197	0.06
Under Windows 98:					
Adaptive RK–Fehlberg		-	-	(736)	0.17
Euler		3,300	0.06	-	-
*GEPASI**		N/A	0.05	N/A	0.08

* In GEPASI the inputted relative tolerance and absolute tolerance were set equal and given the value of 0.01 or 1E-6.

This particular data set requires many more integration time steps because the initial concentration of some participant species is very high compared with the concentration of other species. Therefore, it exhibits increased *stiffness*. The difference between the number of time steps required by the Euler method vs. the modified Euler method is small for a relative error less than 0.01, but the modified Euler method does two function evaluations per time step as compared with one computed by the Euler method. This again causes the Euler method to perform better in time measurements for a relative error less than 0.01. In this example GEPASI slightly outperforms KINSOLVER's Euler Method for a relative error of 0.01 and is approximately twice as fast as KINSOLVER's Adaptive RK-Fehlberg Method for a high precision solution. The LSODES method outperformed the other four methods at both low and high relative error. The relative advantage of LSODES has increased to at least 4X for 0.01 relative error.

The first biological circuit to be explored in detail was the *lac* operon in *Escherichia coli* (Jacob and Monod, 1961), and the model is more complicated than that for the *qa* cluster. The stiffness measure varies from 6,000 to 40,000. Results for a circuit model of the *lac* operon in *E. coli* named "LAC–PTS" with 64 species and 69 reactions are shown in Table 4. The number of time steps shown gives a relative error less than 0.01 but in order to achieve convergence it was necessary to impose a target for the relative error of 0.001%.

Table 4. Times required for 6 differential equation solvers to simulate

the *lac* operon model ("LAC–PTS"). Statistics are averages of 8 independent runs.

Method **Under Solaris:**	No. Derivative Function evaluation	Relative Error < .01		Relative Error < 1E-6	
		Time steps required	Time (seconds)	Time steps required	Time (seconds)
Euler	103,000	103,000	3.67	1 E09	-
Modified Euler	204,000	102,000	7.20	1 E06	7.19
Runge Kutta order 4	293,200	73,300	11.20	73,300*	11.20
Adaptive RK–Fehl-berg	(182,034)	(30,339)	7.64	(30,339)	7.64
LSODES in C	-	21	0.45	59	0.64
Under Windows 98:					
Adaptive RK–Fehlberg		-	-	(30,339)	4.34
Euler		103,000	2.31	-	-
*GEPASI**		N/A	0.11	N/A	0.59

* In GEPASI the inputted relative tolerance and absolute tolerance were set equal and given the value of 0.01 or 1E-6.

This circuit contains about double the number of species and reactions than in the other *qa* gene cluster circuits tested; therefore, computing the solution takes longer. Values given by the Fourth order Runge–Kutta method with 100,000 time steps were considered the exact solution. The number of time steps could be reduced to 73,300 and still achieve a relative error less than 0.001% with the more accurate approximation to the solution. The Adaptive Runge Kutta–Fehlberg method is capable of computing the solution faster than the Fourth order Runge–Kutta Method. Euler and modified Euler methods need more than 100,000 time steps to compute a solution with a relative error less than .01, but still Euler's method was faster among nonhybrid methods. Modified Euler's method performs at a speed similar to the Adaptive Runge Kutta–Fehlberg Method for a relative error of 0.01. For a high precision solution higher order methods were faster than lower order methods. In this example GEPASI outperformed KINSOLVER's Adaptive RK-Fehlberg Method for relative errors of 0.01 or 1E-6, but LSODES outperformed all methods for low and high relative errors.

The adaptive Runge Kutta–Fehlberg method adjusts the integration step *h* as the solution evolves to produce a solution satisfying a user–specified accuracy. The number of integration time steps is not fixed in the input file for this adaptive method (thus we used parenthesis enclosing the number of time steps required to indicate that is not set by user). This method does five function evaluations per time step, and in the time–course integration adjusts the time interval *h*, having to compute again some iterations with the new *h* interval. This method is robust and gives an error estimate. Yet, the LSODES method outperformed the Adaptive RK-Fehlberg Method for low and high relative errors by at least 12X.

The next to last example displays a very different kind of dynamical behavior from the first four examples. The repressilator is capable of generating an oscillatory solution (Elowitz and Leibler, 2000). A related circuit can be found in Figure 6. The stiffness measure S varies cyclically from 775 to 1700.

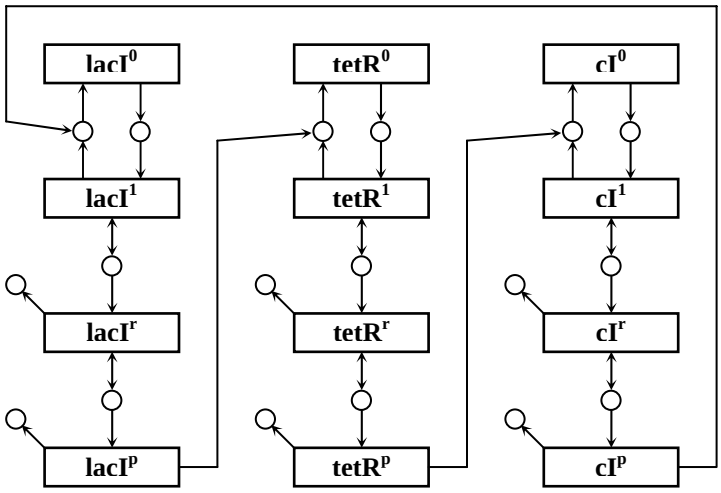

Figure 6. Biological Circuit for the Repressilator

Recently Elowitz and Leibler (2000) synthesized an analogue of this circuit, in which the *lacI* gene product represses the *tetR* gene, whose gene product in turn represses the *cI* gene, whose gene product closes the loop by repressing the *lacI* gene. This model system may provide insights into how biological clocks work (Lee *et al*., 2000). If we treat the full multiplicative mass balance kinetics of the gene activation and suppression and the mRNA transcription reactions in Figure 6 in a steady-state approximation (Segel, 1975) we obtain an effective Michael-Menten-type transcription kinetics similar to that of the Elowitz and Leibler (2000) (EL) model. An important difference between our kinetics model in Figure 6 (or its steady-state approximation) and the EL model is that the EL model exhibits undamped oscillatory behavior *only* if the gene suppression is cooperative, *i.e.* with a Hill coefficient $n \geq 2$. By contrast, our model in Figure 6 shows undamped oscillations even though it is *non-cooperative*, *i.e.* it has a Hill coefficient $n=1$ (Segel, 1975). The oscillatory behavior of the circuit in Figure 6, solving the full multiplicative mass balance kinetics without the steady state approximation, can be seen in Figure 7. For a relative error target of 0.01 on a PC with a Pentium III processor running Windows 98 execution by GEPASI takes 0.05 seconds, and execution by KINSOLVER (using the modified Euler Method) on the same computer takes 0.11 seconds. For a relative error target of 1E-6 GEPASI takes .15 seconds, and execution by KINSOLVER (using the adaptive RK-Fehlberg Method) on the same computer takes 0.66 seconds. Both simulators yielded identical results. An ensemble simulation of this model appears to call for a mixed strategy of lower and higher order methods.

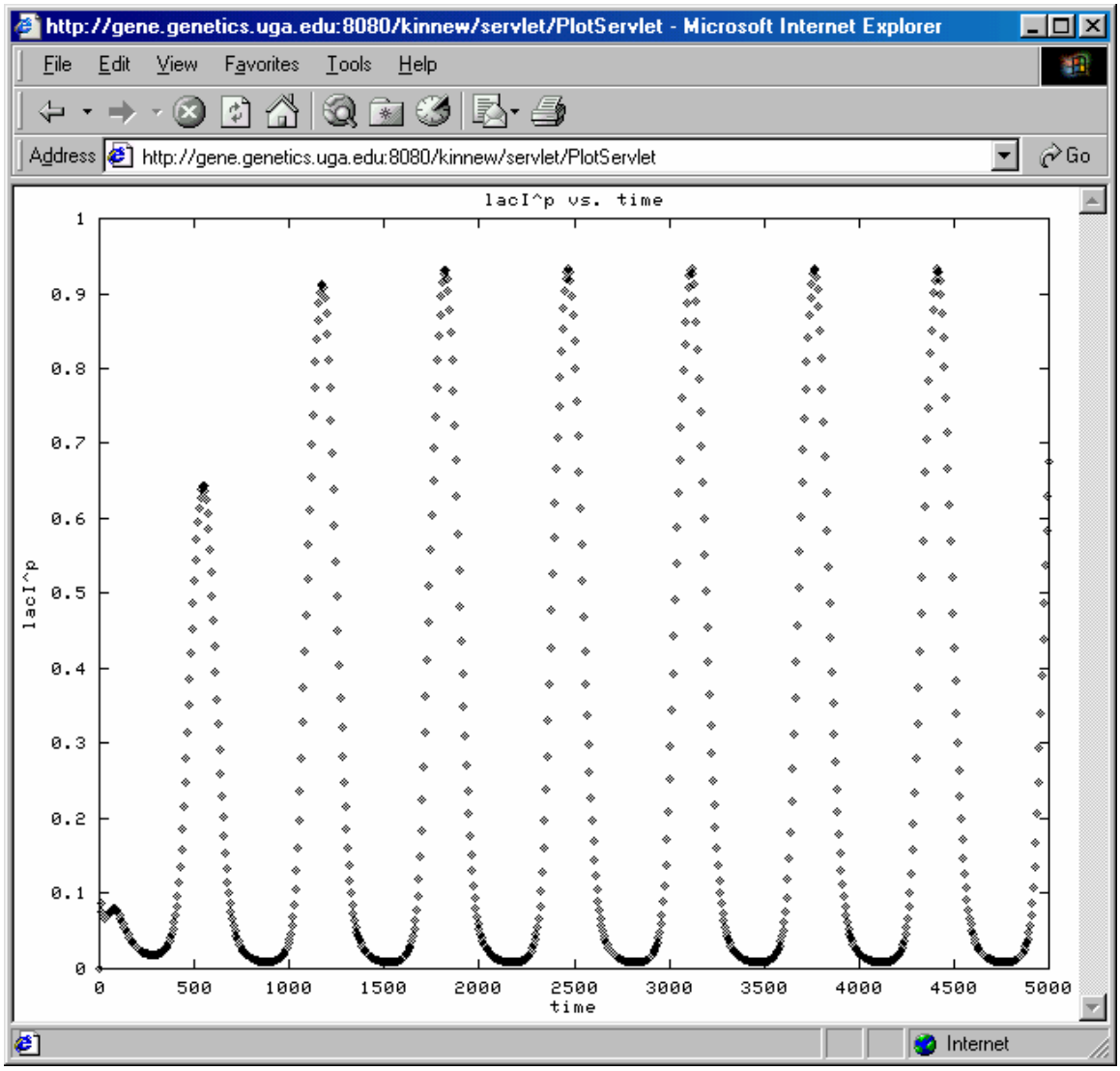

Figure 7. Workspace web page for Repressilator Model

Attempts to drop out the basal transcription reactions or the decay reactions in Figure 6 led to the removal of the oscillations. The parameter domain in the rate constants more favorable to oscillatory behavior apparently occurs when the decay rates between proteins and mRNAs are comparable, when there is a lag step at translation, and when some basal transcription is present.

Another famous oscillatory system is the Oregonator (Field *et al.*, 1972; Tyson, 1985; Murray, 1993) involving x = $[HbrO_2]$, y = $[Br^-]$, and z = $[Ce^{4+}]$ as chemical species. The Oregonator is one of the best studied oscillators both experimentally and theoretically. The oscillatory reaction network can be approximated by its dimensionless form (Murray, 1993):

$$\varepsilon\, dx/dt = qy - xy + x(1-x)$$
$$\delta\, dy/dt = -qy - xy + 2fz$$
$$dz/dt = x - z.$$

This reaction network can display stiffness because the first two reactions are empirically quite fast (ε and δ are small) while the last reaction empirically is quite slow. The stiffness measure S varies cyclically from 500 to 2600. This system was simulated in KINSOLVER. For a relative error target of 1E-6 the Adaptive Runge-Kutta method found the solution in 11.37 seconds in 24,870 iterations on a PC with a Pentium IV processor running Windows XP, and took 1.6 seconds in GEPASI using LSODA.

We have established in Table 1 that model family QA-Tr.B may be a candidate for ensemble simulation using a lower order method with a reasonable error tolerance of 0.01. To illustrate one of the advantages of KINSOLVER, we generated an ensemble of 12,250 models of the form QA-Tr.B with nearly equivalent fit by the method of maximum likelihood (Battogtokh *et al*., 2002), and each member of the ensemble has a nearly equivalent likelihood value relative to the data in Battogtokh *et al.* (2002). Finding this ensemble of 12,250 models involved executing the simulation engine KINSOLVER at least 40,000 times. This ensemble was projected into the plane in which the rate of transcription of *qa-1S* gene (k_1) is plotted on the x-axis while the rate of inactivation of the qa-$1F^p$ protein by the qa-$1S^p$ protein (k_2) is plotted on the y-axis in Figure 8 to examine properties of the ensemble. The kinetics solver gives us the capability to explore ensembles of models explaining data on a particular reaction network, although the display capability is not currently integrated into the Web interface.

While the ensemble approach (Alves and Savageau, 2000, Battogtokh *et al*., 2002) is somewhat similar in spirit to the scanning capability of GEPASI (Mendes, 1993), there is a crucial difference: scanning performs a *brute-force* exhaustive search on a fixed, user-predetermined grid, whereas the ensemble approach automatically finds and then systematically explores the "high-relevance" regions of the model parameter space, without prior user knowledge of the location or extent of such high-relevance regions (Battogtokh *et al*., 2002). Exhaustive scans on large grids in high-dimensional parameter spaces (of dimensions larger than 2 or 3, say) are computationally prohibitive, even for the simplest circuit models discussed here, since the computation time grows exponentially with $D \times log(L)$, where D is the model parameter space dimension and L is a typical linear grid dimension (*e.g.* the number of grid intervals along a particular grid axis). By contrast, in the ensemble approach, computational effort scales typically linearly with D and there is no predetermined grid size L. The ensemble approach involves a random walk in the parameter space guided by a figure of merit. As the random walk settles into a steady state, the ensemble is identified. This Markov Chain Monte Carlo Method (MCMC) is described in detail (Battogtokh *et al.,* 2002). Hence, high-dimensional model parameter spaces can be explored very efficiently.

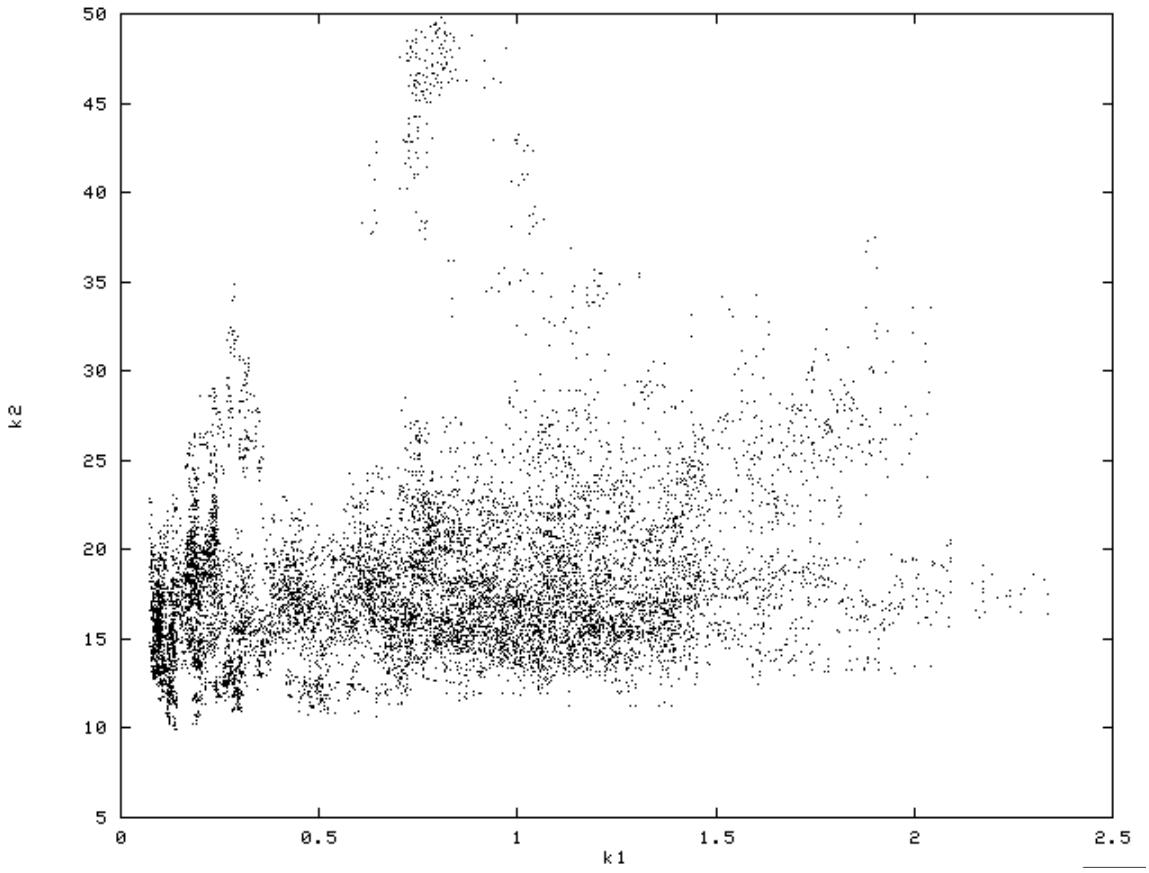

Figure 8. Scatter plot of an ensemble of 12,250 models in the (k_1, k_2) plane.

6. Discussion

This general purpose simulator of a model ensemble is one tool needed in carrying out the process of hypothesis-driven genomics, which begins with a formal model of the system, namely a chemical reaction network. A system like the *lac* operon or *qa* cluster is first perturbed genetically with a gene knockout, environmentally (*i.e.*, with changing carbon source), or chemically (*i.e.*, with a protein inhibitor). Then the system is observed as a whole with RNA or protein profiling to obtain the cellular state. The simulator is invoked at several succeeding stages in fitting, predicting, and evaluating the fit to refine (modify) the model (Figure 8).

This kinetics solver is designed to be integrated into a tool to fit a model to profiling data using a number of standard approaches like the method of maximum likelihood or method of least squares (Mendes and Kell, 1998; Battogtokh *et al*., 2002), although the code as currently described here is not so configured. The kinetics solver is invoked for each new fitted model tried and the solution, compared through a figure of merit with the observed profiles. Once an ensemble of models consistent with the data is identified, the solver is then used to predict how the system will respond to a perturbation. Inevitably in evaluating fit of models the hypothesized ensemble will be modified, and the simulator is setup to make it easy to revise existing models. Multiple runs of different models can be executed from one input file as may be required in model fitting. At the final stage the kinetics solver may be used to evaluate the fit of a particular model through some methodology like bootstrapping before trying new experiments (Efron, 1982). The kinetics solver is an essential task repeatedly invoked in the automated workflow summarized in Figure 8 for identifying a reaction network (Kochut *et al*., 2003).

Many of the fitting and evaluation procedures for fitting kinetics models will involve invoking the simulator each time that a new network is tried in the fitting or model evaluation process (Battogtokh *et al*., 2002). As a consequence there is a premium on finding a solution procedure that is quick and accurate enough! With a relative error in the solution of 0.01 the question arises about propagation of error in the solution. What we observed in Tables 1-4 were solutions graphically indistinguishable from the high precision solutions. Among the five solution methods examined, the LSODES method uniformly outperformed other methods for low and high relative error. On the other hand, the Adaptive RK-Fehlberg method guarantees a global error target for the solution, but does not perform as well as LSODES. There is thus an advantage of having several solution tools in the toolbox. The KINSOLVER package has a modular design like E-CELL (Tomita *et al*., 1999) and GEPASI (Mendes, 1993) so that as better solution procedures are identified they can be incorporated.

As an example of this process of hypothesis-driven genomics (see Figure 8), we present a formal model for the *qa* gene cluster (Giles *et al*., 1985). The model in Figure 1 behaves in a number of ways qualitatively as it should. For example, adding sucrose has the effect of shutting down the *qa* cluster in Figure 1 (results not shown), which is observed experimentally (Geever *et al*., 1989). The framework facilitates the incorporation of more elaborate reaction networks as needed. For example Covert *et al*. (2001) and Case *et al*. (1972) both hypothesize that carbon metabolism is coupled to aromatic amino acid biosynthesis, and the simulator allows us to make predictions about this enlarged reaction network.

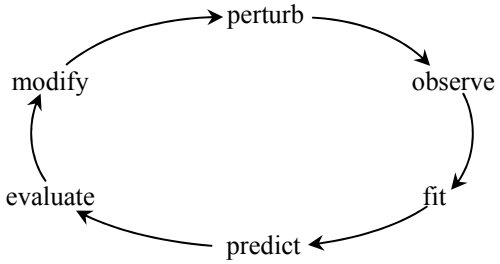

Figure 8. Hypothesis-driven Genomics

The *lac* operon model is concordant with experimental observations in a qualitative way as well. Shifting the carbon source to glucose shuts down the *lac* operon as expected, and increasing cAMP increases transcription through an internal signaling cascade serves to increase transcription of the *lac* operon (results not shown). A number of authors have constructed simplified models for the *lac* operon, which KINSOLVER can simulate without invoking steady state approximations leading to Michaelis-Menten kinetics.

The simulator has a number of limitations. First of all, the underlying model is deterministic, but Arkin, Ross, and McAdams (1998) have shown the importance of stochastic effects in one classic system, the phage λ switch (Ptashne, 1992). If the number of regulatory molecules is small in number, then phenotypic switching may take place between a lytic and nonlytic state for □. The saving grace of deterministic models is that under certain conditions these deterministic models have been shown to be limiting cases of stochastic models (Gillespie, 1977). Kierzek (2002) has recently developed new tools for simulating the underlying stochastic models when the data call for it.

The simulator KINSOLVER is based on a model in which the cell is hypothesized to be well-stirred. Weng *et al.* (1999) point out that consideration needs to be given to cellular compartments, protein scaffolding, and reaction channeling. Compartmentalization can be handled in part by the current simulator by indexing the species by the compartment containing them (Mendes and Kell, 2001). An example is quinic acid external to the cell (QA^e) and quinic acid in the cell (QA) in Figure 1. Similarly, scaffolding can be represented by allowing for additional concentration variables and corresponding reactions for a chemical species participating in a protein scaffold. This is not an elegant solution, but it does capture the idea that what a protein does may depend on where the protein is. Another option is the approach of E-CELL (Tomita *et al*., 1999), which introduces another table describing the compartmentalization of reaction species. Modeling spatial inhomogeneities of species in the cell is a challenging problem.

The formal model is based on collision dynamics determining the RHS of the coupled differential equations. So, the species concentrations enter multiplicatively on the RHS. Most of the well known kinetics formulations like Michaelis-Menten kinetics are derived as steady state approximations to such models (Segel, 1975). Whatever the functional form of the dynamics, the Weirstrass Theorem (Rudin, 1976) insures that we can approximate the true dynamics arbitrarily closely with such a functional form.

Currently, the layout of the input is such that the topology cannot be graphically manipulated. What is needed is an easier way to manipulate the topology of a reaction network, and this is a subject of current work (Becker and Rojas, 2001). In that the simulator KINSOLVER is setup to execute many reaction networks at once, more flexibility in the output interface is desirable to display ensembles of models (Alves and Savageau, 2000).

Even with these limitations to the interface, KINSOLVER like GEPASI is quite easy to use. Both systems have been employed for educational purposes quite successfully. For example, 16-17 freshmen utilized KINSOLVER in an introductory biology course with little difficulty in Springs 2002, 2003, and 2004. Unlike GEPASI input begins by entering the reactions from the screen, while KINSOLVER takes the specification of the reactions from a file, which is checked for formatting. While the information about a reaction network is summarized on 3 screens in GEPASI, KINSOLVER presents this information on one screen for modification in successive runs of the simulator (see Figure 4). While GEPASI has more flexibility in graphing, KINSOLVER ties the plots of trajectories over time directly to the input screen via buttons (see Figure 4). Both GEPASI and KINSOLVER use GNUPLOT to implement the graphics, providing ease of extension and portability.

In spite of these limitations, a number of limitations have been removed. The simulator is setup for a deterministic reaction network that satisfies mass balance. The simulator will solve reaction networks of arbitrary size, topology, rate constants, and initial conditions by 5 standard methods. As a consequence, the desired protein-DNA interactions, protein-protein interactions, regulation, and metabolic pathways can be included. An arbitrary number of reactants and products can participate multiplicatively in any particular reaction, enabling higher order kinetics. The simulator allows calculation not only of species concentrations, but also allows calculation of the concentration of different reactions over time. The solver

is flexible in method and enables the simulation of ensembles of models. In principle the simulator could be used to simulate the essential eukaryotic core once identified.

Acknowledgements

Students Jennifer Gillis, Elizabeth Hall, Lynley Holt, Tong Lee, and John Petrie participated actively in studying, testing, and refining the models "QA–A", "QA–Tr.A", "LAC–PTS". We are grateful for their contributions. We are also grateful to Dr. D. Battogtokh for comments that improved the manuscript. This work was supported by NSF grant DBI-0243754.

References

Alves, R. and Savageau, M.A., 2000. Systemic properties of ensembles of metabolic networks: application of graphical and statistical methods to simple unbranched pathways. *Bioinformatics* 16, 534-547

Arkin, A., Ross, J., and McAdams, H.H., 1998. Stochastic kinetic analysis of development pathway bifurcation in phage λ-infected *Escherichia coli* cells. *Genetics* 149, 1633-1648

Arnold, J., Schuttler, H.-B., Logan, D.A., Battogtokh, D., Griffith, J., Arpinar, I.B., Bhandarkar, S., Datta, S., Kochut, K.J., Kraemer, E., Miller, J.A. Sheth, A., Strobel, G., Taha, T. Aleman-Meza, B. Doss, J., Harris, L., and Nyong, A., 2004. Metabolomics in Chapter 22 of *Handbook of Industrial Mycology*. Marcel-Dekker, NY. pp. 597-633

Barshop, B.A., Wrenn, R.F., and Frieden, C., 1983. Analysis of numerical methods for computer simulation of kinetic processes: development of KINSIM-a flexible, portable system. *Anal. Biochem.* 130, 134-145

Battogtokh, D., D. K. Asch, M. E. Case, J. Arnold, and Schuttler, H.-B., 2002. An ensemblemethod for identifying regulatory circuits with special reference to the *qa* gene cluster of *Neurospora crassa*. PNAS USA 99, 16904-16909

Beadle, G. W. and Tatum, E.L., 1941 Genetic control of biochemical reactions in Neurospora. *Proc. Natl. Acad. Sci. USA* 27, 499-506

Becker, M. Y. and Rojas, I., 2001. A graph layout algorithm for drawing metabolic pathways. *Bioinformatics* 17, 461-467

Bhalla, U. S.; and Iyengar, I., 1999 Emergent Properties of Networks of Biological Signaling Pathways. *Science* 283, 381–387.

Case, M.E., Giles, N.H, and Doy, C.H., 1972 Genetical and biochemical evidence for further interrelationships between the polyaromatic synthetic and the quinate-shikimate catabolic pathways in *Neurospora crassa*. *Genetics* 71, 337-348

Cornish-Bowden, A. and Hofmeyer, J.H, 1991. MetaModel: a program for modling and control analysis of metabolic pathways on the IBM PC and compatibles. *Comput Applic. Biosci.* 7, 89-93

Covert, M.W., Schilling, C.H. and Palsson, B.O., 2001 Regulation of gene expression in flux balance models of metabolism. *J. Theor. Biol*. 213, 73-88

Dang, Q. and Frieden, C., 1997. New PC versions of the kinetic-simulation and fitting programs, KINSIM and FITSIM. *Trends Biochem Sci*. 22, 317

DeRisi, J.L., Iyer, V.R., and Brown, P.O., 1997. Exploring the metabolic and genetic control of gene expression on a genomic scale. *Science* 278, 680-686

Efron, B., 1982. *The Jackknife, the Bootstrap and other Resampling Plans.* SIAM Press, Philadelphia, PA

Ehlde, M. and Zacchi, G., 1995. MIST: a user-friendly metabolic simulator. *Comput. Applic. Biosci*. 11, 201-207

Eisenstat, S. C., Gursky, M. C., Schultz, M. H., and Sherman, A. H., 1977. Yale Sparse Matrix Package: II. The Nonsymmetric Codes. *Yale University Computer Science Dept. report* no. 114

Eisenstat, S. C., Gursky, M. C., Schultz, M. H., and Sherman, A. H., 1982. Yale Sparse Matrix Package: II. The Symmetric Codes. *Int. J. Num. Math. Eng.*, 18, 1145-1151

Elowitz, M. B. and Leibler, S., 2000. A synthetic oscillatory network of transcriptional regulators. *Nature* 403, 335-338

Fehlberg, E., 1969. *Classical fifth–, sixth–, seventh– and eighth order Runge–Kutta formulas with step size control*. *Computing* 4, pp. 93–106.

Field, R. J., Koros, E., and Noyes, R. M., 1972. Oscillations in chemical systems, Part 2. Thorough analysis of temporal oscillations in the bromate-cerium-malonic acid system J. Am. Chem. Soc. 94, 8649-8664

Friedman, N., Linial, M., Nachman, I. And Pe'er, D., 2000. Using Bayesian network to analyze expression data. *J.Comput. Biol.* 7, 601-620

Geever, R.F., Huiet L.; Baum J.A., Tyler, B.M. Patel, V. B, Rutledge B.J. Case, M. E.; and Giles, N.H., 1989. DNA Sequence, organization and regulation of the *qa* gene cluster of *Neurospora crassa. Journal of Molecular Biology* 207, 15–34.

"Gene". Computing Life Web Page, Genetics Department, University of Georgia. http://gene.genetics.uga.edu/stc/

Giles, N. H., Case, M. E., Baum, J., Geever, R, Huiet, L, Patel, V., and Tyler, B., 1985. Gene organization and regulation in the *qa* (Quinic Acid) gene cluster of *Neurospora crassa*. *Microbiological Reviews* 49, 338-358

Gillespie, D.T. 1977. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem*. 81, 2340-2361

Gygi, S.P. Rist, B. Gerber, S.A. Turecek, F. Gelb, M.H. and Aebersold, R., 1999. Quantitative analysis of complex protein mixtures using isotope-coded affinity tags. *Nature Biotechnology* 17, 994-999

Hairer, E.; Nørsett, S. P. and Wanner, G., 1987. *Solving Ordinary Differential Equations* I. Nonstiff Problems. Springer–Verlag.

Hindmarsh, A. C., 1980. LSODE and LSOD, Two New Initial Value Ordinary Differential Equation Solvers, *ACM SIGNUM Newsletter*, 15, No. 4, 10-11

Hindmarsh, A. C., 1983. ODEPACK, a Systematized collection of ODE Solvers, in Scientific Computing, R. S. Stepleman et al., Eds., 55-64

Huang, S., 1999. Gene expression profiling, genetic networks and cellular states: an integrating concept for tumorigenesis and drug discovery. *J. Mol. Med* **77**, 469-480

Jacob, F. and Monod, J., 1961. Genetic regulatory mechanisms in the synthesis of proteins. *J. Mol. Biol.* 3, 318-356

Johnston, M., 1987. A model fungal regulatory mechanism: the *GAL* genes of *Saccharomyces cerevisiae*. *Microbiolog. Rev.* 51, 458-476

Lee, K., Loros, J.J. and Dunlap, J.C., 2000. Interconnected feedback loops in the Neurospora circadian system. *Science* 289, 107-110

Kierzek, A. M., 2002. STOCKS: STOChastic kinetic simulations of biochemical systems with Gillespie algorithm. *Bioinformatics* 18, 470-481.

Kochut, K.J., Arnold, J., Sheth, A., Miller, J.A., Kraemer, E., Arpinar, I.B. and Cardoso, J., 2003. IntelliGEN: a distributed workflow system for discovering protein-protein interactions. *Parallel and Distributed Databases* 13, 43-72

Mendes, P., 1993 GEPASI: A software package for modelling the dynamics, steady states and control of biochemical and other systems. *Comput. Applic. Biosci.* 9, 563–571

Mendes, P., 1997 Biochemistry by numbers: simulation of biochemical pathways with Gepasi 3. *Trends Biochem. Sci.* 22, 361–363.

Mendes, P. and Kell, D.B., 1998. Non-linear optimization of biochemical pathways: applications to metabolic engineering and parameter estimation. *Bioinformatics* 14, 869-883

Mendes, P. and Kell, D.B., 2001. MEG (Model Extender for Gepasi): a program for the modeling of complex, heterogeneous cellular systems. *Bioinformatics* 17, 288-289

Murphy, K. and Mian, S., 1999. Modeling gene expression data using dynamic Bayesian networks. *Technical Report*. Berkeley.

Murray, J. D., 1993. *Mathematical Biology*. Springer-Verlag, NY

Petzold, L., 1983. Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. S*IAM J. Sci. Statist. Comput* 4, 136-148

Ptashne, M., 1992. *A Genetics Switch, Phage Lambda and Higher Organisms*. Cell and Blackwell Scientific Publications.

Purich, D. L.; and Allison R. D., 2000. *Handbook of Biochemical Kinetics*. Academic Press, NY, NY

Rice, R. G.; and Do, D. D., 1995. *Applied Mathematics and Modeling for Chemical Engineers*. John Wiley & Sons, Inc.

Rossbach P.; and Schreiber H., 2000. *Java Server and Servlets. Building Portable Web Applications*. Addison-Wesley.

Rudin, W., 1976. *Principles of Mathematical Analysis*. McGraw Hill, NY, NY

Salvadori, M. G.; and Schwarz, R. J., 1954. *Differential Equations in Engineering Problems*. Prentice–Hall.

Sauro, H.M., 1993. SCAMP: a general-purpose simulator and metabolic control analysis program. *Comput. Applic. Biosci*. 9, 441-450

Segel, I. H. , 1975. *Enzyme Kinetics*. Wiley, NY

Shmulevich, I., Yli Harja, O. and Astola, J., 2001. Inference on genetic regulatory networks under the best-fit extension paradigm. In *Proceedings of the IEEE-EURASIP Workshop on Nonlinear Signal and Image Processing (NSIP-01)*, June 3-6, Maryland, Baltimore.

Shmulevich, I, Dougherty, E.R., Kim, S., and Zhang, W., 2002. Probabilistic Boolean networks: a rule-based uncertainty model for gene regulatory networks. *Bioinformatics* 18, 261-274

Sveiczer, A., Csikasz-Nagy, A. Gyorffy, B. Tyson, JJ and Novak, B., 2000. Modeling the fission yeast cell cycle: quantized cycle times in *wee1- cdc25D* mutant cells. *Proc. Natl. Acad. Sci. USA* 97, 7865-7870

Tomita, M., Hashmoto, K, Takahasi, K., Shimizu, T.S., Matsuzaki, Y., Miyoshi, F., Saito, K., Tanida, S., Yugi, K., Venter, J.C., and Hutchison, C.A., 1999. E-CELL: software environment for whole-cell simulation. *Bioinformatics* 15, 72-84

Tyson, J. J., 1985. A quantitative account of oscillations, excitability, and travelling waves in a realistic model of the Belousov-Zhabotinskii Reaction. *In Oscillations and Traveling Waves in Chemical Systems.* Pp. 92-144. R. J. Field and M. Burger (eds.). Wiley, NY

Wagner, A., 2001. How to reconstruct a large genetic network from n gene perturbations in fewer than n^2 easy steps. *Bioinformatics* 17, 1183-1197

Weaver, D.C., Workman, C.T. and Stormo, G.D., 1999. Modeling regulatory networks with weight matrices. *Pac. Symp. Biocomput.* 4, 112-123

Weng, G.; Bhalla, U. S.; and Iyengar, R., 1999. Complexity in Biological Signaling Systems. *Science* 284, 92–96.

Yanofsky, C. and Kolter, R., 1982. Attenuation in amino acid biosynthesis operons. *Ann. Rev. Genetic* 16, 113-134

Yanofsky, C., 2001. Advancing our knowledge in biochemistry, genetics, and microbiology through studies of tryptophan metabolism. *Annu. Rev. Biochem.* 70, 1-37

Zwolak, J. W., Tyson, J. J., and Watson, L. T., 2001. Estimating rate constants in cell cycle models. A*dvanced Simulation Technologies Conference*, Seattle, WA