DEVELOPING A RATE EQUATION SIMULATION ENVIRONMENT USING MICROSOFT SILVERLIGHT

A Thesis

by

ADAM LANEY STEVENSON

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

December 2009

Major Subject: Biology

DEVELOPING A RATE EQUATION SIMULATION ENVIRONMENT USING MICROSOFT SILVERLIGHT

A Thesis

by

ADAM LANEY STEVENSON

Submitted to the Office of Graduate Studies of
Texas A&M University
in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE

Approved by:

Chair of Committee,	Bruce Riley
Committee Members,	Nancy Amato
	Christopher Menzel
	Greg Klein
Head of Department,	U.J. McMahan

December 2009

Major Subject: Biology

ABSTRACT

Developing a Rate Equation Simulation Environment Using Microsoft Silverlight.

(December 2009)

Adam Laney Stevenson, B.S., Texas A&M University

Chair of Advisory Committee: Dr. Bruce Riley

The exponential growth of information demands the automated movement of data and software via new software models that are able to integrate data and components on their own without scientists' direct involvement. However, current stand-alone software modeling environments do not support a secure software execution, nor do client server applications allow user customization of the software running on the servers. To address this problem, a biological pathway modeling environment was built as a stand-alone Rich Internet Application (RIA). The modeling environment was tested by constructing a simulation of the glycolysis pathways in the human erythrocytes, and the results were compared against one of the latest and richest erythrocyte metabolism models developed by Kuchel and Mulquiney. The working simulation was able to settle into a quasi-stable state, with substrate concentrations close to what Kuchel and Mulquiney presented. In later versions, it is hoped that the performance of the simulator can be increased and that it will become possible to link models together and add collaboration tools.

ACKNOWLEDGEMENTS

I would like to thank my committee chair, Dr. Bruce Riley, and my committee members, Mr. Greg Klein, Dr. Nancy Amato, and Dr. Christopher Menzel for their guidance and support throughout the course of this research.

Bruce has been great as committee chair. He has been both supportive and patient and has offered numerous insights throughout my graduate studies at Texas A&M, for which I am exceedingly grateful.

Greg has been there from day one supporting me on my research endeavors. His door has always been open, and he is always willing to answer my math questions--no matter how crazy they may be. He also provided critical guidance on how to implement mathematical algorithms in this thesis.

I am also very grateful for Nancy and Christopher for keeping me sane this summer by not letting me bite off more than I could chew; I do always try to push the limit, and I have truly appreciated their help in focusing the scope of this project. Without their help, it would not have been possible to finish in the time allotted.

I would also like to thank Dr. Rodolfo Aramayo for the many hours spent talking and discussing how to approach the merging of the field of biology with the field of computer science. Countless hours were spent between the two us debating concepts and whether the two fields are merging; and if so, in what ways. I am left with many fond memories of those experiences.

A special thanks also goes out to Tim Peterson, who helped shape this research during its early development. I always left his office with a book and many more

questions than answers. I only wish that it had been possible to have more discussions with him before he left Texas A&M.

Thanks also go to my friends and colleagues and the department faculty and staff for making my time at Texas A&M University a great experience. They have always made my days brighter, and I do not see how I could have finished without their unwavering support.

Finally, thanks to my mother and father for their encouragement and to my brother and sister for being only a phone call away. Their patience, understanding, and love throughout this study have sustained me. For them, I am truly blessed.

TABLE OF CONTENTS

Page

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

A rich Internet application (RIA) simulation environment needs to be developed to give scientists the necessary software flexibility and integration capability required to handle the exponential information growth before them. The magnitude and complexity of this information demand the availability and use of customizable automation tools that are capable of continuously turning this rising surge of information into a steady stream of predictive models and data summaries; anything less is outmoded.

In the recent past, data often just came in the form of papers, online databases, and distributed flat files. But recently, metabases and wikis have been added to the list and are being increasingly used to gather and distribute user-generated content. Examples include Molecular Biology Database Collection and EcoliWiki. These systems allow data inputters to be less constrained by data format and to share and collaborate in real time. It is not uncommon to find relevant information about an experiment or methods to debug a simulation, to get help with training or a software tool, to talk to sales staff, or to view blogs, social networking sites, forums, and even virtual worlds [1].

A good example of how growing information complexity is affecting scientists can be found by studying the evolution of software used to perform biological pathway simulations. When biological pathway simulations were first being built in the 1970s,

---

This thesis follows the style of *Biochemical Journal*.

speed and efficient use of memory were the primary constraints of the algorithms. Because the resources of the computers were so limited, often computer software models were limited to a few thousand lines of code. However, it was also possible for scientists to build their own simulation software, obviating the need to rely on existing technology. This ability allowed them to test new ideas without being constrained by design choices of others. Thus, issues related to software diversification in the past were easier to overcome due to the small size of the models. When desired, incorporating a feature of another program could be accomplished easily in a few hours because the amount of source code that needed to be added to the program was small. As technology improved, so also did the predictive power of the models and the complexity of the data analysis routines. These technological changes resulted in much larger software code bases and, thus, it became cost prohibitive to develop modeling environments from scratch each time a new modeling experiment needed to be performed. The result was that engineers bifurcated the modeling of software packages, separating the actual data models from the simulation routines so that investments in time and capital could become more reusable. Over time, often as a result of the need to maximize performance or to guard market share if the software was commercialized, these modeling programs grew and became increasingly specialized. As of 2006, there were twelve simulation environments that allowed scientists to simulate biological pathways with overlapping feature sets [2]. The downside to this increasing complexity is that building a competitive model often first requires the purchase, installation, and configuration of complex software packages like *Mathematica*, *Matlab*, or *Maple* to run

the packages [3]; alternatively, if scientists have enough time available, a modeling program can be developed from scratch, but even this possibility is discouraged [3]. Also, due to complexity, size of the libraries involved, and the fact that many of these programs are closed source, making large modifications to the algorithms involved in the simulation can be unwieldy or perhaps impossible, given typical time constraints. The result is that the diversification of the technologies has become a barrier to creating more complex, integrative models.

It is not uncommon to find models of the same pathway using different technologies to perform the simulation, which often results in different input and output formats. While there have been efforts to standardize the formats used, many programs either do not support the full standard or have bug-riddled implementations. Additionally, if a new feature is added to the program, the cost of implementing the feature in a standard fashion can cause the need for either the feature or the support for data standardization to be dropped.

An example of how increasing complexity simulates divergence can be observed by studying the evolution of the simulations dedicated to modeling the metabolism in erythrocytes. The model has been steadily evolving for over thirty-five years [3, 4, p. 64], with the first mathematical model of erythrocyte glycolysis being published in 1974 [5]. Since then, as the model has been tested and discrepancies found between the proposed model and the actual pathways, more parameters have been added. The models have expanded until erythrocyte glycolysis has become one of the most well-described and complex kinetic models published thus far. To date, it has been the topic

of at least five major papers [6-10], and two books [3, 11]. Notably, almost all of these simulations use a different set of software tools.

This problem of increased complexity will worsen as biologists build models that integrate multiple biological levels which are observable in different time scales [4, p. 671, 12, p. 9]. The increased complexities suggest the need to develop meta-models, a set of interconnected models of different biological pathways that can be used to more accurately mimic real biological systems.

To enable the creation of these larger models and meta-models, developers need the ability to share and interconnect models to form a self-updating information web. This self-updating web will prevent the increased quantity of available data from swamping modelers and also will allow the modeling environment to automatically re-run the simulations when values change. Such a system will enable the continuous integration and testing of models, which will greatly aide in the production of larger meta-models.

Enabling modeling systems to support the sharing and linking of parameters also will avoid the problem of data sources going offline. As of 2009, the two major repositories of kinetic modeling constants are the Enzymes and Metabolic Pathways Database (EMP) [13] and the BRENDA database [14]. Unfortunately, the EMP database, one of the oldest and richest sources of kinetic modeling data, went offline this past spring, leaving the BRENDA database as the only major source available today. That database is in financial jeopardy as well. This problem of losing data sources can be avoided by enabling the individual models and modeling environments to act as data

sources, instead of relying upon specific databases for information. *Facebook* is a good example of an application that is adopting this strategy, as it provides both an application environment to manage social connections and resources for other applications to share and distribute data. Establishing the same data redundancy grid between various kinetic simulators and other biological applications will help eliminate the problems associated with databases going offline and databases containing different sets of information [12]. Additionally, building large meta-models will require the ability to link together independent models. This will enable them to coordinate the sharing and updating of both model data and executable components, down to the individual compiled types, which contain the functions that execute the simulation. Due to security and performance concerns, current server-based collaboration suites cannot offer this level of customization. Desktop applications could do so if the application could run within a secure environment, as nobody wants a computer virus from an auto-updating model. Therefore, there is a need for modeling environments that can be coded as Rich Internet Applications (RIAs) and that can offer the customization of a desktop application, the collaborative features offered by server applications, and the necessary security to keep the application safe. RIAs can also allow the resulting application to be cross-platform. Three new competing technologies can be used to develop RIAs, of which *Silverlight* provides the best mathematical performance and allows programmers to use a multitude of different languages to target a single runtime, thus making algorithm comparisons simpler to perform.

Nevertheless, there are some downsides to using *Silverlight*. The platform is only in its third conception with the fourth about to be in beta. Thus, much functionality will need to be developed to allow for kinetic pathway modelers to build deterministic, ordinary differential equation (ODE)-based models. Furthermore, to enable the linking of models and the collaboration of modelers in peer-to-peer fashion, it will be necessary to develop a small backend system to facilitate information synchronization between two *Silverligh*t applications. Solutions to these problems are in reach and several prototypes of these technologies have already been developed by the author.

Thus, a *four-month study was performed to determine how to build a highly modular internet client application capable of simulating the human erythrocyte glycolysis pathway.* The initial capabilities of the simulation modeling environment were tested by building a model of the human erythrocyte glycolysis pathway and comparing the results to previous models of the same system. This process will allow scientists to observe the potential for scaling the resulting prototype into a full-blown modeling environment. Once the stand-alone version of the simulator is built, it will be possible then to propose the development of the multi-user version of the simulation environment and to add all the remaining collaboration features previously described. Furthermore, it is hoped that the development and testing of this prototypical simulation engine will encourage future extension of the model's attributes and, thus, enable the depiction of all the erythrocytes' pathways. The culmination will be one integrated model that can be shared and not constrained by the tools used or the websites accessed during its development. It is also hoped that this new model integration approach will

pave the way for scientists to build much more complicated models than can be built by current mathematical modeling programs.

The following summarize the major objectives of this project:

- Build a stand-alone pathway simulator that can be run from the web using C#, the .NET Framework, and *Microsoft Silverlight*;
- Test the framework by constructing a model of glycolysis pathways in the human erythrocytes and simulating it in the modeling environment;
- Analyze the data from the human glycolytic erythrocyte model by comparing it to known experimental data and the data produced by other human erythrocyte metabolism models.

2. BACKGROUND

This section presents an overview of glycolysis and then details what is generally known about the enzymes present in the model.

2. 1 GLYCOLYSIS

Glycolysis is a cellular process that extracts the energy from chemical bonds in glucose, a 6-carbon sugar, and transfers it temporarily to ATP (adenosine triphosphate) for later use by energy-requiring processes throughout the cell. This biochemical pathway exists in all cells on earth, and because of this, it is considered one of the oldest energy-producing systems in existence.

When glucose is readily available and ATP is in need, glucose enters the pathway from extracellular fluid at a rate of 2 μmol per ml per hour [11]. Once it enters, it undergoes ten biochemical reactions. The pathway initially requires the expenditure of two ATP molecules per glucose to prime the pathway for subsequent glucose oxidation. Each cycle of the pathway yields four ATP molecules, two NADH molecules (another energy-carrying molecule), two pyruvates, and inorganic phosphate. In most other cells that have mitochondria, pyruvate and NADH are fully oxidized to yield an additional 34 ATP; however, as erythrocytes lack mitochondria, the pyruvate is converted instead to lactate via the enzyme lactate dehydrogenase and is exported from the cell as a waste product.

Lactate production also converts NADH back to NAD^+, thereby facilitating the next cycle of glycolysis. Therefore, in erythrocytes, the net yield of the pathway is the production of two ATP molecules per glucose [11].

The overall rate of the pathway is primarily controlled by the enzymes hexokinase (HK), phosphofructokinase (PFK), and pyruvate kinase (PK), with PFK having the most influence. Pyruvate kinase acts at the end of the pathway and does not have much regulatory control. Furthermore, while the reaction with glucose and HK is the first one, it is not always a required step if the sugar source is fructose (fruit sugar) instead of glucose. Thus, this process leaves PFK as the only major flux-control point in the pathway. PFK is always necessary for the production of pyruvate and acts near the beginning of the pathway, where feedback regulation can optimally control overall energy flow. PFK activity is stimulated by high levels of AMP and ADP, which are prevalent in energy-depleted cells, and it is inhibited by the high levels of ATP to prevent over-utilization of glucose reserves [3, 15, 16].

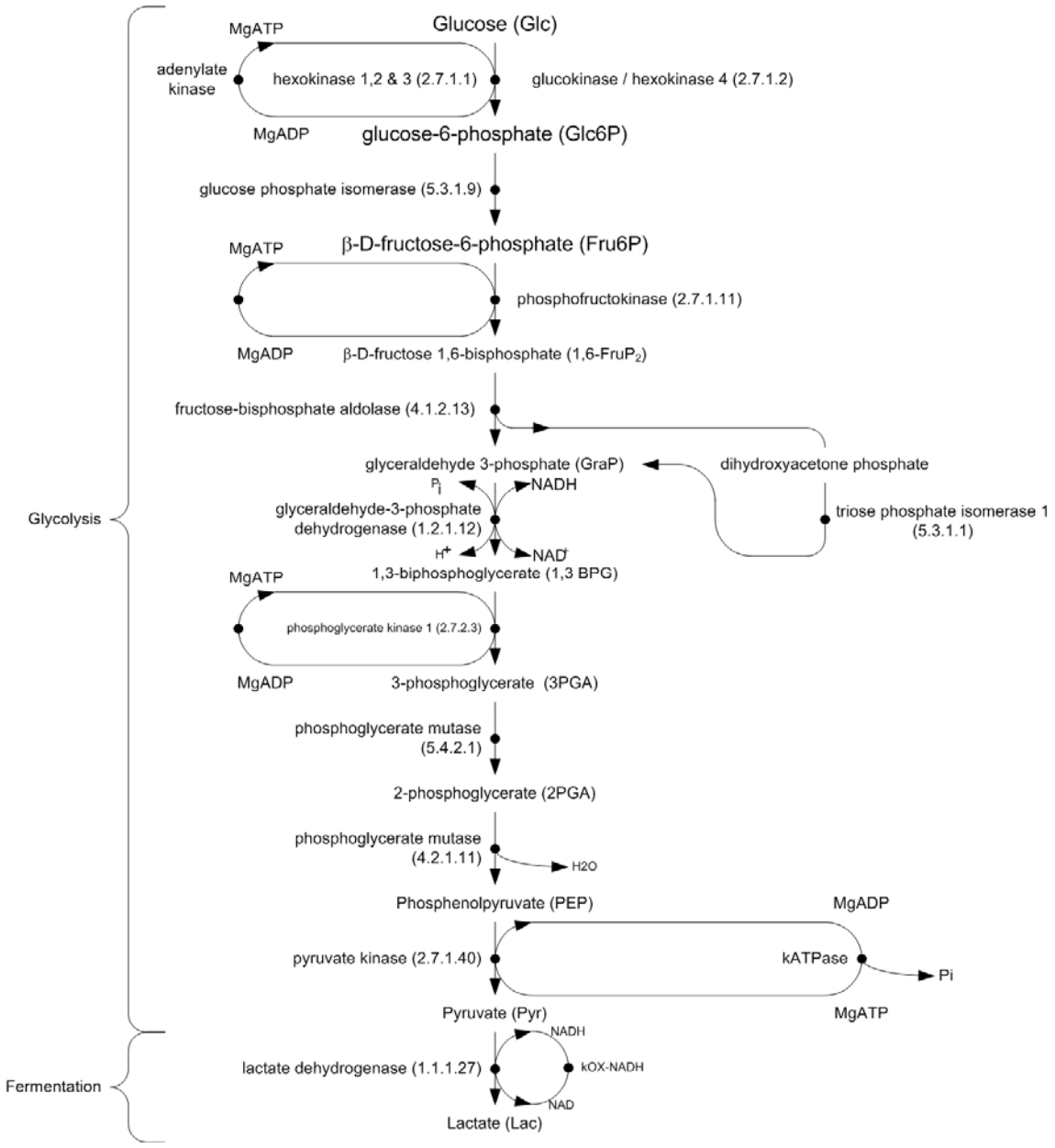


Figure 1 Glycolysis and the removal of pyruvate via fermentation [3, p. 176].

A diagram of the pathway with all substrates (reactants) and enzymes (proteins that accelerate each reaction) is shown in Figure 1.

2.2 SPECIFIC GLYCOLYSIS REACTIONS

The following summarizes all of the individual reactions that were modeled during the course of the study. All equations were based on Mulquiney and Kuchel's 2003 book. All table rate constants are from their 1999 paper.

Phosphorylation of Glucose via Hexokinase

The reaction of MgATP and glucose with hexokinase (HK) is the first reaction in glycolysis. Hexokinase requires a sugar and an ATP-Mg^{+2} to initiate the reaction [15, p. 431]. The sugar is usually D-glucose, but the process will also operate on D-fructose and D-mannose [16]. The $\Delta G'^{0}$ hexokinase is -16 kJ/mol, and it has a k_{eq} of 2000 [16]. The high $\Delta G'^{0}$ of the reaction is not due to the cleavage of the sugar, but to the hydrolysis of ATP into ADP^{3-}, P_i^{2} and H^{+}.

The activity of HK is regulated by the competitive inhibitors Glucose-6-Phosphate, 2,3-Bisphosphaogluycerate, Glucose-1,6-Bisphosphate, glutathione (GSH), and ATP [3, 15, p. 431]. When ATP is not bound to Mg^{+2}, it acts as a strong competitive inhibitor, as the negative charges of phosphate oxygen atoms are not shielded by the Mg^{+2} ion, given the concentrations are greater than 4mM [17, p. 1]. Therefore, the process results in the γ-phosphorus atom being less accessible for nucleophilic attack by C6-OH group of the sugar [15, p. 431].

Table 1 Michaelis-Menten constants for hexokinase

Reactant Name	Km	Ki	kCat
MgAtp	1.00E-03	n/a	180
Glucose	n/a	4.70E-05	n/a

Product Name	Km	Ki	kCat
MgAdp	1.00E-03	n/a	1.16
Glucose-6-Phosphate	n/a	4.70E-05	n/a

Table 2 Inhibitors for hexokinase

Inhibitor	Type	Binding Site	Ki
Glucose-6-Phosphate	Competitive	MgATP	1.00E-05
2,3-Bisphosphoglycerate	Competitive	MgATP	4.00E-03
Glucose-1,6-Bisphosphate	Competitive	MgATP	3.00E-05
Glutathione (GSH)	Competitive	MgATP	3.00E-03
ATP	Competative	MgATP	1.00E-04

A rate equation for the enzymatic reaction can be formed by modeling the enzyme after the b-bi-random model [18] and incorporating the following constraints. The rate constants for the equation are listed in Table 1 and Table 2.

The hydrogen ion concentration affects the rate at which hexokinase produces products. This dependency can be accounted for in the model by multiplying the K_{Cat} values by the bell equation, which is listed as Equation 2.1.

$$k_{pH} = \frac{k}{1 + \frac{10^{-pH}}{10^{-pk1}} + \frac{10^{-pk2}}{10^{-pH}}} \tag{2.1}$$

Multiplying the forward and reverse rate constants by the bell equation results in Equations 2.2 and 2.3.

$$kcat_f = \frac{180 \cdot 1.662}{1 + \left(\frac{10^{-pH}}{10^{-7.02}}\right) + \left(\frac{10^{-9.55}}{10^{-pH}}\right)} \tag{2.2}$$

$$kcat_r = \frac{1.16 \cdot 1.662}{1 + \left(\frac{10^{-pH}}{10^{-7.02}}\right) + \left(\frac{10^{-9.55}}{10^{-pH}}\right)} \tag{2.3}$$

Equation 2.3 describes the inhibitor effects of the rate equation.

$$I = \frac{Glc \cdot Glc6P}{K_{i,Glc} \cdot K_{i,Glc6P}} + \frac{Glc \cdot ATP}{K_{i,Glc} \cdot K_{i,Glc6P}} \tag{2.4}$$

Inserting Equation 2.3 into 2.4 forms the rate equation denominator.

$$d = 1 + \frac{MgAtp}{K_{i,MgAtp}} + \frac{Glc}{K_{i,Glc}} + \frac{MgAtp \cdot Glc}{K_{m,MgAtp} \cdot K_{i,Glc}} + \frac{MgAdp}{K_{i,MgAdp}} + \frac{Glc16P2}{K_{i,Glc16P2}} + \frac{MgAdp \cdot Glc16P2}{K_{m,MgAdp} \cdot K_{i,Glc16P2}} + I \tag{2.5}$$

Then, inserting 2.1, 2.2, and 2.3 into 2.5 forms the complete rate equation, 2.6.

$$v = vol_i \cdot \frac{E_0}{d}\left(\frac{kcat_f MgAtp \cdot Glc}{K_{m,MgAtp} \cdot K_{i,Glc}} + \frac{kcat_r MgAdp \cdot Glc16P2}{K_{m,MgAdp} \cdot K_{i,Glc16P2}}\right) \tag{2.6}$$

Note, that because glucose is being imported from the outside of the cell, an internal volume term has to be added to the equation (Mulquiney and Kuchel 2003, p. 180).

If solving for the equation by the mechanism alone, the reaction conforms to the random bi-bi mechanism. This mechanism is outlined below in Figure 2. The rate constants for the reaction follow in Table 3.

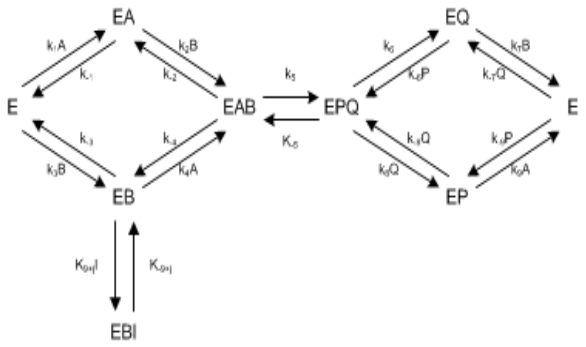


Figure 2 Random bi-bi mechanism with inhibition. This mechanism can be used to construct a rate equation for hexokinase in a mechanistic fashion. The inhibitor shown is actually four inhibitors, with each having its own k constant.

Table 3 Constants used to simulate HK mechanism

Constant		Value	Units	Constant		Value	Units
k1	k1	0.3	$M^{-1}*s^{-1}$	k-1	k2	3	s^{-1}
k2	k3	6.4	$M^{-1}*s^{-1}$	k-2	k4	3	s^{-1}
k3	k5	6.4	$M^{-1}*s^{-1}$	k-3	k6	3	s^{-1}
k4	k7	0.3	$M^{-1}*s^{-1}$	k-4	k8	3	s^{-1}
k5	k9	1.80E-02	s^{-1}	k-5	k10	1.36E-04	s^{-1}
k6	k11	3	s^{-1}	k-6	k12	0.3	$M^{-1}*s^{-1}$
k7	k13	3	s^{-1}	k-7	k14	6.4	$M^{-1}*s^{-1}$
k8	k15	3	s^{-1}	k-8	k16	6.4	$M^{-1}*s^{-1}$
k9	k17	3	s^{-1}	k-9	k18	0.3	$M^{-1}*s^{-1}$
k10	k19	7.1	$M^{-1}*s^{-1}$	k-10	k20	1	s^{-1}
k11	k21	0.075	$M^{-1}*s^{-1}$	k-11	k22	3	s^{-1}
k12	k23	4.5	$M^{-1}*s^{-1}$	k-12	k24	1	s^{-1}
k13	k25	0.1	$M^{-1}*s^{-1}$	k-13	k26	3	s^{-1}

Glucose Phosphate Isomerase (GPI)

The conversion of glucose-6-phosphate to fructose 6-phosphate via the use of glucose phosphate isomerase is the second preparatory step in glycolysis. During this step, the covalent bond between the first carbon atom and the oxygen is broken and then reformed, but with the second carbon atom receiving the electrons instead of the first. This action initiates the process of making the carbon chain symmetrical before cleavage, so that both halves of the chain can proceed to become a pyruvate molecule downstream. Unlike the previous step that is exergonic, this reaction is endergonic with a $\Delta G'^{o}$ of 2.2; however, this positive free energy change is easily overcome from the favorability of other reactions in the pathway, as this reaction's k_{eq} of 0.4, is very near equilibrium [16, p. 573].

This process is modeled using the standard reversible Michaelis-Menten model. Table 4 contains the constants for the standard model that can be inserted into Equations 1.7 and 1.8.

Table 4 Km and KCat constants used to build a model of GPI

Reactant Name	Km	Ki	kCat
Glucose-6-Phosphate	1.81E-04	n/a	1470
Product Name	**Km**	**Ki**	**kCat**
Fructose-6-Phosphate	7.10E-05	n/a	1760

$$d = 1 + \frac{Glc6P}{K_{m,Glc6P}} + \frac{Fru6P}{K_{m,Fru6P}} \tag{2.7}$$

$$v = \frac{E_0}{d} \cdot \left(\frac{k_{cat,f} \cdot Glc6P}{K_{m,Glc6P}} - \frac{k_{cat,r} \cdot Fru6P}{K_{m,Fru6P}} \right) \quad (2.8)$$

Phosphofructokinase (PFK-1)

The phosphorylation of fructose 6-phosphate to fructose 1,6-bisphosphate is the second priming reaction in the glycolysis pathway. In step one, a phosphate group was added to the sixth carbon; this reaction completes the symmetry by transferring a second phosphate group to the first fructose carbon from another MgATP complex. Under normal cellular conditions, products are strongly favored due to the enzyme having a k_{eq} equal to 1200 [3] with a $\Delta G'^{o}$ of -14.2 kJ/mol [16]. Table 5 lists the Km and kCat constants associated with PFK.

Table 5 Reactants and products of PFK

Reactant Name	Km	kCat
MgAtp	6.80E-05	822
Fructose 6-phosphate	7.50E-05	n/a

Product Name	Km	kCat
MgAdp	5.40E-04	36
Fructose 1,6-bisphosphate	5.00E-04	n/a

Phosphofructokinase is one of the primary regulatory enzymes in the pathway and can be strongly inhibited by high ATP, free Magnesium, and 2,3-bisphosphoglycerate. When ATP concentrations are above 1E-4M, ATP inhibits the formation of fructose 1-6-bisphosphate by binding strongly to an allosteric site. This

regulation helps prevent the pathway from burning excess amounts of glucose that could instead be stored for later catabolism using glycogenesis. The other negative feedback mechanism is the production of 2,3-bisphosphoglycerate from 1,3-bisphosphoglycerate, through the 2,3-bisphosphoglycerate shunt. This indirectly slows the production of ATP since 1,3-bishosphoglycerate is down stream of PFK. A table of the inhibitors has been provided below in Table 6.

Table 6 Associated Ki values of inhibitors regulating phosphofructokinase activity

Inhibitor	KT
Atp	1.00E-04
Mg	4.00E-03
B23PG	5.00E-03

The enzyme is also activated by high levels of AMP which increase as ATP is consumed. Phosphate and glucose-1,6-Bisphosphate are also known activators [3]. A list of PFK activators and their associated constants are listed in Table 7.

Table 7 Associated Ki values of activators regulating PFK

Activators	KR
Amp	3.00E-04
Phosphate	3.00E-02
Glucose-1,6-Bisphosphate	1.00E-02

PFK is an allosteric enzyme that can be modeled with two-state symmetry [19]. This is done by adding an extra equation, 1.9, to the Michaelis-Menten model. Unlike

most enzymes, allosteric enzymes take on different shapes based upon which substrates are currently bound. This chameleon-like nature is due to the fact that they are often composed of multiple identical subunits, rather than just one subunit. When bound to one subunit, a substrate can encourage or discourage other substrates from binding to the other identical units.

Examining Equation 1.9, the top terms are used to express the inhibitors and effects of pH on the enzyme, while the bottom terms represent both the substrates to be converted and the activators.

$$L = \frac{\left(\frac{10^{-pH}}{K_a}\right)^n \cdot \left(1+\frac{Atp}{K_{t,Atp}}\right)^4 \cdot \left(1+\frac{Mg}{K_{t,Mg}}\right)^4 \cdot \left(1+\frac{B23PG}{K_{t,B23PG}}\right)^4}{\left(1+\frac{Fru6P}{K_{m,Fru6P}}+\frac{Fru16P2}{K_{m,Fru16P2}}\right)^4 \cdot \left(1+\frac{Amp}{K_{r,Amp}}\right)^4 \cdot \left(1+\frac{Phos}{K_{r,Phos}}\right)^4 \cdot \left(1+\frac{Glc16P2}{K_{r,Glc16P2}}\right)^4} \tag{2.9}$$

With the allosteric properties defined, the equation can be applied to the existing 1.10 and 1.11 rate equations as if it were an additional inhibitor. Without the application of 1.9, the rate equation is modeled as a normal random bi-bi rate equation.

$$d = 1+\frac{Fru6P}{K_{m,Fru6P}}+\frac{MgAtp}{K_{m,MgAtp}}+\frac{MgAtp \cdot Fru6P}{K_{m,MgAtp} \cdot K_{m,Fru6P}}+ \frac{MgAdp}{K_{m,MgAdp}}+\frac{Fru16P2}{K_{m,Fruc16P2}}+\frac{MgAdp \cdot Fru16P2}{K_{m,MgAdp} \cdot K_{m,Fru16P2}} \tag{2.10}$$

$$v = \frac{E_0}{(1+L) \cdot d} \cdot \left(\frac{kcat_f \cdot MgAtp \cdot Fru6P}{K_{m,MgAtp} \cdot K_{m,Fru6P}}+\frac{kcat_r \cdot MgAdp \cdot Fru16P2}{K_{m,MgAdp} \cdot K_{m,Fru16P2}}\right) \tag{2.11}$$

Aldolase

The cleavage of fructose 1,6-bisphosphate into glyceraldehyde 3-phosphate and dihydroxyacetone phosphate is the fourth preparatory step of glycolysis. During this system, the sugar is cleaved in half, and an aldose, glyceraldehyde 3-phosphate, and a ketose, dihydroxyacetone phosphate, are formed. The aldose continues, while the ketose is converted to the aldose form in the next step of glycolysis. Using the constants in Table 8, the mechanism in Figure 3 can be entered into any system supporting the King-Altman algorithm.

Table 8 Quasi first order constants used to solve for the Aldolase rate equation. All of the constants correspond to rate constants for the mechanism depicted in Figure 3.

Constant		Value	Units
k1	k1	1.18E+07	$M^{-1}*s^{-1}$
k-1	k2	234	s^{-1}
k2	k3	995	s^{-1}
k-2	k4	6.50E+06	$M^{-1}*s^{-1}$
k3	k5	73	s^{-1}
k-3	k6	6.62E+06	$M^{-1}*s^{-1}$
k4	k7	1.00E+09	$M^{-1}*s^{-1}$
k-4	k8	1.50E+06	s^{-1}

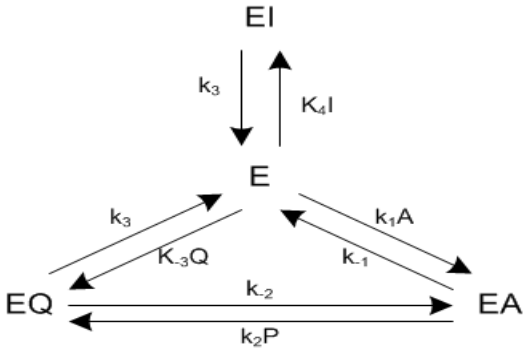


Figure 3 Example of an ordered uni-bi mechanism with inhibition. All of the rate constants correspond to values in Table 8.

The resulting rate equation is an ordered uni-bi rate equation. Equation 2.12 depicts the denominator of the equation. Equation 2.13 depicts the solved rate, with the associated constants referenced in Table 9.

$$d = 1 + \frac{B23PG + MgB23PG}{K_{i,B23PG}} + \frac{Fru16P2}{K_{m,Fru16P2}} + \frac{K_{m,GrnP} \cdot GraP}{K_{i,GrnP} \cdot K_{m,GraP}} \cdot \left(1 + \frac{B23PG + MgB23PG}{K_{i,B23PG}}\right) + \frac{GrnP}{K_{i,GrnP}} + \frac{K_{m,GrnP} \cdot Fru16P2 \cdot GraP}{K_{i,Fru16P2} \cdot K_{i,GrnP} \cdot K_{m,GraP}} + \frac{GraP \cdot GrnP}{K_{m,GraP} \cdot K_{i,GrnP}} \quad (2.12)$$

$$v = \frac{E_0}{d} \cdot \left(\frac{k_{cat,f} \cdot Fru16P2}{K_{m,Fru16P2}} - \frac{k_{cat,r} \cdot GrnP \cdot GraP}{K_{i,GrnP} \cdot K_{m,GraP}}\right) \quad (2.13)$$

Table 9 Solved Km, Kcat and Ki values for use in Eq. 2.12 and 2.13

Reactant Name	Km	kCat	Ki
Fructose 1,6-bisphosphate	7.10E-06	68	1.98E-05

Product Name	Km	kCat	Ki
Glyceraldehyde-3-Phosphate	1.89E-04	234	n/a
Dihydroxyacetone Phosphate	5.00E-04	n/a	1.10E-05

Triosphosphate Isomerase

Only glyceraldehyde-3-phosphate (GAP) can proceed through the pathway, and dihydroxyacetone phosphate must be converted to glyceraldehyde-3-phosphate. This conversion is performed by the enzyme trisphosphate isomerase, which converts the ketose to an aldose by breaking the double bond on the second carbon [15]. The reaction can be modeled by using the standard uni-uni reversible equation [18]. When this mechanism is solved, it results in Equations 2.14 and 2.15, with their respective constants listed in Table 10.

$$d = 1 + \frac{GraP}{K_{m,GraP}} + \frac{GrnP}{K_{m,GrnP}} \tag{2.14}$$

$$v = \frac{E_0}{d} \cdot \left(\frac{k_{cat,f} \cdot GraP}{K_{m,GraP}} - \frac{k_{cat,r} \cdot GrnP}{K_{m,GrnP}} \right) \tag{2.15}$$

Table 10 Solved Km and Kcat values for use in Eq. 2.14 and 2.15

Reactant Name	Km	kCat	Ki
Dihydroxyacetone Phosphate	446E=6	1280	n/a

Product Name	Km	kCat	Ki
Glyceraldehyde-3-Phosphate	1.62E-04	14560	n/a

Glyceraldehyde Phosphate Dehydrogenase

Glyceraldehyde phosphate dehydrogenase (GAPDH) oxidizes and phosphorylates glyceraldehyde-3-phosphate by NAD and P_i. This is the last preparatory step before the reaction starts producing ATP molecules [15]. The reaction can be modeled by modeling the mechanism and solving for the rate equation [18]. When this mechanism is solved, it results in Equations 2.16 and 2.17, with their respective mechanism constants listed in Table 11.

$$d = \frac{GraP}{K_{i,GraP}} \cdot \left(1 + \frac{GraP}{K_{id,GraP}}\right) + \left(\frac{B13PG}{K_{i,B13PG}}\right) \cdot \left(1 + \frac{GraP}{K_{id,GraP}}\right) + \frac{K_{m,B13PG} \cdot NADH}{K_{i,B13PG} \cdot K_{m,NADH}}$$
$$+ \frac{K_{m,GraP} \cdot NAD \cdot Phos}{K_{i,GraP} \cdot K_{m,NAD} \cdot K_{i,Phos}} + \frac{NAD \cdot GraP}{K_{i,NAD} \cdot K_{i,GraP}} + \frac{Phos \cdot GraP}{K_{i,Phos} \cdot K_{i,GraP}} \cdot \left(1 + \frac{GraP}{K_{id,GraP}}\right) +$$
$$\frac{NAD \cdot B13PG}{K_{i,NAD} \cdot K_{i,B13PG}} + \frac{K_{m,B13PG} \cdot Phos \cdot NADH}{K_{i,Phos} \cdot K_{i,B13PG} \cdot K_{m,NADH}} + \frac{GraP \cdot NADH}{K_{i,GraP} \cdot K_{i,NADH}} + \qquad (2.16)$$
$$\frac{B13PG \cdot NADH}{K_{i,B13PG} \cdot K_{m,NADH}} + \frac{NAD \cdot Phos \cdot GraP}{K_{m,NAD} \cdot K_{i,Phos} \cdot K_{i,GraP}} + \frac{K_{m,Grap} \cdot NAD \cdot Phos \cdot B13PG}{K_{i,GraP} \cdot K_{m,NAD} \cdot K_{i,Phos} \cdot K_{id,B13PG}} +$$
$$\frac{Phos \cdot GraP \cdot NADH}{K_{i,Phos} \cdot K_{i,GraP} \cdot K_{i,NADH}} + \frac{K_{m,B13PG} \cdot Phos \cdot B13PG \cdot NADH}{K_{i,B13PG} \cdot K_{m,NADH} \cdot K_{i,Phos} \cdot K_{id,B13PG}}$$

$$v = \frac{E_0}{d} \cdot \left(\frac{kcat_f \cdot GraP \cdot NAD \cdot Phos}{K_{i,GraP} \cdot K_{m,NAD} \cdot K_{i,Phos}} - \frac{kcat_r \cdot B13PG \cdot NADH}{K_{i,B13PG} \cdot K_{m,NADH}} \right) \quad (2.17)$$

Table 11 Quasi first order constants used to solve for the GAPDH rate equation. All of the constants correspond to rate constants for the mechanism depicted in Figure 5.

Simulation k	Reference k	Value	Units
k1	k1	5.16E+06	$M^{-1}s^{-1}$
k-1	k2	232	s^{-1}
k2	k3	1.00E+09	$M^{-1}s^{-1}$
k-2	k4	3.48E+06	s^{-1}
k3	k5	255	s^{-1}
k-3	k6	2.55E+08	$M^{-1}s^{-1}$
k4	k7	3.06E+06	$M^{-1}s^{-1}$
k-4	k8	6.50E+02	s^{-1}
k5	k9	2.55E+03	s^{-1}
k-5	k10	2.55E+08	$M^{-1}s^{-1}$
k6	k11	1.00E+09	$M^{-1}s^{-1}$
k-6	k12	3.10E+04	s^{-1}

Table 12 shows the Michaelis-Menten constants that are solved for by rearranging the equation for the King-Altman algorithm. Figure 4 outlines the mechanism.

Table 12 Solved Km, Kcat, and Ki values for use in Eq. 2.16 and 2.17

Reactant Name	**Km**	**kCat**	**Ki**	**Kid**
Glyceraldehyde-3-Phosphate	9.50E-05	232	1.59E-19	3.10E-05
NAD	4.50E-05	n/a	4.50E-05	n/a
Phosphate	3.16E-03	n/a	3.16E-03	n/a
Product Name	**Km**	**kCat**	**Ki**	**Kid**
1,3-Bisphosphhoglycerate	6.71E-07	171	1.52E-21	1.00E-06

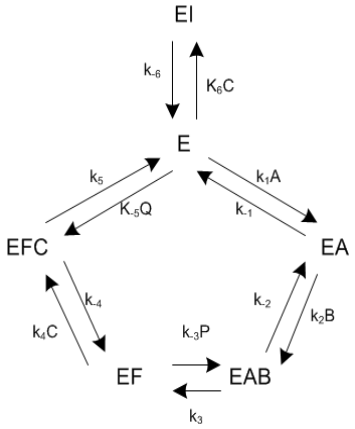


Figure 4 Diagram of GAPDH mechanism and the associated constants. The values for these constants can be found in Table 11.

Phosphoglycerate Kinase

Phosphoglycerate Kinase (PGK) transfers the phosphoryl group associated with the first carbon in 1,3-Bisphosphhoglycerate to ADP. This results in the formation of 3-phoshoglycerate and two ATP molecules per glucose molecule, which marks the first payoff step in the pathway [15]. The reaction can be modeled using a bi-bi rate equation [3, 18]. Solving this mechanism results in Equations 2.18 and 2.19, with their respective Michaelis-Menten constants listed in Table 13.

$$d = 1 + \frac{B13PG}{K_{m,B13PG}} + \frac{MgAdp}{K_{i,MgAdp}} + \frac{MgAdp \cdot B13PG}{K_{i,MgAdp} \cdot K_{m,B13PG}} + \frac{MgAtp}{K_{i,MgAtp}} + \frac{P3GA}{K_{m,P3GA}} + \frac{MgAtp \cdot P3GA}{K_{i,MgAtp} \cdot K_{m,P3GA}} \tag{2.18}$$

$$v = \frac{E_0}{d} \cdot \left(\frac{kcat_f \cdot B13PG \cdot MgAdp}{K_{m,B13PG} \cdot K_{i,MgAdp}} - \frac{kcat_r \cdot P3GA \cdot MgAtp}{K_{m,P3GA} \cdot K_{i,MgAtp}} \right) \tag{2.19}$$

Table 13 Solved Km and Kcat values for use in Eq. 2.18 and 2.19

Reactant Name	Km	kCat	Ki
MgAdp	1.00E-04	2290	8.00E-05
1,3-Bisphosphhoglycerate	2.00E-06	n/a	1.60E-06

Product Name	Km	kCat	Ki
MgAtp	1.00E-03	917	1.86E-04
Glyceraldehyde-3-Phosphate	1.10E-03	n/a	2.05E-04

Phosphoglycerate Mutase

Phosphoglycerate mutase shifts the phosphoryl group associated with the number three carbon in 3-phosphoglcerate to the number two carbon, forming 2-phosphoglycerate. This is a primary preparatory step, and the reaction is practically energetically neutral [15]. The reaction can be modeled by using the standard uni-uni reversible equation [18]. When this mechanism is solved, it results in Equations 2.20 and 2.21 with their respective constants listed in Table 14.

$$d = 1 + \frac{P3GA}{K_{m,P3GA}} + \frac{P2GA}{K_{m,P2GA}} \tag{2.20}$$

$$v = \frac{E_0}{d} \cdot \left(\frac{k_{cat,f} \cdot P3GA}{K_{m,P3GA}} - \frac{k_{cat,r} \cdot P2GA}{K_{m,P2GA}} \right) \tag{2.21}$$

Table 14 Solved Km and Kcat values for use in Eq. 2.20 and 2.21

Reactant Name	**Km**	**kCat**	**Ki**
Dihydroxyacetone Phosphate	446E=6	1280	n/a
Product Name	**Km**	**kCat**	**Ki**
Glyceraldehyde-3-Phosphate	1.62E-04	14560	n/a

Enolase

Enolase (ENO) dehydrates 2-phosphoglycerate to form phosphoenolpyruvate. This forms a double bond between the second and third carbons in phosphoenolpyruvate. [15]. The reaction can be modeled using a bi-bi rate equation [3, 18]. When this mechanism is solved, it results in Equations 2.22 and 2.23, with their respective Michaelis-Menten constants listed in Table 15.

$$d = 1 + \frac{P2GA}{K_{i,P2GA}} + \frac{Mg}{K_{i,Mg}} + \frac{P2GA \cdot Mg}{K_{i,Mg} \cdot K_{m,P2GA}} + \frac{PEP}{K_{i,PEP}} + \frac{Mg \cdot PEP}{K_{i,PEP} \cdot K_{m,Mg}} \tag{2.22}$$

$$v = \frac{E_0}{d} \cdot \left(\frac{kcat_f \cdot P2GA \cdot Mg}{K_{m,P2GA} \cdot K_{i,Mg}} - \frac{kcat_r \cdot PEP \cdot Mg}{K_{i,PEP} \cdot K_{m,Mg}} \right) \tag{2.23}$$

Table 15 Solved Km and Kcat values for use in Eq. 2.22 and 2.23

Reactant Name	Km	kCat	Ki
Glyceraldehyde-2-Phosphate	1.40E-04	190	1.40E-04
Mg	4.60E-05	n/a	4.60E-05

Product Name	Km	kCat	Ki
Phosphoenolpyruvate	1.11E-04	50	1.11E-04

Pyruvate Kinase

The last step in the glycolysis pathway is the conversion of phosphoenolpyruvate to pyruvate via pyruvate kinase. This step produces another two ATP molecules per glucose molecule that enters the system, which raises the total ATP produced to four and the net to two. During this reaction, the phosphoryl group is transferred from phosphoenolpyruvate to ADP; the double bond also is shifted from being between the third carbon to being between the second and the associated oxygen, once the phosphoryl group leaves the oxygen. Like PFK, this enzyme is allosteric in nature and can be modeled by multiplying the denominator by the term (1+L), with L defined in Equation 2.22. The rest follows a bi-bi-reaction mechanism, and can be described with Equations 2.23 and 2.24. The associated Michaelis-Menten constants are listed in Table 16.

$$L = \frac{\left(\frac{10^{-6.8}}{10^{-pH}}\right)\cdot\left(1+\frac{Atp}{K_{t,Atp}}\right)^4}{\left(1+\frac{PEP}{K_{R,PEP}}+\frac{Pyr}{K_{R,Pyr}}\right)^4\cdot\left(1+\frac{Fru16P2}{K_{R,Fru16P2}}+\frac{Glc16P2}{K_{R,Glc16P2}}\right)^4} \tag{2.22}$$

$$d = 1 + \frac{PEP}{K_{R,PEP}} + \frac{MgAdp}{K_{R,MgAdp}} + \frac{PEP \cdot MgAdp}{K_{R,PEP} \cdot K_{R,MgAdp}} + \frac{Pyr}{K_{R,Pyr}} + \frac{MgATP}{K_{R,MgATP}} + \frac{Pyr \cdot MgAtp}{K_{R,Pyr} \cdot K_{R,MgAtp}} \quad (2.23)$$

$$v = \frac{E_0}{(1+L) \cdot d} \cdot \left(\frac{kcat_f \cdot MgAdp \cdot PEP}{K_{R,MgAdp} \cdot K_{R,PEP}} - \frac{kcat_r \cdot MgAtp \cdot Pyr}{K_{R,MgAtp} \cdot K_{R,Pyr}} \right) \quad (2.24)$$

Table 16 Solved Km and Kcat values for use in Eq. 2.22, 2.23, and 2.24

Reactant Name	KR	kCat	KT
MgAdp	4.74E-04	1386	n/a
Phosphoenolpyruvate	2.25E-04	n/a	n/a
Atp	n/a	n/a	3.39E-03

Product Name	KR	kCat	Ki
MgAtp	3.00E-03	3.26	n/a
Pyruvate	2.00E-03	n/a	n/a

Lactate Dehydrogenase (LDH)

The removal of pyruvate via the conversion to lactate is not technically in the glycolysis pathway by definition, but without it, the NAD concentration cannot be regenerated and the pyruvate in the system cannot be adequately removed. Thus, it is required to have lactate dehydrogenase in the system. LDH can be modeled by random-ordered-ternary-complex mechanism, given rapid-equilibrium is assumed [20]. When this mechanism is solved, Equations 2.25 and 2.26 are generated. Their respective constants are listed in Table 17.

$$d = 1 + \frac{K_{m,NADH} \cdot Pyr}{K_{i,NADH} \cdot K_{m,Pyr}} + \frac{K_{m,NAD} \cdot Lac}{K_{m,Lac} \cdot K_{i,NAD}} \cdot \left(1 + \frac{Pyr}{K_{id,Pyr}}\right) + \frac{NADH}{K_{i,NADH}} + \frac{NAD}{K_{i,NAD}} + \frac{NADH \cdot Pyr}{K_{i,NADH} \cdot K_{m,Pyr}} + \frac{K_{m,NAD} \cdot NADH \cdot Lac}{K_{i,NADH} \cdot K_{m,Lac} \cdot K_{i,NAD}} + \frac{K_{m,NADH} \cdot Pyr \cdot NAD}{K_{i,NADH} \cdot K_{m,Pyr} \cdot K_{i,NAD}} + \frac{Lac \cdot NAD}{K_{m,Lac} \cdot K_{i,NAD}} + \frac{NADH \cdot Pyr \cdot Lac}{K_{i,NADH} \cdot K_{m,Pyr} \cdot K_{i,Lac}} + \frac{NAD \cdot Pyr \cdot Lac}{K_{i,NAD} \cdot K_{i,Pyr} \cdot K_{m,Lac}} \quad (2.25)$$

$$v = \frac{E_0}{d} \cdot \left(\frac{kcat_f \cdot Pyr \cdot NADH}{K_{i,NADH} \cdot K_{m,Pyr}} - \frac{kcat_r \cdot Lac \cdot NAD}{K_{i,NAD} \cdot K_{m,Lac}}\right) \quad (2.26)$$

Table 17 Solved Km, Ki, Kcat, and Kid values for use in Eq. 2.22, 2.23, and 2.24

Reactant Name	Km	kCat	Ki	Kid
Pyruvate	1.37E-04	458	2.28E-04	1.01E-04
NADH	8.44E-06	n/a	2.45E-06	n/a

Product Name	Km	kCat	Ki	Kid
Lactate	1.07E-03	115	7.33E-03	n/a
NAD	1.07E-04	n/a	-3.00E+00	n/a

Other Minor Supporting Reactions

There are a few other reactions that have to be included in the model. These include the production of MgATP, MgADP, Amp, the consumption of NADH, and the membrane transport reactions. Their full definitions can be found in Kuchel's book [3, p. 283 - 290]. Because these reactions do not follow the standard Michaelis-Menten pattern, these reactions are modeled by creating equation classes, instigating them and loading them into simulation.

2.3 HISTORY

The first paper presenting a mathematical model of glycolysis was published by Tom Rapoport, Reinhart Heinrich, *et al*. in 1974. The goal was to "…test whether their linear theory [would] suffice for a description of the steady state" and to determine whether a simpler understanding could be derived from glycolysis if the entire chain were connected together [21].

To test their linearization theory, they constructed a linear system of equations through a process of setting the velocities of two reactions in individual equilibrium equal to each other, solving for a particular substrate, and then substituting it back into another equation. By repeating this process, they were able to arrive at a single equation. This could be done because they did not consider the reverse reactions in the model. For instance, they assumed that the accumulation or removal of lactate would not affect the formation of glucose-6-phosphate or fructose-6-phosphate. By modeling glycolysis in erythrocytes, they were able to eliminate many of the subsequent pathways that are present in other cells but not present in erythrocytes, the most important being the citric acid cycle and oxidative phosphorylation. They also chose not to include any of the pentose phosphate pathways in the experiment, as they felt that at pH 7.2, they would contribute only about 10% of the output flux. The authors considered the reactions of hexokinase, phosphofructokinase, and pyruvate kinase to be irreversible, due to their high negative ΔG values and all others to be in equilibrium. The concentration levels of glucose were considered constant and saturating. Lactate was always considered to be removed from the cell. Last, they did include 2,3-

bisphosphoglycerate bypass in their system as an irreversible reaction and modeled the cycling of NAD and NADH.

The result of their work was the successful creation of a crude model through which they obtained values for Glucose—6-Phosphate, Pyruvate, and MgATP that roughly approximated the values that are known to exist today. But the many simplifications left much to be expanded upon by future models.

Rapoport and Heinrich followed up their 1974 paper twice in the next three years by adding to their model the synthesis and consumption of ATP [22-24] . They still modeled the reactions as forward-only processes and considered all the reactions involving hexokinase, phosphofructokinase, pyruvate kinase, bisphosphoglycerate mutase, bisphosphoglycerate phosphatase, and the ATP-consuming processes (labeled ATPase in the diagram) to be irreversible. Their model assumes the reactions involving aldolase, glyceraldehyde 3-phosphate dehydrogenase, triosephosphate isomerase and lactate dehydrogenase to be in equilibrium and those reactions are skipped in the model presented in Figure 5. This work was a step forward in the modeling of glycolysis because it explicitly modeled the use and creation of ATP.

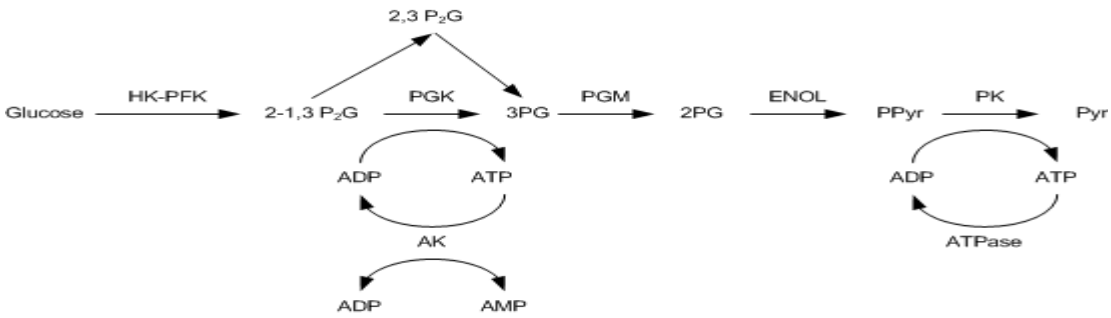


Figure 5 Rapoport and Heinrich's 1975 model of erythrocyte glycolysis.

In 1981, Ataullakhanov et al. expanded upon Rapoport and Heinrich's work by determining ATP is a strong inhibitor to phosphofructokinase and glucose-6-phospahte is an inhibitor to hexokinase. These insights were eluded by exposing erythrocytes to levorin, ouahain, and different sodium and potassium ion concentrations. These helped raise and lower the ATP concentrations and other metabolites in the cell without disrupting the glycolysis pathway. The result was they could measure the inhibitory effects [25, 26].

In 1989, Joshi and Palsson published one of the most comprehensive models [26]. The model included glycolysis, the phosphate pentose pathway, the adenosine metabolism, the NA/K pump, osmotic pressure, and membrane transport. While all of these components had been performed together by other models, they had not yet been all combined together into a single model.

Ten years later Mulquiney and Kuchel published an updated model and focused on including the 2,3-bisphosglycerate (2,3-BPG) shunt to the model. The shunt is an alternate pathway for glyceraldehyde-3-phosphate to follow to 3-phosphoglycerate and 2,3-BPG is used to build many secondary metabolites in erythrocytes; thus, its regulation was important to deduce. Previous models did not account for these regulatory elements, and papers that did discuss there inclusion did not account for the discrepancies between the in vivo and in vitro substrate concentrations related to its inclusion [9, 20, 27] . Four years later, the same authors wrote a book discussing how to model metabolism in erythrocytes, called *Modeling Metabolism with Mathematica* [3]. This text was very useful in understanding which equations needed to be included in the

model and outlined most of the methods necessary to produce the human erythrocyte model, even though further material needed to be found to actually produce working algorithms. This was because most of the methods described required the use of *Mathematica* and its library, which are not able to be integrated into *Silverlight* applications.

3. METHODOLOGY

The following outlines the input formats and major algorithms that were written to produce the simulation. An outline of all the equations generated by the King-Altman method is included in the Appendix.

3.1 INPUTTING MODEL INTO INPUT FORMAT

This section outlines how to enter the model into the XML data format that is read by the software.

Reaction Constituents

Reaction Constituents outlines the reaction molecules recognized by the model. This list includes names of substrates and enzymes that are present in the model.

Variables

In Variables, all the model level variables are entered as name-value pairs. Common variables are pH and external volume size. The first is used to specify the general pH of the environment, assuming the pH in the environment is buffered. The second is used to specify the volume of the external environment, which in this case would be the area around the cell that is able to pass constituents to the erythrocyte.

Mechanism Schematics

All the enzyme mechanism declarations that can be used to create a rate equation for a particular enzyme are included in Mechanism Schematics. Since a single mechanism declaration can be used for multiple enzymes, it is necessary to declare them

in a separate input section. Each mechanism is given a name that can be referenced by the enzyme declarations.

Enzymes

The enzyme declaration defines how an enzyme will be numerically described in the modeling environment if a reaction utilizes the enzyme. Describing an enzyme indicates that it is available for use—not that it necessarily will be used for a reaction. However, an enzyme has to be used by a reaction for the rate equation to be included in the simulation. Separating the reaction definitions from the enzyme definitions permits the same enzyme declaration to be used in more than one reaction or container.

The dynamics of an enzyme in this modeling environment can be described in two ways, depending upon the information available and the complexity of the enzyme mechanism that is being described. If the enzyme mechanism standard (uni-uni, bi-bi, etc.) and the KCat, kM, and kI values are known, then the enzyme can be defined by just specifying those values in nested declaration sections.

However, if the enzyme is using a mechanism-based method for describing its rate equation, then only the mechanism attribute in the enzyme declaration with the value set to the name of the mechanism is needed. Additionally, the enzyme declaration requires a mechanism constant mappings component nested within the enzyme declaration specifying the values for each of the constants in the mechanism.

Binding Sites Declarations

If a non-mechanism-based description is employed, a list of binding sites needs to be entered for each enzyme, primarily to specify where inhibitors and activators bind to the enzyme. With this information, it is possible to determine which substrate and inhibition constants need to be bound together when building expressions for competitive inhibitors.

Km Declarations

A list of k_M values must be supplied for enzymes using a non-mechanism-based description. Also, the primary reactant and product in the enzymatic reaction require defined k_M values.

Kcat Declarations

Enzymes using a non-mechanism-based description need a list of k_{Cat} values. The primary reactant and product in the reaction require defined k_{Cat} values, as they are used to determine the forward and reverse rate constants for enzymatic reactions.

Ki Declarations

A list of k_I values needs to be supplied for enzymes that are using a non-mechanism-based description. The primary and all secondary reactants and products in the enzymatic reaction need defined k_I values. These are used to build denominator terms as well as inhibitor and activator expressions.

Inhibition Groups

For each enzyme that is defined without specifying a mechanism, inhibition and activation groups can be defined. An inhibition group is a collection of inhibitors whose concentration can be considered a single concentration. An activation group acts in the same manner as an inhibition group and is implemented the same way. Often, their constants are the only differences between the two.

Inhibitors and Activators

An inhibitor or activator declaration is used to define the existence of a particular inhibitor/activator in an inhibition/activation group. The name of the inhibitor/activator must correspond to a reaction constituent declaration above.

Allosteric Declaration

In some cases, such as that involving phosphofructokinase, to properly describe the enzyme, information about the allosteric structure must be described. This description includes specifying the number of subunits, the inhibitors and activators, the standard pH value that was present when the inhibition and activation constants were derived, and whether increases in pH inhibit or activate the enzyme. All the above properties are attributes that can be set in the allosteric section, except for the inhibitor and activation information specified in the same manner as normal inhibitors and activators described earlier.

Containers

The model is first broken down into multiple simulation containers, each representing a bound region of space where molecules can interact. For the purpose of this model, there is just one container that needs to be simulated and that is the inside contents of the erythrocyte.

Reactions

The modeling environment supports two types of functions. The first is the ability to programmatically drive the reaction. In this situation, the user needs to supply the name of the class that can be instigated and that will act as a rate equation. The second, which is what most users will use, is the enabling of reactions by specifying an enzyme, reactants and products. If the second is being used, the user must specify the name of the enzyme and a list of products and reactants that will result from executing the reaction.

To aid debugging of the software, each of the reactions in the model can be disabled or enabled by setting the enabled attribute of each reaction declaration to *true* or *false*.

Initial Conditions

Before the simulation can be run, all of the reaction constituents must be assigned initial concentrations for each container. If a value is not assigned an initial concentration, then the value is assumed to be zero at the time the simulation starts, which can result in the simulation crashing.

Expected Values

At the end of each container, it is often useful to define a list of expected values that should result once the simulation is complete. This allows the system to compare obtained values with expected values and then to provide a percentage difference calculation in the results. This information is useful not only in comparing different models but also in debugging a model.

Entered Enzymes

The following outlines how the various enzymes were entered into the simulation, since it supports multiple methods.

Hexokinase was entered into the simulator by specifying its $k_{M,}$ $k_{I, and}$ k_{Cat} constants along with the reactants and products. It follows a normal bi-normal reaction scheme model with four possible inhibitors that can interact with the enzyme, depending upon what reaction constituents are present in the container.

Glucose phosphate isomerase is just a standard uni-uni reaction and was entered by specifying reactant, product, and relevant constants.

The enzyme phosphofructokinase was modeled the same way as hexokinase, by specifying the reactants and products involved in the reaction and k_{Cat} and k_M constants. In addition, the KT and KR constants describing its allosteric activators and inhibitors were provided. Effects of pH were taken into account by specifying the preferred pH for allosteric activation.

The aldolase enzyme does not use a standard uni-uni or standard bi-bi random reaction, and, thus, the standard modeling software could not support its creation using

k_M parameters and substrates alone. Instead, this enzyme was modeled by outlining its mechanism, supplying a list of rate constants for each of the individual steps in the mechanism, and solving for the rate equation using the King-Altman method.

The interconversion of dihydroxyacetone phosphate to glyceraldehyde 3-phosphate via triosephosphate isomerase is best inputted using the k_M values of the reaction, since the reaction is a standard uni-uni reaction.

Glyceraldehyde 3-phosphate dehydrogenase is modeled the same way as aldolase enzyme by specifying its mechanism. This procedure is again due to its mechanism being non-standard.

Phosphoglycerate kinase follows the standard bi-bi reaction scheme and, thus, was entered the same way as hexokinase was entered. Pyruvate and enolase were entered this way as well.

Phosphoglycerate mutase follows the standard uni-uni reaction scheme and was entered the same way as glucose phosphate.

Last, to remove excess pyruvate from the system, it is necessary to include a definition for the enzyme lactate dehydrogenase in the reaction scheme. This enzyme was modeled by entering its mechanism and constants, as its mechanism was not a standard bi-bi or uni-uni reaction.

Entered Reactions

A reaction for each of the above enzymes was entered into the reaction list. In addition, to the standard glycolysis reactions, it was necessary to include additional reactions used to properly maintain the balances of various energetic compounds in the

model. For instance, to regulate the amount of Mg-Atp and Mg-Adp in the system, simple first order kinetics reactions were coded and added to the system as custom rate equations. In addition, custom reactions were created to model the export of lactose and pyruvate, consumption of ATP by other extraneous processes in the cell, and the consumption of NADH. The equations of all of these supporting reactions are described in [3].

Initial Conditions

Table 18 lists the initial conditions that were used in the simulation.

Table 18 - Assigned initial conditions

Reaction Constituent	Concentration (M)		Reaction Constituent	Concentration (M)
Glucose	5.00E-03		Atp	2.10E-03
Glucose-6-Phosphate	4.00E-05		Adp	3.10E-04
Fructose-6-Phosphate	1.30E-05		Amp	3.00E-05
Fructose-1,6-Bisphosphate	2.70E-06		NAD	6.00E-05
Glyceraldehyde-3-Phosphate	5.70E-06		NADH	1.40E-07
Dihydroxyacetone Phosphate	1.90E-05		NADP	1.25E-07
1,3-Bisphosphoglycerate	7.00E-07		NADPH	6.40E-05
2,3-Bisphosphoglycerate	2.95E-03		Hexokinase	2.50E-08
3-Phosphoglycerate	6.40E-05		Glucose Phosphate Isomerase	2.18E-07
2-Phosphoglycerate	1.00E-05		Phosphofructokinase	1.10E-07
Phosphoenolpyruvate	2.30E-05		Aldolase	3.70E-07
Pyruvate	6.00E-05		Triose Phosphate Isomerase	1.14E-06
Lactate	1.40E-03		Glceraldehyde-3-Phosphate Dehy	7.66E-06
Glucose-1,6-Bisphosphate	1.22E-04		Phosphoglycerate Kinase	2.74E-06
Glutathione	3.20E-03		Phosphoglycerate Mutase	4.10E-07
Mg	3.00E-03		Enolase	2.20E-07
MgAtp	0.00E+00		Pyruvate Kinase	8.70E-08
MgAdp	0.00E+00		Pyruvate (External)	8.50E-05
MgB23PG	7.61E-04		Lactate (External)	1.82E-03
MgB13PG	0.00E+00		Phosphate (External)	1.92E-03
MgPhosphate	0.00E+00		MgFructose-1,6-Bisphosphate	0.00E+00

3.2 CREATING A DYNAMIC MODEL

With the simulation input specified, the simulation environment next parses the input file using a supplied XML reader class and then builds an object model. This object model is considered to be dynamic, due to the fact that information can be added or removed from the model without causing the model to become invalid. That ability is because the simulation does not assume anything about the definition state of the model and only acts to hold and collect information that could be useful in defining a particular instance of the model. The same information also can be used to run multiple concurrent models, since this is not the model that is used to actually perform the simulation.

While the current user interface permits only the text file to be imported and used to generate a static model directly from the dynamic model, the next version of the program will allow the user to change information in the dynamic model before generating a static model. Thus, while this separation is not a huge factor in this version of the software, it will be in future versions.

3.3 CREATING A STATIC MODEL

When a simulation is ready to run, a copy of the dynamic model is produced and a static model is formed. This static model represents an unchanging model and is necessary for quick execution and for helping to ensure that no information is missing before the model is simulated. Unlike the dynamic model, the static model is a ridged model that is *read only*, and all of the named references are resolved into actual pointers to objects in memory. As these objects are created, they are assigned references to their

schematics, and individual instances of the enzymes and constants involved in the reactions are assigned variables.

3.4 CREATING A SIMULATORY MODEL

Once the static model is created, the last step is to collect all of the information that is needed to actually create a model that can be simulated. This includes generating all of the rate equations from the static model, gathering a collection of variables that can be manipulated during the execution, and creating a set of equation variable mappings.

3.5 USING THE KING-ALTMAN METHOD TO GENERATE EQUATIONS

When the user specifies a mechanism to be used instead of a rate equation, the simulation environment must solve for the rate equation mathematically before being able to construct a working simulation. Therefore, the user must know all the elementary rate constants involved in the mechanism.

For an example of what text is being parsed, see Equations 2.12 and 2.13 with respect to the aldolase reaction. A user must enter an overall equation, a list of intermediates (if there are any), and a list of the mechanism steps. The following is an example of a valid input that, when parsed, generates the standard Michaelis-Menten equation $v=k_{cat}*S / (k_M+S)$.

```
[reaction]
      E + A ---> E + P
[mechanism]
      E + A <==> EA {1}
      EA ---> E + P {2}
[end]
```

For a more complicated example, the hexokinase reaction would be entered in the following manner:

```
[reaction]
      E + A + B <==> E + P + Q
[inhibitors]
      I
      J
      K
      L
[mechanism]
      E + A  <==> EA
      EA + B <==> EAB
       E + B <==> EB
      EB + A <==> EAB
         EAB <==> EPQ
         EPQ <==> EP + Q
          EP <==> E + P
         EPQ <==> EQ + P
          EQ <==> E + Q
      EB + I <==> EIB
      EB + J <==> EJB
      EB + K <==> EKB
      EB + L <==> ELB
[end]
```

The mechanism is a standard Bi-Bi reaction, meaning two inputs and two outputs, with four inhibitors that can react with the enzyme complex EB.

Once this text has been entered and identified in the XML file, the next step is for it to be scanned and tokens identified.

Scanner

The first thing that must be done to convert a text description of a mechanism to a working rate equation is to scan the text and identify tokens, or groupings, of text. As these tokens are identified, they are sent to the parser to determine the meaning of their sequence. For instance, the following line of text would be broken down into eight tokens, ignoring white space.

```
A + B <--> AB {1}
```

The above line says the substrates A and B react and form the substrate AB, that the reaction is a bidirectional reaction, and it should be assigned the k constants k_1 and k_{-1}. It also represents one step for a mechanism that would have N steps, where N is at least one.

Also for scanning, it was necessary to develop a line reading tool that looked for all of the possible input types reading the sequence one character at a time. These input types include the names of the substrates, the plus sign, the directional arrows, the constant number assignment in brackets that could exist at the end of each mechanism, and line breaks. When one of these patterns was identified, the scanner generated a token and passed it to the parser.

Parser

The parser's job is to determine the context of the text tokens. Recognizing the existence of a token does not mean the token is in the correct location or that it has a particular meaning in the current context. For instance, a plus sign has to follow a name of a substrate and cannot start a line.

As tokens are generated by the scanner, the parser uses them to generate object graph in memory representing the inputted text. This procedure includes determining the inputs and outputs of the overall equation, making a list of inhibitors and other modifiers, constructing a list of mechanism steps, and producing a set of rate constants for each mechanism. For all this to occur, a single parsing method is defined that takes in the current token, the last token, and assesses whether an equal sign has been seen yet

on the current line as arguments. Given no errors, the result is an abstract syntax tree that represents the inputted mechanism and that is ready to be converted into a rate equation.

Building the Rate Equation

With all of the information parsed, the next job is to actually build the rate equation. This is done in a series of nine steps. The basis of this procedure is documented [18, 28], but it leaves many details for the user to work out and clarify. This section tries to resolve those issues by outlining an implementation of the procedure.

The first step in constructing the rate equation is to build a table that will specify which constants and substrates are coming into an enzyme complex and toward which enzyme complex they are heading next. The size of the table, both length and width, is equal to the number of unique enzyme intermediates that exist, including the raw enzyme. The first step in constructing this table is to assign each of the enzyme substrates a row number. Next the actual array that will be used for the table is instigated in memory, and every element in the table is assigned its row and column number. This allows for the elements to be added to a list later and also for a later algorithm to figure out what table position they were initially assigned.

With the table built, it is now possible to go through the list of reaction mechanisms and assign each one of them to a slot. But before this is done, the system ensures that each mechanism has a pair of constants associated with it. It is possible that during the parsing that a constant was not assigned if one was not specifically

mentioned; and in that case, before this step is over, a constant pair is assigned to the mechanism.

The next step is for the system to create a list of enzyme complexes that are in the system and to separate them from the substrates and intermediates that enter and leave the system. This is managed by first identifying the substrate that is conserved in the overall reaction and then identifying all of the substrates that start with the same pattern of characters. If there is no conserved substrate, it assumes that the enzyme complexes in the mechanism list start with the capital letter E. Next, a search is done against the reactants and products of each reaction to identify those reactants and products in the enzyme complex. For each set of substrates, the enzyme complex is recorded. Then the mechanism is broken up into one or two reaction segments, one segment for each direction the mechanism goes. For each segment, the associated rate constants, substrates produced, and direction are assigned. This is an important step, as later this information will be used to determine if substrate cycles exist, which are eliminated from the numerical results. All mechanisms that produce products are marked as well. This procedure is used to determine which ones are candidates for producing the numerator expression.

The next step is to create an expression for each entry in the previously produced table. The expression is the sum of all of the reaction segment expressions, and each reaction segment expression is the product of all of the constants and substrates that occur at that particular segment. The result of this calculation is an adjacency matrix which is displayed in the programs output.

Next the determinate of this matrix is calculated, and a potential set of numerator candidate terms is produced. To calculate the determinate, one row is skipped at a time; then for each remaining row, a set of expressions is produced from each cell in the adjacency matrix, with one cell, and thus expression, from each row. The resulting set is multiplied together and this process continues until all possible combinations of expressions are created, given no row contributes more than one cell at a time. After the terms are multiplied, they are simplified using the computer algebra system, if possible. Not all of the previous resulting expressions are valid numerator and denominator terms. To determine which ones are valid, they must be analyzed to determine if any loops are present. If so, they need to be removed. (A loop is defined as two constants from the same mechanism appearing in the same term, thus creating a partial net cancellation of molecular flux in the mechanism.) The expressions are checked one at a time by isolating the constants and seeing if every constant is discoverable from every other constant in the expression through the use of the adjacency matrix.

Numerator terms can then be outputted. Which set of numerator terms is chosen to output does not matter, as long as expressions describe only one of the substrates, not all of them. For instance, if it is a bi-bi reaction and two outputs, P and Q, are produced, only one set of numerator terms should be produced for either P or Q. This is because the expressions describe the change in flux of the system, and the change in flux in P is the same as the change in flux in Q.

The denominator terms can also be outputted at this time, but there is no restriction on what terms should be outputted. Any term that did not have a cycle in it gets added to the denominator.

Before the final numerator and denominator are produced, like terms are collected and grouped together to produce significantly shorter expressions. This step, while not necessary, does result in faster simulation speeds.

3.6 DEVELOPMENT OF A COMPUTER ALGEBRA SYSTEM

It was necessary to create a small computer algebra system to express all of the dynamic expressions that are built by the rate equation builder and to perform necessary rearrangements and simplifications to produce results that can be simulated from the King-Altman Algorithm.

The first step in building the computer algebra system was to realize that the central abstract concept in the system needed to be the concept of an expression. An expression was recursively defined in this system as a collection of additional expressions and a single function that can be applied to the expression list. Operations that are currently supported include addition, subtraction, multiplication, division, and the power function.

The next step was the development of algorithms to simplify expressions and to group constants together. This procedure is necessary because the King-Altman algorithm can produce equations with thousands of terms in them, and most are repeated over and over again. If the equations were evaluated in this expanded format, the simulation environment would slow to a crawl and waste large amounts of memory. To

prevent this slowdown from occurring, the equations produced by the King-Altman algorithm must be passed to a simplification and grouping algorithm.

The computer algebra system also supports the ability to print individual expressions to the console or to the browser by applying html markup.

3.7 SIMULATING A MODEL

A copy of the simulation machinery is created in memory and initialized. This initialization includes specifying error thresholds, the amount of time the simulation should be simulated, and the minimum step size that should be used by the differential equation numeric solver.

Building the Equations

The simulation environment can build equations in three different ways, depending upon the complexity of the rate equation: through the specification of kM and kCat constants, from a specified mechanism, and from a type declaration.

From Km and Kcat Constants

The first way of building a rate equation is by building it one term at a time based upon the list of reactants, products, kM, kI, and kCat values, and by determining whether the rate equation is a standard uni-uni, bi-uni, or bi-bi reaction. Using the computer algebra system, the reaction information and the enzyme information are read and a numerator expression and a denominator expression are built for both the forward and reverse equations, resulting in a standard rate equation.

To calculate the numerator, the reactants or products concentrations are multiplied against each other along with the forward or reverse rate constants, and then the entire expression is divided by the relevant kM or kI terms for each of the products or reactants. The denominator is the numerator expression plus the individual reactants or products divided by the relevant kM or kI value.

If the equation has allosteric properties, the denominator of the rate equation is modified by multiplying the current denominator by (1 + L), where L is the allosteric modification equation defined earlier.

If the individual constants are pH specific, expressions for constants are created that take the pH into account, and then those expressions are incorporated into the equation as if they were constants.

If the equation operates across containers, the equation is multiplied by the internal volume.

From a Mechanism

The second way of building a rate equation is to solve for the rate equation using the King-Altman algorithm discussed in detail above. Given a closed reaction mechanism and a conserved enzyme, a rate equation can be produced and added to the list of equations to be solved numerically. After the static model is built, but before the simulation is run, the simulation environment checks for reactions which have enzymes with mechanisms. If any is found, the King-Altman algorithm is called, and a rate equation, along with a series of constants, is returned. Each of the returned constants is

matched up first with the mechanism constant mappings declared in the enzyme definition and then assigned a value.

From a Type

The last way of simulating equations is the universal catch all, which permits equations to be written directly in any .NET language, instigated, and then added to the equation list. This process allows any equation form to be added to the system, regardless of whether it is a simple rate equation or a complex formula that is pulling information from another website. To use this mechanism, it is necessary to set `componentType` attribute in the reaction declaration to an instantiable type. If the type is not part of the built in simulation software, but is available in the executing directory in an external assembly, it can be loaded by specifying the name of the assembly in the main configuration file.

Execution of the Adaptive Runge-Kutta Algorithm

The Runge-Kutta algorithm used for this model is defined in *Numerical Mathematics and Computing* in an abstract mathematical form [29]. It specifies how to solve a single equation using the Runge-Kutta algorithm with a fixed step size, a single equation with a variable step size with built in error detection, and a series of equations with a fixed step size. But it does not specify how to use the adaptive algorithm for multiple equations or how to adapt it for the use of a Computer Algebra System. Instead, it leaves the reader to combine the two algorithms. Since one of the themes of this thesis is saving scientists time and learning how to automate tasks, this section will explain

how adaptive step size Runge-Kutta algorithm was merged with the variable and expressions emanating from the computer algebra system.

A total of two Runge-Kutta-based algorithms were developed for this project. The first was a fixed step size algorithm. This was used in the initial testing phases of the project, but as the number of equations passed one and some volatile equations were produced, it was necessary to develop an algorithm that had an adaptive step size. Thus, the second algorithm was based upon the Runge-Kutta 5th order implementation with error checking. This one always permits the step size to vary during the course of the simulation, as long as the error bounds are not violated. If the error for any particular calculation moves above the higher bound, the step size will decrease; and if no upper bound is exceeded, but a lower bound is exceeded, the step size will grow larger. The rest of the description pertains to this latter algorithm.

The algorithm is divided into two phases, the first being the repetitive phase and the second being the calculation phase. When the Runge-Kutta algorithm is called, all of the following are passed in to be modified: a series of variables, a series of equations, a set of variable to equation mappings, the index of the variable used to specify the time (if there is one) an estimated starting step size, the maximum allowed error per calculation, the minimum error per calculation, and a data emitter used to output statistics. The algorithm starts by initializing variables and then segues into the time loop that will determine when the algorithm ends, which occurs when the simulation time equals the target time. The next step is to evaluate each of the passed equations with the current step size and variable values. This is done by calling the second evaluation function,

which will be described shortly. The results of the evaluation call are a change amount and an error. The next line checks to see if any of the error boundaries were violated during the process. An error boundary can be violated if the error returned is not a number and if it exceeds the error maximum or the error minimum. In the first two cases, the loop is aborted and the step size is reduced. The reason why the 'not a number' result is treated as a maximum error violation is that if the step size is too big, it can cause any term with a power quickly hit infinity during the five-step evaluation phase. On the other hand, if no aborts occurred, and a minimum error violation occurred, the step size will increase in subsequent steps to speed the execution along. Last, if no errors occurred, or if only minimum error violations occurred, after all the functions are evaluated, the changes in each function are mapped to the current variables based upon the stoichiometry of the system. This stoichiometry information comes in the form of the variable mappings that were passed as an argument to the function.

The calculation phase of the Runge-Kutta algorithm is relatively straight forward and follows what is given in the book *Numerical Mathematics and Computing* [29]. The calculation phase takes in a list of variables and in the equation that is being simulated. All six steps of the Runge-Kutta algorithm are calculated as outlined in the literature, and the error is calculated by subtracting the fourth order answer from the fifth order.

Outputting Results

The system outputs a series of standard HTML reports to a browser after each run of the program. These include final concentrations for each substrate, and the step-by-step production of rate equations which were inputted using rate equations.

These reports are fairly easy to generate, with the result being a generated HTML file, but to display the report inside the *Silverlight* environment is a problem. This is because *Silverlight* does not support the rendering of HTML within a *Silverlight* control; instead it must send back to the browser. The *Silverlight* environment does support the ability of the program to send information to the browser through a JavaScript bridge. Using this bridge, it is possible for the C# code executing on the client to send a segment of html to the browser.

This works out rather well, given the report never needs to be printed. But if the data does need to be printed, then the presence of any *Silverlight* UI on the screen control can cause gaps in the report to appear, even if the report is in front of the *Silverlight* control on the z-axis; therefore, even though the UI should be hidden, it still remains visible when printed. This printing problem can be resolved and the user can gain a better user interface experience by sending the results of a simulation to another tab. Allowing this resolution required the development of an inter-window communication protocol, which allows for very large amounts of html to be sent between tabs.

Using the third version of *Silverlight*, it is possible to send up to 10,000 characters of text between two instances of *Silverlight* at a time. To enable reports to

show up, the system is set up so that when it first starts, it checks to see if another instance of the program is running. If so, it boots up as a slave to the primary software application; otherwise it becomes the primary application. If it does boot up as a slave system, it checks to see if the current system has any tasks that need to be performed, including the display of a report. This communication is accomplished by the development of an inter-*Silverlight* communication protocol. A communication protocol was required so that multiple tabs can communicate to the primary window, the primary window would know where the messages were coming from, and responses could be sent to the correct window. In addition, because only 10,000 characters of text could be sent at a time and some reports are easily over 20,000 characters in length when HTML markup is included in the count, a report has to be sent to another tab in parts and then reassembled. To facilitate larger reports, a multi-part messaging protocol was developed. If the slave system does receive a multi-part message instructing it to display just a report, the slave system closes its *Silverlight* user interface and only displays the HTML on screen.

Assumptions

This model uses the same set of assumptions that was by Mulquiney and Kuchel in their 2003 book. This allows for the simulation results from the new simulation technology to be comparable with their past model. Of these assumptions, there are few worth repeating here.

The glucose concentration was kept constant at 0.005M. This can be done because glucose freely diffuses through the membrane and its concentration in the blood

stream is buffered. It is also assumed that the levels of pyruvate, lactate, and inorganic phosphate are buffered in the blood stream and any output from the cell to the blood stream did not change these levels. The pH of the system was held to 7.2. Substrates that were important in the regulation of the system but which could not be produced by the system of equations as a product in this model were set to their expected values. These included glucose 1-6-bisphosphate, glutathione, and 2,3-bisphosphoglycerate. None of the reactions involving the pentose phosphate pathway, the 2,3-bisphosphoglycerate, haemoglobin-metabolite binding, were implemented. Two magnesium-metabolite binding reactions were implemented to get the hexokinase reaction working, along with the associated assumption that MgATP is broken down by other secondary processes. These secondary processes were not individually modeled in Mulquiney and Kuchel, and an as such were grouped into as single equation in this model as well. The conversion of MgADP to MgATP and AMP via adenylate kinase was added to produce AMP that could be an activator for PFK. Without these processes, the ATP levels in the model can become unstable [25].

4. RESULTS AND DISCUSSION

4. 1 SIMULATION RESULTS

The model was simulated for a total of twenty minutes using the previously stated Runge-Kutta algorithm, and no individual calculation was allowed to have an error greater than 1E-9. During this time, the average fluxes per second for all of the substrates converged toward zero; thus, the system was able to enter a quasi-steady state. Table 19 lists the final steady state concentrations that were not held constant in the simulation.

Table 19 Final steady state concentrations of all substrates not held constant

Fructose-6-Phosphate	1.40E-05	1.21E-05	15.09%
Fructose-1,6-Bisphosphate	8.36E-07	2.23E-06	62.46%
Glyceraldehyde-3-Phosphate	7.49E-07	5.14E-06	85.44%
Dihydroxyacetone Phosphate	3.18E-06	2.14E-05	85.10%
1,3-Bisphosphoglycerate	1.15E-06	3.57E-07	222.27%
3-Phosphoglycerate	1.19E-04	7.09E-05	68.28%
2-Phosphoglycerate	1.99E-05	1.19E-05	68.29%
Phosphoenolpyruvate	3.77E-05	1.98E-05	90.07%
Pyruvate	5.68E-05	5.86E-05	3.08%
Lactate	2.10E-03	1.41E-03	49.54%
Mg	7.26E-04	3.83E-04	89.36%
MgAtp	2.20E-03	1.53E-03	44.34%
MgAdp	7.20E-05	9.56E-05	24.65%
Atp	1.17E-04	1.53E-04	23.78%
Adp	4.31E-05	1.08E-04	60.21%

There are discrepancies between the results presented here and the results published by Kuchel and Mulquiney, but the concentrations are close, and the

discrepancies are understandable. Kuchel and Mulquiney implemented the entire erythrocyte metabolism, whereas this model implements only glycolysis. Therefore, exact alignment of the concentrations was not expected. The largest value difference occurs with 1-3-bisphosphoglyeracate and can be explained by the fact that the pathway is supposed to split in the full model at 1-3-bisphosphoglyeracate to produce 2-3-bisphosphoglyeracate . If the 2-3-bisphosphoglyeracate shunt had been implemented, over half of the 1-3-bisphosphoglyeracate concentration would have been converted into 2-3-bisphosphoglyeracate and other secondary downstream derivatives, based upon the equations and steady state levels presented in Kuchel and Mulquiney model.

The higher than expected amount of 2-3-bisphosphoglyeracate explains why the immediate upstream substrate concentrations of fructose-1,6-bisphosphate, glyceraldehyde-3-phosphate, and dihydroxyacetone phosphate are lower, as they are experiencing a net negative feedback. Continuing this reasoning, glucose-6-phosphate and fructose-6-phosphate are higher because they are experiencing a net positive feedback from the high levels of fructose-1,6-bisphosphate. Additionally, since glucose-6-phosphate and fructose-6-phosphate are supposed to feed into the pentose phosphate pathway (PPP), which is expected to consume ~10% of the arriving glucose [21], it makes sense that these substrates are off by approximately the same amount. Last, the higher upstream levels of 1,3-bisphosphoglycerate will cause the higher levels of 3-phosphoglycerate, 2-phosphoglycerate, and phosphoenolpyruvate downstream. The levels of MgATP and other energy carriers are off also, but these differences were anticipated because a portion of these substrates are expected to be consumed in other

pathways if the full model was implemented. Because MgATP is a product of glycolysis and the upstream products are higher, the increased levels of MgATP are expected as well. The high level of NADH is also warranted, given the high level of 1,3-Bisphosphoglycerate; both NADH and 1,3-bisphosphoglycerate are products of glyceraldehyde phosphate dehydrogenase. The remainder of the energy metabolites are consumed in the production of MgATP and NADH; thus, their lower levels are expected, given the higher levels of MgATP and NADH in the model.

The overall substrate flux of the model appears to be correct since the error associated with pyruvate concentration is only 3%, and because pyruvate is one of the three substrate exit points in the system. The other two are phosphate and lactate. Phosphate levels are slightly elevated, but so is the amount of MgATP. The lactate concentrations are higher, at 49.54%, but higher concentration levels of this final substrate in the pathway are appropriate, given the higher substrate concentration levels upstream and also given that the speed of lactate export is about ten times less than the speed of pyruvate at pH 7.2. Thus, between pyruvate and lactate, it is expected that lactate levels will build up first.

To determine whether the system was evolving into a quasi-steady state, the simulation was allowed to run for 14 hours, and average net flux of each substrate was gathered over the course of the simulation. These averages are depicted graphically in an included appendix and visibly demonstrate that the fluxes of the substrates are converging toward zero over time. The convergence was anticipated, as the equations

used in this system are the same ones used in Kuchel and Mulquiney's model, and those equations converged as well.

4.2 PERFORMANCE

The performance of the simulation is currently much slower than other biological pathway simulators. It is currently taking 14 hours to simulate 20 minutes. This performance level was expected due to the un-optimized object-oriented nature of the system and due to the slowdowns associated with executing code within the browser using the *Silverlight* dynamic runtime. During early performance tests, it was found that code executing within the browser ran about ten times slower than what would be expected if the same code were run as a console application. Thus, one of the first improvements planned for later development is a reduction in the number of objects being referenced in calculations as well as the number of steps in the simulation without drastically increasing the amount of error in results.

4.3 NEXT STEPS

The next step in the development of this modeling engine will be the addition of a more extensive user interface and peer-to-peer functionality. The current user interface is rather minimal, as most of the input is done by entering information into a XML text document. This procedure suffices for the first version, since the objectives were focused around the development of the simulation engine and not ease of input. These user interface additions will also allow the user to manage inputs arriving from additional sources through the application.

After modification of the user interface, the next goal is the addition of extensive peer-to-peer collaboration functionality to the simulation environment. That upgrade should be a minor undertaking because the system was built using *Silverlight* and runs within the browser. Many of the current Web 2.0 technologies are designed to interface with the browser and web applications.

5. SUMMARY AND CONCLUSIONS

The purpose of this study was to build a biological pathway simulator in *Silverlight* and to test it by using it to model glycolysis in erythrocytes. The long term goal of this project is to enable and encourage the collaborative development of meta-simulations. It is hoped that in laying the foundation for the development of collaborative features in this project that the long term objectives of the project can be accomplished by future versions of the software. This study required the development of the simulator as well as several supporting technologies, including a small computer algebra system and the modification of a King-Altman algorithm for use in *Silverlight*. The simulator simulated a model of glycolysis converged to a quasi-steady state and whose final steady state concentrations were relatively close to those produced by other more complex erythrocyte metabolism models. Last, while the current system is slower than that of current modeling engines, it is anticipated that through further development and optimization, it can become an important modeling tool for the community by being able to produce meta-models through real-time collaboration.

REFERENCES

1 Li, C. and Bernoff, J. (2008) Groundswell: winning in a world transformed by social technologies. Harvard Business Press, Boston, Mass.

2 Alves, R., Antunes, F. and Salvador, A. (2006) Tools for kinetic modeling of biochemical networks. Nature Biotechnology **24**, 667-671

3 Mulquiney, P. J. and Kuchel, P. W. (2003) Modeling metabolism with Mathematica : detailed examples including erythrocyte metabolism. CRC Press, Boca Raton, Fla.

4 Takahashi, K., Yugi, K., Hashimto, K., Yamada, Y., Pickett, C. J. F. and Tomita, M. (2002) Computational challenges in cell simulation: a software engineering approach. IEEE Intelligent Systems **17**, 64-71

5 Jacobasch, G., Miakami, S. and Rapoport, S. M. (1974) Glycolysis of the erythrocyte. In Cellular and Molecular Biology of Erythrocytes (Yoshikawa, H. and Rapoport, S. M., eds.), University Park Press, Tokyo

6 Hinterberger, U., Ockel, E., Gerkscher-Mothes, W. and Rapoport, S. (1961) On the nature of allosteric transitions: a plausible model. Acta Biol. Med. Germ. **7**, 50-56

7 Magnani, M., Stocchi, V., Serafini, G. and Bossu, M. (1983) The effects of buffers and pH on rabbit red blood cell hexokinase. Italian Journal of Biochemistry **32**, 28-35

8 Monod, J., Wyman, J. and Changeux, J. P. (1965) On the nature of allosteric transitions: a plausible model. Molecular Biology **12**, 88-118

9 Mulquiney, P. J., Bubb, W. A. and Kuchel, P. W. (1999) Model of 2,3-bisphosphoglycerate metabolism in the human erythrocyte based on detailed enzyme kinetic equations: in vivo kinetic characterization of 2,3-bisphosphoglycerate synthase/ phosphatase using 13C and 31P NMR. Biochemistry **342**, 567-580

10 Surgenor, D. M. and Bishop, C. W. (1974) The red blood cell. Academic Press, New York

11 Yoshikawa, H. and Rapoport, S. M. (1974) Cellular and molecular biology of erythrocytes. University Park Press, Baltimore, MD

12 Demin, O. and Goryanin, I. (2009) Kinetic modeling in systems biology. CRC Press, Boca Raton, FL

13 Selkov, E., Basmanova, S., Gaasterl, T., Goryanin, I., Gretchkin, Y., Maltsev, N., Nenashev, V., Overbeek, R., Panyushkina, E., Pronevitch, L., Selkov (Jr.), E. and Yunus, I. (1996) The metabolic pathway collection from EMP: the enzymes and metabolic pathways database. Nucleic Acids Research **24**, 26-28

14 Schomburg, I., Chang, A. and Schomburg, D. (2002) BRENDA, enzyme data and metabolic information. Nucleic Acids Research **30**,47-49

15 Voet, D., Voet, J. G. and Pratt, C. W. (2006) Fundamentals of biochemistry: life at the molecular level. Wiley, Hoboken, N.J.

16 Lehninger, A. L., Nelson, D. L. and Cox, M. M. (2005) Lehninger principles of biochemistry. W.H. Freeman, New York

17 Gerber, G., Preissler, H., Heinrich, R. and Rapoport, S. M. (1974) Hexokinase of human erythrocytes. Eur. J. Biochem. ***45*** 39-52

18 Segel, I. H. (1975) Enzyme kinetics: behavior and analysis of rapid equilibrium and steady state enzyme systems. Wiley, New York

19 Monod, J., Wyman, J. and Changeux, J.-P. (1965) On the nature of allosteric transitions: a plausible model. Journal of Molecular Biology **12**, 88-118

20 Mulquiney, P. J. and Kuchel, P. W. (1999) Model of 2,3-bisphosphoglycerate metabolism in the human erythrocyte based on detailed enzyme kinetic equations: equation and parameter refinement. Biochemistry **342**, 581-596

21 Rapoport, T. A., Heinrich, R., Jacobasch, G. and Rapoport, S. (1974) A linear steady-state treatment of enzymatic chains. Eur. J. Biochem. **42**, 107-120

22 Rapoport, T. A. and Heinrich, R. (1975) Modeling of the glycolysis of human erythrocytes. BioSystems **7**, 120-129

23 Heinrich, R. and Rapoport, S. M. (1977) Metabolic regulation and mathematical models. Progress in Biophysics and Molecular Biology **32**, 1-82

24 Rapoport, T. A., Heinrich, R. and Rapoport, S. M. (1976) The regulatory principles of glycolysis in erythrocytes in vivo and in vitro. Biochemistry Journal **154**, 449-469

25 Ataullakhanov, F. I., Vitvitsky, V. M., Zhabotinsky, A. M., Pichugin, A. V., Platonova, O. V., Kholodenko, B. N. and Ehrlich, L. I. (1981) The regulation of glycolysis in human erythrocytes: the dependence of the glycolytic flux on the ATP concentration. Eur. J. Biochem. **115**, 359-365

26 Joshi, A. and Palsson, B. O. (1990) Metabolic dynamics in the human red cell: Part Ill--metabolic reaction rates. Journal of Theoretical Biology **142**, 41-68

27 Mulquiney, P. J. and Kuchel, P. W. (1999) Model of 2,3-bisphosphoglycerate metabolism in the human erythrocyte based on detailed enzyme kinetic equations: computer simulation and metabolic control analysis. Biochemistry **342**, 597-604

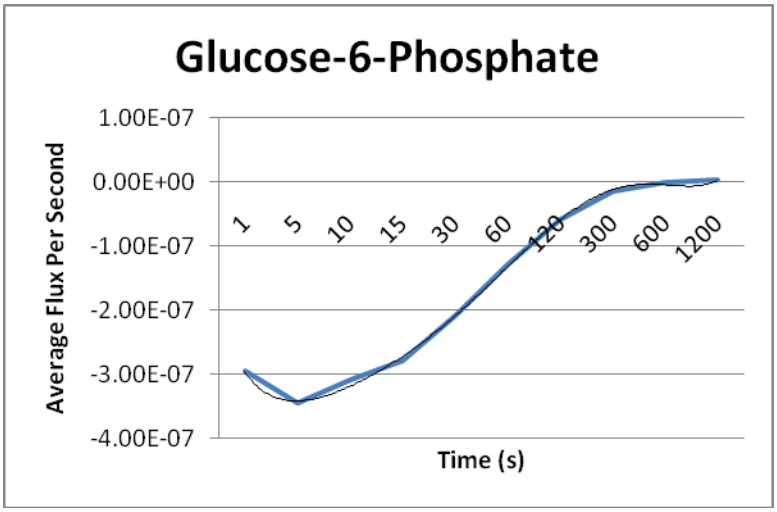
28 Cornish-Bowden, A. (1979) Fundamentals of enzyme kinetics. Butterworths, London, Boston, MA

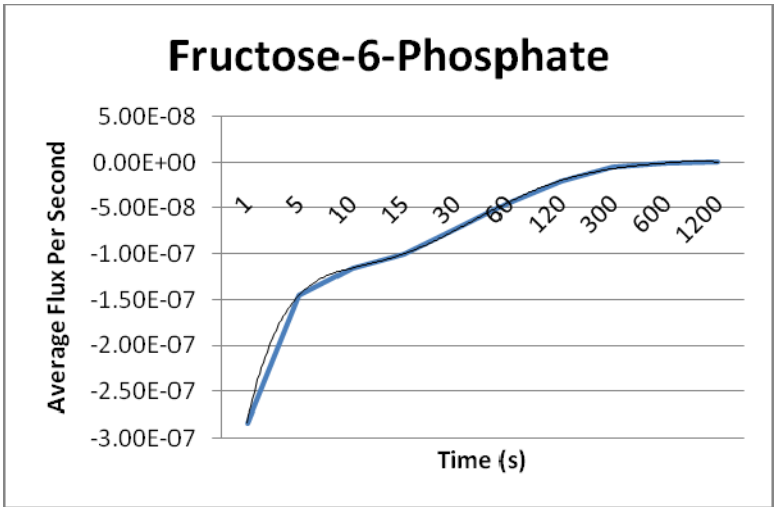
29 Cheney, E. W. and Kincaid, D. (2007) Numerical mathematics and computing. Brooks/Cole, Belmont, CA

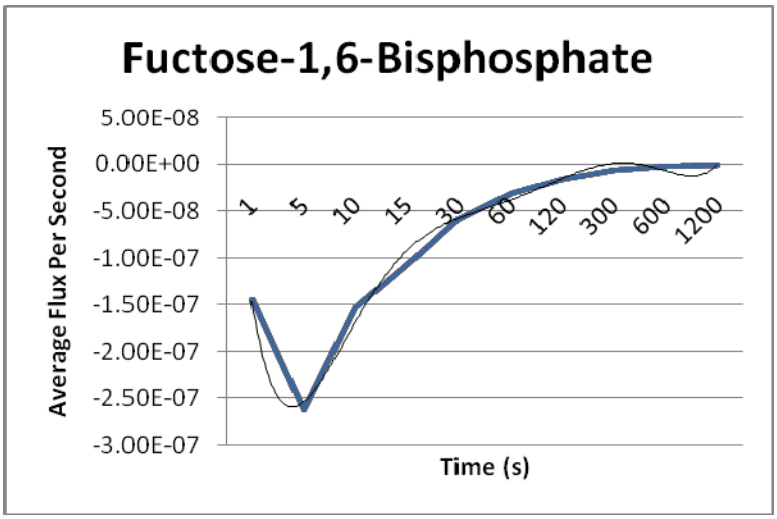
APPENDIX A

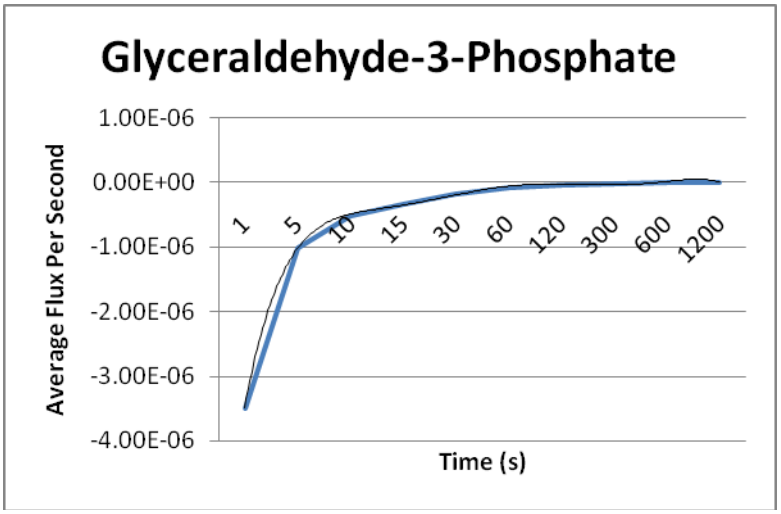
The following is a table and a list of corresponding graphs showing the average flux of each of the substrates whose concentrations were variable converging toward zero as time increased.

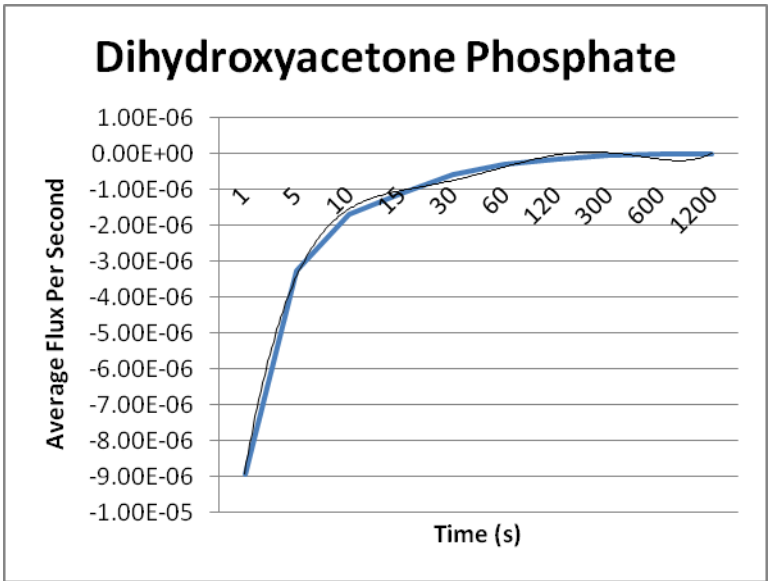
Elapsed Seconds	1	5	10	15	30	60	120	300	600	1200
Elapsed Minutes	0.02	0.08	0.17	0.25	0.50	1.00	2.00	5.00	10.00	20.00
Glucose-6-Phosphate	-2.9E-07	-3.4E-07	-3.1E-07	-2.8E-07	-2.1E-07	-1.3E-07	-6.0E-08	-1.5E-08	-1.8E-09	2.7E-09
Fructose-6-Phosphate	-2.8E-07	-1.4E-07	-1.2E-07	-1.0E-07	-7.2E-08	-4.3E-08	-2.0E-08	-5.0E-09	-7.3E-10	8.1E-10
Fuctose-1,6-Bisphosphate	-1.4E-07	-2.6E-07	-1.5E-07	-1.1E-07	-5.8E-08	-3.1E-08	-1.6E-08	-6.4E-09	-3.2E-09	-1.6E-09
Glyceraldehyde-3-Phosphate	-3.5E-06	-1.0E-06	-5.2E-07	-3.5E-07	-1.8E-07	-8.9E-08	-4.5E-08	-1.8E-08	-8.7E-09	-4.1E-09
Dihydroxyacetone Phosphate	-9.0E-06	-3.3E-06	-1.7E-06	-1.1E-06	-5.8E-07	-2.9E-07	-1.5E-07	-5.8E-08	-2.8E-08	-1.3E-08
1,3-Bisphosphoglycerate	-4.1E-07	-7.4E-08	-3.5E-08	-2.3E-08	-1.1E-08	-5.3E-09	-2.6E-09	-8.2E-08	-1.3E-10	3.8E-10
3-Phsphoglycerate	1.2E-05	4.9E-06	2.7E-06	1.9E-06	9.9E-07	4.6E-07	1.9E-07	7.3E-08	5.1E-08	4.6E-08
2-Phosphoglycerate	2.5E-06	9.4E-07	5.2E-07	3.6E-07	1.9E-07	8.7E-08	3.6E-08	1.4E-08	9.7E-09	8.3E-09
Phosphoenolpyruvate	-1.1E-06	-4.6E-07	-2.2E-07	-1.3E-07	-3.8E-08	-1.4E-08	-1.1E-08	-8.6E-10	6.4E-09	1.2E-08
Pyruvate	-1.3E-05	-4.7E-06	-2.3E-06	-1.4E-06	-4.6E-07	-1.8E-07	-6.3E-08	-6.9E-09	-4.5E-09	-2.7E-09
Lactate	1.7E-05	9.1E-06	6.7E-06	5.8E-06	4.6E-06	3.6E-06	2.7E-06	1.9E-06	1.1E-06	5.8E-07
Mg	-2.2E-03	-4.4E-04	-2.2E-04	-1.5E-04	-7.4E-05	-3.7E-05	-1.9E-05	-7.5E-06	-3.8E-06	-1.9E-06
MgAtp	2.0E-03	4.1E-04	2.0E-04	1.4E-04	6.8E-05	3.4E-05	1.7E-05	7.0E-06	3.6E-06	1.8E-06
MgAdp	1.8E-04	3.5E-05	1.7E-05	1.1E-05	5.5E-06	2.6E-06	1.2E-06	4.2E-07	1.7E-07	6.0E-08
Atp	-2.0E-03	-4.0E-04	-2.0E-04	-1.3E-04	-6.7E-05	-3.3E-05	-1.7E-05	-6.7E-06	-3.3E-06	-1.7E-06
Adp	-2.1E-04	-4.3E-05	-2.2E-05	-1.4E-05	-7.3E-06	-2.5E-08	-1.9E-06	-7.9E-07	-4.2E-07	-2.2E-07
Amp	8.4E-06	1.1E-06	4.3E-07	2.3E-07	4.9E-08	-2.5E-08	-4.4E-08	-3.7E-08	-2.9E-08	-2.0E-08
NAD	-3.6E-07	-6.5E-08	-3.4E-08	-2.4E-08	-1.3E-08	-9.5E-09	-7.7E-09	-5.9E-09	-5.8E-09	-5.7E-09

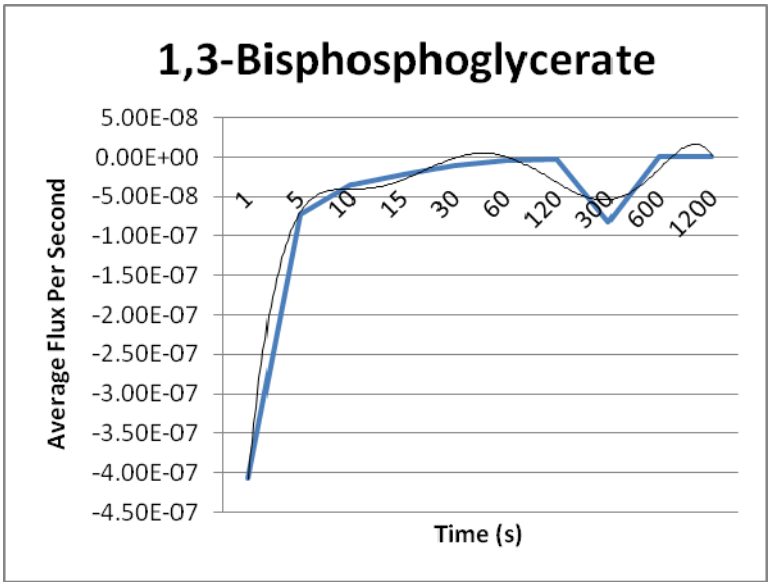

Glucose-6-Phosphate
Average Flux Per Second
1.00E-07
0.00E+00
-1.00E-07
-2.00E-07
-3.00E-07
-4.00E-07
1
5
10
15
30
60
120
300
600
1200
Time (s)

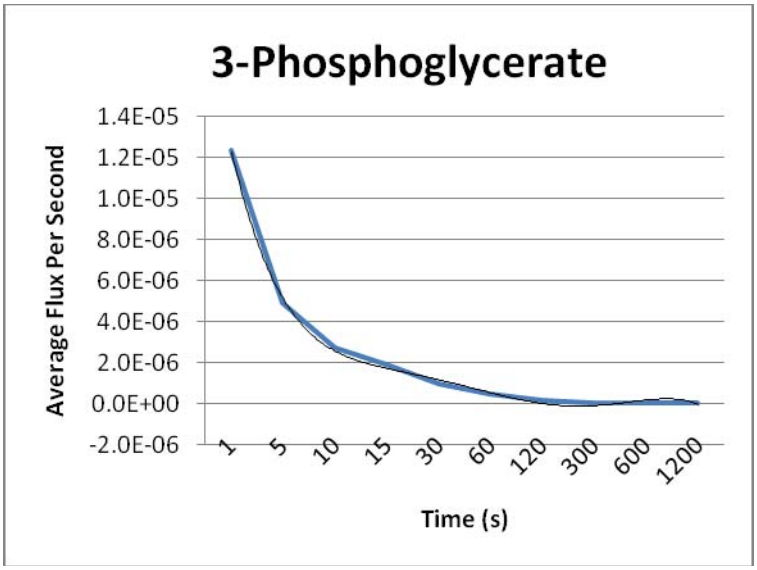

Fructose-6-Phosphate
Average Flux Per Second
5.00E-08
0.00E+00
-5.00E-08
-1.00E-07
-1.50E-07
-2.00E-07
-2.50E-07
-3.00E-07
1
5
10
15
30
60
120
300
600
1200
Time (s)

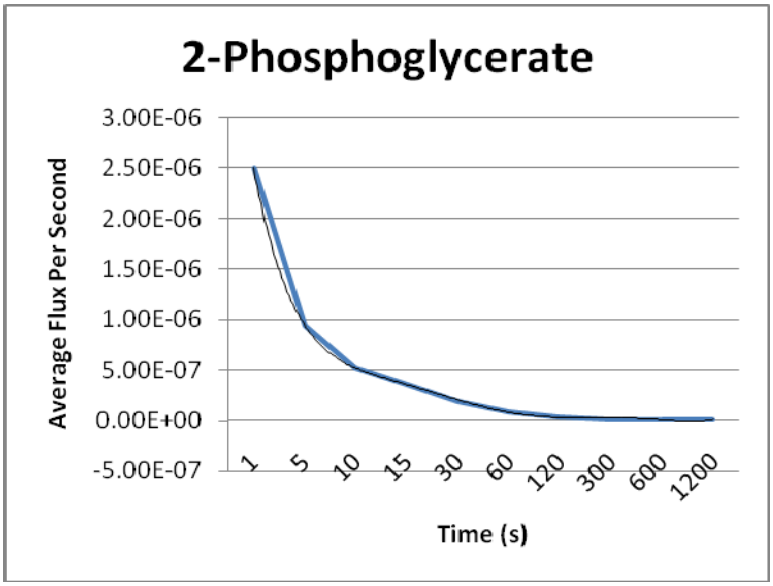

Fuctose-1,6-Bisphosphate
Average Flux Per Second
5.00E-08
0.00E+00
-5.00E-08
-1.00E-07
-1.50E-07
-2.00E-07
-2.50E-07
-3.00E-07
1
5
10
15
30
60
120
300
600
1200
Time (s)

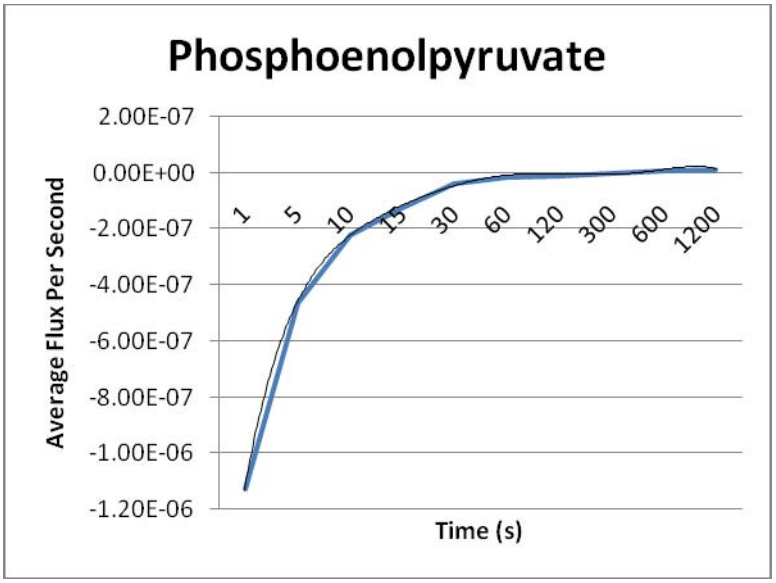

Glyceraldehyde-3-Phosphate
Average Flux Per Second
1.00E-06
0.00E+00
-1.00E-06
-2.00E-06
-3.00E-06
-4.00E-06
1
5
10
15
30
60
120
300
600
1200
Time (s)

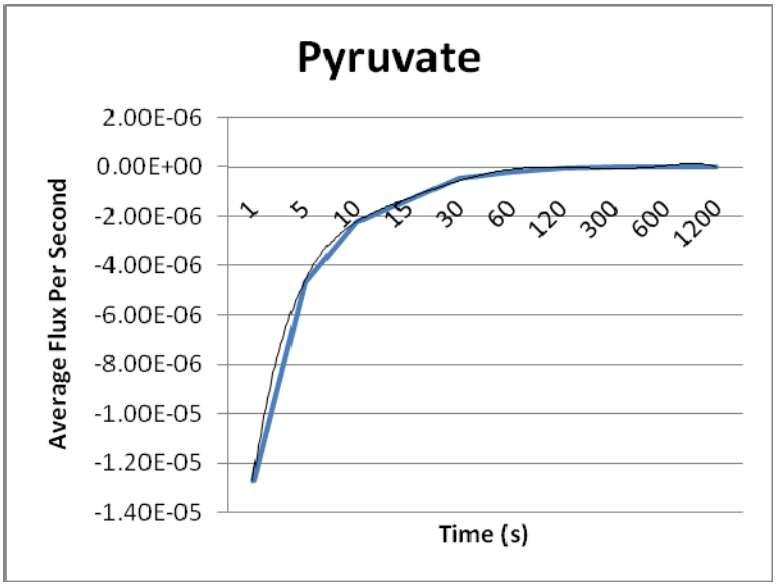

Dihydroxyacetone Phosphate
Average Flux Per Second
1.00E-06
0.00E+00
-1.00E-06
-2.00E-06
-3.00E-06
-4.00E-06
-5.00E-06
-6.00E-06
-7.00E-06
-8.00E-06
-9.00E-06
-1.00E-05
1
5
10
15
30
60
120
300
600
1200
Time (s)

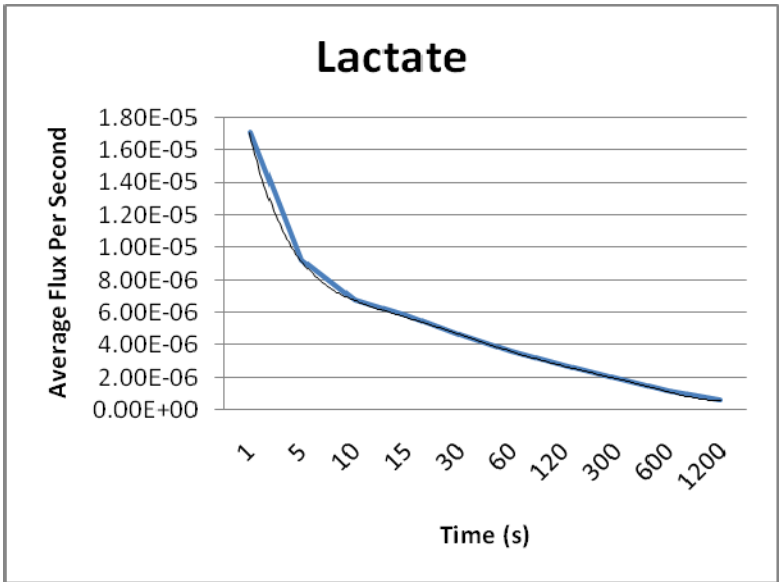

1,3-Bisphosphoglycerate
Average Flux Per Second
5.00E-08
0.00E+00
-5.00E-08
-1.00E-07
-1.50E-07
-2.00E-07
-2.50E-07
-3.00E-07
-3.50E-07
-4.00E-07
-4.50E-07
1
5
10
15
30
60
120
300
600
1200
Time (s)

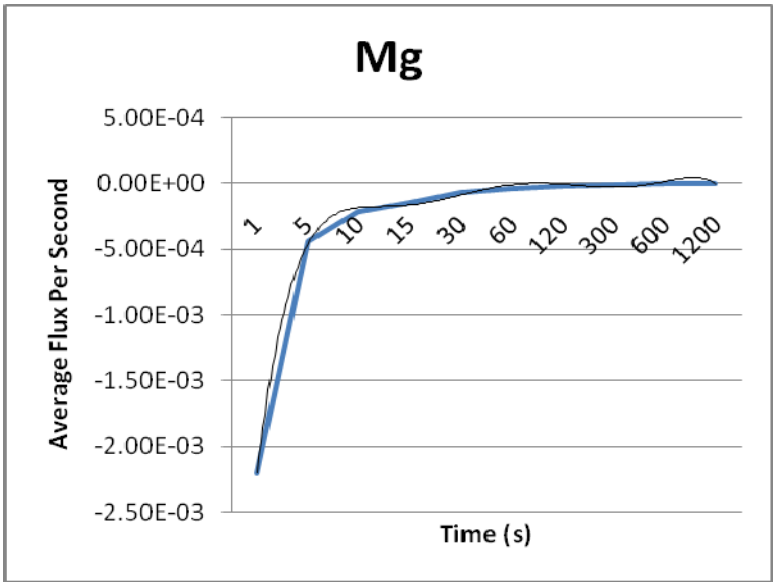

3-Phosphoglycerate
Average Flux Per Second
1.4E-05
1.2E-05
1.0E-05
8.0E-06
6.0E-06
4.0E-06
2.0E-06
0.0E+00
-2.0E-06
1
5
10
15
30
60
120
300
600
1200
Time (s)

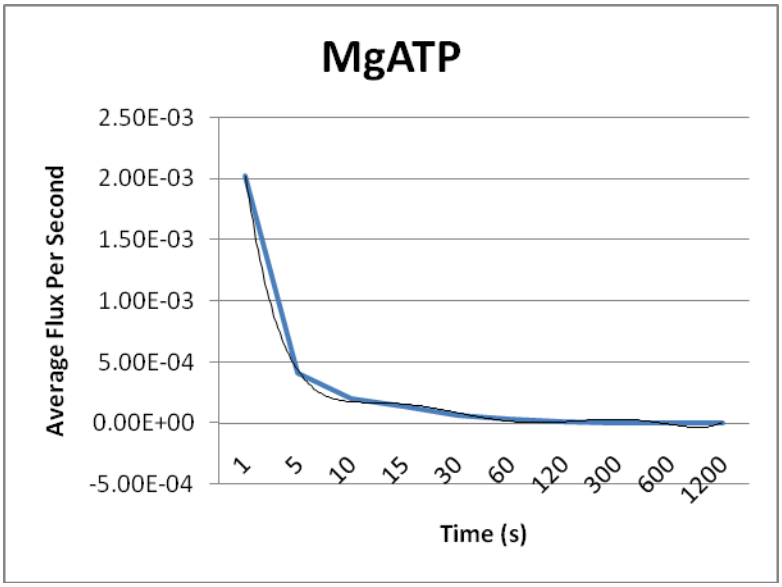

2-Phosphoglycerate
Average Flux Per Second
3.00E-06
2.50E-06
2.00E-06
1.50E-06
1.00E-06
5.00E-07
0.00E+00
-5.00E-07
1
5
10
15
30
60
120
300
600
1200
Time (s)

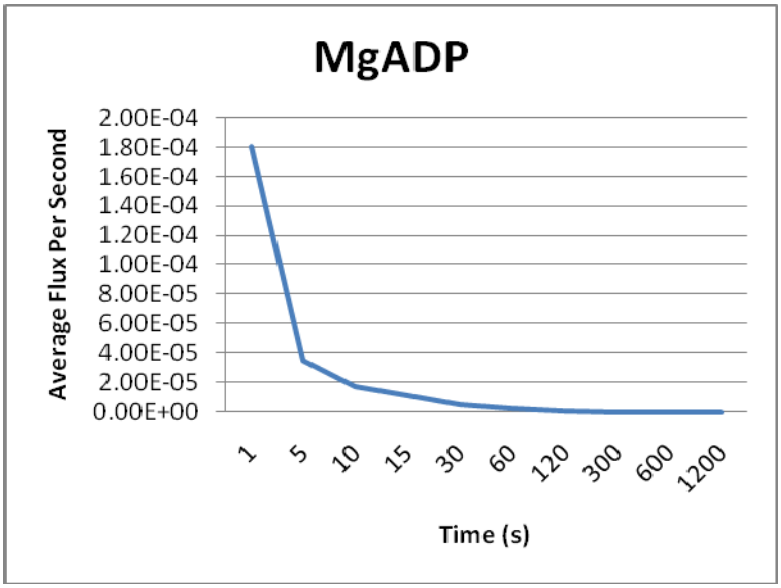

Phosphoenolpyruvate
Average Flux Per Second
2.00E-07
0.00E+00
-2.00E-07
-4.00E-07
-6.00E-07
-8.00E-07
-1.00E-06
-1.20E-06
1
5
10
15
30
60
120
300
600
1200
Time (s)

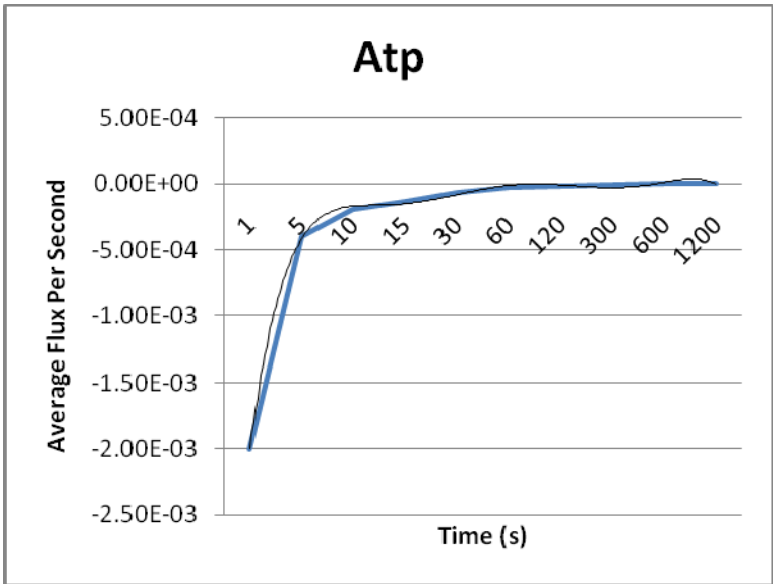

Pyruvate
2.00E-06
0.00E+00
-2.00E-06
-4.00E-06
-6.00E-06
-8.00E-06
-1.00E-05
-1.20E-05
-1.40E-05
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)

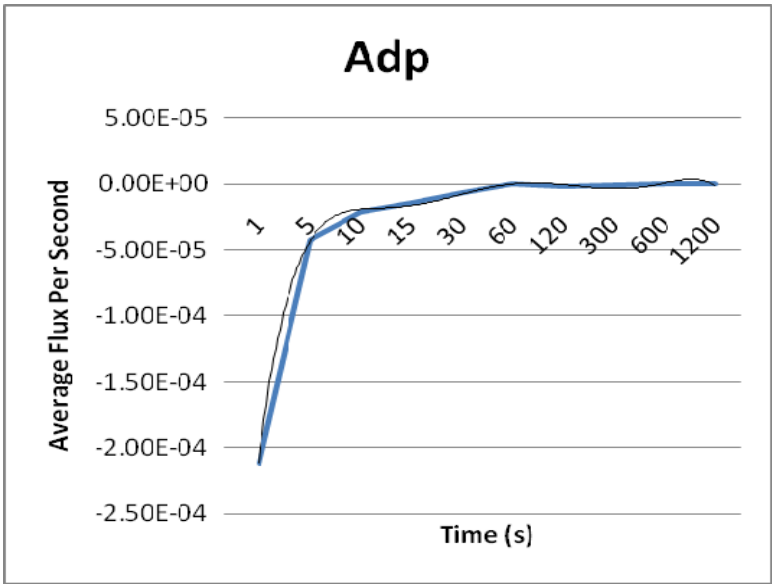

Lactate
1.80E-05
1.60E-05
1.40E-05
1.20E-05
1.00E-05
8.00E-06
6.00E-06
4.00E-06
2.00E-06
0.00E+00
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)

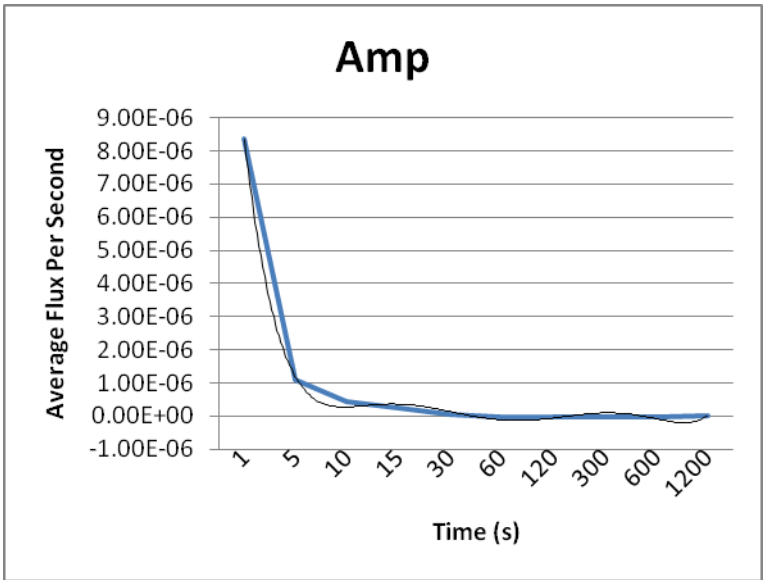

Mg
5.00E-04
0.00E+00
-5.00E-04
-1.00E-03
-1.50E-03
-2.00E-03
-2.50E-03
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)

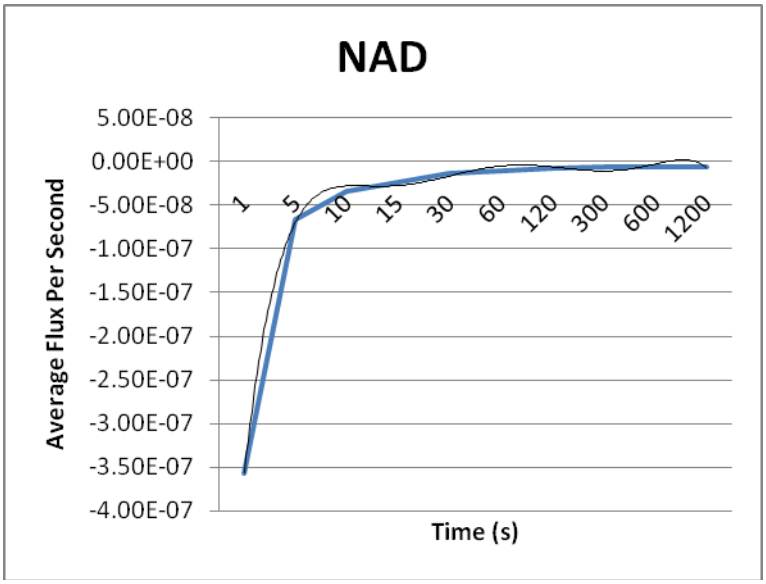

MgATP
2.50E-03
2.00E-03
1.50E-03
1.00E-03
5.00E-04
0.00E+00
-5.00E-04
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)

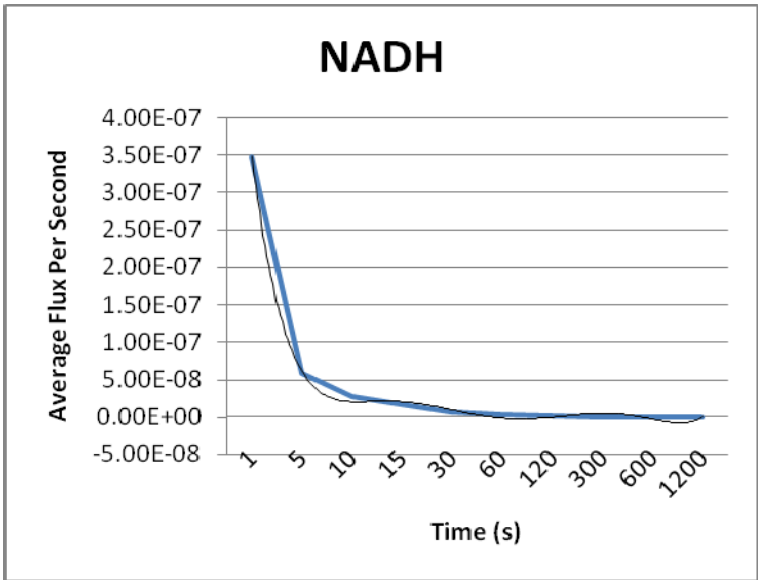

MgADP
Average Flux Per Second
2.00E-04
1.80E-04
1.60E-04
1.40E-04
1.20E-04
1.00E-04
8.00E-05
6.00E-05
4.00E-05
2.00E-05
0.00E+00
1
5
10
15
30
60
120
300
600
1200
Time (s)

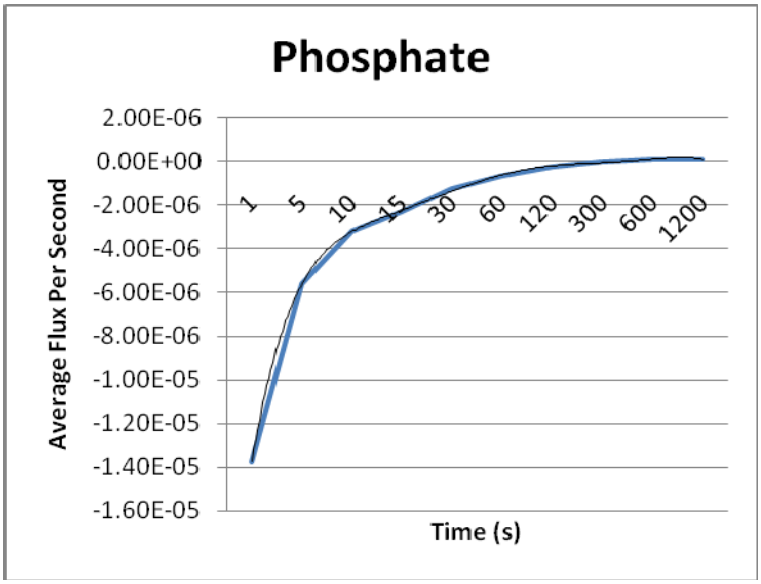

Atp
Average Flux Per Second
5.00E-04
0.00E+00
-5.00E-04
-1.00E-03
-1.50E-03
-2.00E-03
-2.50E-03
1
5
10
15
30
60
120
300
600
1200
Time (s)


Adp
5.00E-05
0.00E+00
-5.00E-05
-1.00E-04
-1.50E-04
-2.00E-04
-2.50E-04
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)


Amp
9.00E-06
8.00E-06
7.00E-06
6.00E-06
5.00E-06
4.00E-06
3.00E-06
2.00E-06
1.00E-06
0.00E+00
-1.00E-06
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)


NAD
5.00E-08
0.00E+00
-5.00E-08
-1.00E-07
-1.50E-07
-2.00E-07
-2.50E-07
-3.00E-07
-3.50E-07
-4.00E-07
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)


NADH
4.00E-07
3.50E-07
3.00E-07
2.50E-07
2.00E-07
1.50E-07
1.00E-07
5.00E-08
0.00E+00
-5.00E-08
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)


Phosphate
2.00E-06
0.00E+00
-2.00E-06
-4.00E-06
-6.00E-06
-8.00E-06
-1.00E-05
-1.20E-05
-1.40E-05
-1.60E-05
Average Flux Per Second
1
5
10
15
30
60
120
300
600
1200
Time (s)

VITA

Adam Laney Stevenson received his Bachelor of Science degree in Computer Engineering from Texas A&M University in 2005. He began a Master's program in Biology at Texas A&M University in the fall of the same year and graduated in the fall of 2009. His research interests include the application of evolutionary patterns in biology to the management of complexity in computer information systems, the building of flexible software designs, and the incorporation of biological feedback mechanisms into mainstream software. He plans to publish a book on these topics with a focus on convergence of computer software designs and biological patterns in the next few years.

Adam Stevenson may be reached at Evobolics, Inc located at 115 Slaughter Ranch Road, Trinidad, Texas, 75163. His email is a.l.stevenson@gmail.com.