ELSEVIER

Biophysical Chemistry 109 (2004) 1–15

Biophysical Chemistry

www.elsevier.com/locate/bpc

Review

Conservation analysis in biochemical networks: computational issues for software writers

Herbert M. Sauro[a,b,*], Brian Ingalls[c]

[a]*Keck Graduate Institute, 535 Watson Drive, Claremont 91711, USA*
[b]*Control and Dynamical Systems 107-81, California Institute of Technology, CA 91125, USA*
[c]*Department of Applied Mathematics University of Waterloo, Waterloo, ON, Canada N2L 3G1*

Received 24 June 2003; received in revised form 23 August 2003; accepted 25 August 2003

Abstract

Large scale genomic studies are generating significant amounts of data on the structure of cellular networks. This is in contrast to kinetic data, which is frequently absent, unreliable or fragmentary. There is, therefore, a desire by many in the community to investigate the potential rewards of analyzing the more readily available topological data. This brief review is concerned with a particular property of biological networks, namely structural conservations (e.g. moiety conserved cycles). There has been much discussion in the literature on these cycles but a review on the computational issues related to conserved cycles has been missing[1]. This review is concerned with the detection and characterization of conservation relations in arbitrary networks and related issues, which impinge on simulation simulation software writers. This review will not address flux balance constraints or small-world type analyses in any significant detail.
© 2003 Elsevier Science B.V. All rights reserved.

Keywords: Biochemical networks; Flux balance; Conservation relations; Software; Simulation

1. Introduction

In recent years there has been a renewed and vigorous interest in understanding biochemical networks through simulation and mathematical analysis. This has resulted in the development of new exchange standards such as SBML [22] and CellML [16], new theoretical analyses [13,47] and new open source simulation tools such as Jarnac [36], BioSPICE [4] and commercial ventures such as Gene Network Sciences [15] and Entelos [10].

One of the ways in which researches are trying to understand biochemical networks is through computer simulation. However, this type of study requires access to detailed kinetic data, which is generally unavailable. This issue has been highlighted in the literature (cf., e.g. Ref. [13]); there have been successes of course, for example [2,5]

*Corresponding author. Tel.: +1-909-607-0377; fax: +1-909-607-8086.
E-mail address: hsauro@kgi.edu (H.M. Sauro).

[1] The term 'cycle' is used here for historical reasons. There are clear but rare instances in networks where elements of a conservation law do not form cyclic structures.

0301-4622/04/$ - see front matter © 2003 Elsevier Science B.V. All rights reserved.
doi:10.1016/j.bpc.2003.08.009

but in general, with the present state of knowledge, it is difficult to build accurate dynamical models of biochemical networks.[2] However, with the advent of genomic wide studies, topological data on networks is readily available. In particular Palsson's group in San Diego have derived a number of network topologies from genomic data [9]. Stefan Schuster, one of the early pioneers in this area has also made significant contributions [39]. A key question that is often asked is what constraints does the topology of the network impose on the dynamics of the network [6]. In addition, it has been known for some time [28] that network topology is an important consideration to those who write reaction network simulation software.

The analysis of stoichiometric networks is not a new field. In the early 1960s the chemical engineering community published a number of pioneering papers, in particular Aris [1] and later by Horn and Jackson [21], Clarke [6] and Feinberg [12]. Much of this work is directly relevant to the analysis of biochemical networks. The topic continues to be an active area of research [11].

This review will be concerned with the analysis of conservation laws derived from network topology, including algorithms for determining conservation constraints and practical issues relating to the development of simulators of biochemical networks. An excellent source of material related to the analysis of the stoichiometry matrix can be found in the text book by Heinrich and Schuster [17].

2. Reaction networks

2.1. Stoichiometry matrix

The analysis of any biochemical network starts by considering the network's topology. This information is embodied in the stoichiometry matrix, ***N***. The columns of this matrix correspond to the distinct chemical reactions in the network, the rows to the molecular species, one row per species. Thus, the intersection of a row and column in the matrix indicates whether a certain species takes part in a particular reaction or not and, according to the sign of the element, whether it be a reactant or product, and by the magnitude, the relative quantity of substance that takes part in that reaction. Stoichiometry thus concerns the relative mole amounts of chemical species that react in a particular reaction; it does not concern itself with the rate of reaction. The stoichiometry matrix is, therefore, constant (except on evolutionary timescales when the cast of proteins might change) and is determined by the genetic constitution of the organism. As a result, information on stoichiometry is more readily available than kinetic rate laws and values of kinetic parameters. The stoichiometry matrix reflects the total potential of an organisms' reaction network.

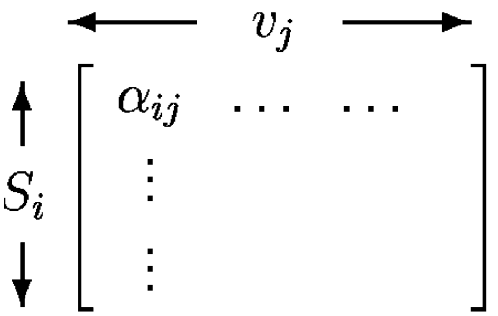


Fig. 1. Stoichiometry matrix: $\boldsymbol{N}$: $m \times n$, where α_{ij} is the stoichiometric coefficient.

The stoichiometric matrix represents a compact mathematical representation of a biochemical network Fig. 1. If a given network is composed of m molecular species involved in n reactions then the stoichiometry matrix is an $m \times n$ matrix. Only those molecular species, which evolve through the dynamics of the system are included in this count. Any source and sink species needed to sustain a steady state (non-equilibrium in the thermodynamic sense) are set at a constant level and, therefore, do not have corresponding entries in the stoichiometry matrix.

2.2. System equations

To fully characterize a system one needs to consider the kinetics of the individual reactions as

[2] See model databases at: http: //jjj.biochem.sun.ac.za, http: //ww.sys-bio.org, http: //www.sbml.org or http: //www.cellml.org for lists of models.

well as the network's topology. Modelling the reactions by differential equations, we arrive at a system equation, which involves both the stoichiometry matrix and the *rate vector*, thus:

$$\frac{d\boldsymbol{S}}{dt} = \boldsymbol{N}\boldsymbol{v}$$

where $\boldsymbol{N}$ is the $m \times n$ stoichiometry matrix and v is the n dimensional rate vector, whose ith component gives the rate of reaction i as a function of the species concentrations. This review is concerned exclusively with the analysis of $\boldsymbol{N}$ and in particular with the computation of structural conservation laws from the stoichiometry matrix.

2.3. Structural properties

There are two key structural properties, which can be derived from the stoichiometry matrix. These are flux balance constraints and mass conservation constraints, both of which are derived from the law of conservation of mass.

2.3.1. Flux balance

The flux balance constraints are only valid when the system is at steady state, in which each molecular concentration (or *species pool*) reaches a steady state value. As a result, the sum of all fluxes into a species pool must equal the sum of all fluxes out of the pool.

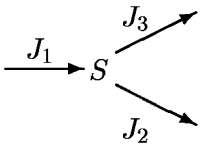
$$\sum \text{Fluxes In} = \sum \text{Fluxes Out}$$

For each species, this is equivalent to the statement that $\boldsymbol{N}_i v = 0$, where $\boldsymbol{N}_i$ is the ith row of $\boldsymbol{N}$. Altogether then, we arrive at the matrix equation

$$\boldsymbol{N}\boldsymbol{v} = 0.$$

This constraint is imposed on the distributions of steady state fluxes at all nodes in the network and is in direct analogy with Kirchhoff's first law of current flow.

As a simple example, a branch point, as shown below:



where J_i are the steady state fluxes, has the restriction, at steady state, that,

$$J_1 - (J_2 + J_3) = 0$$

Thus, to characterize the steady state flow through this system, we need only measure two of the fluxes, as the third is then determined. Of course, the conclusion is obvious in this trivial example, but for large systems the flux balance constraints can be a valuable tool for the analysis of steady state flow. In general, one can divide the set of fluxes into two subsets, the independent fluxes and the dependent fluxes. The independent fluxes (J_1 and J_2 in the above example, say) can take any values, while the values of the dependent fluxes (J_3) are fixed by the independent fluxes and the flux constraints. This dependence is the essence of flux balance analysis; this is an extensive and well-documented topic. Interested readers are encouraged to consult Stephanopoulos and Heinrich and Schuster for more detailed information [44,17].

2.3.2. Elementary modes

Elementary modes have received a considerable amount of attention in the literature, especially among the metabolic engineering community as a means of defining the notion of a pathway and enabling the computation of maximal conversion yields [41]. In addition, elementary modes have important theoretical applications in the analysis of network stability [6].

Elementary modes are defined as solutions to the homogeneous equation

$$\boldsymbol{N}\boldsymbol{E} = 0$$

where reactions in the network are categorized as

either reversible or irreversible. Irreversible reactions in particular constrain the solution, such that $\boldsymbol{E}_{\text{irr}} \geq 0$.

Each elementary mode $\boldsymbol{E}$ represents a path through the network. The irreversibility condition ensures that if a path traverses an irreversible reaction it does so in the allowed direction. Calculation of elementary modes is a fairly complex operation and algorithms for their computation make use of convex analysis, since the requirement is that all solutions must lie in the positive orthant. (More precisely, the set of elementary modes spans a convex cone in the positive orthant with apex at the origin.) For further details, interested readers are referred to the excellent papers by [39,38,41]. For a more general and readable account of elementary modes and extreme currents, the reader is also directed to the review by Papin [27].

2.3.3. Moiety conservation analysis

Molecular subgroups, which are conserved during the evolution of a network, are termed *conserved moieties* [32]. The total amount of a particular moiety in a network is time invariant and is determined solely by the initial conditions imposed on the system.[3]

A typical example of a conserved moiety in a computational model is the conservation of adenine nucleotide, i.e. when the total amount of ATP, ADP and AMP is constant during the evolution of the system. Other examples include NAD/NADH, phosphate and so on. Fig. 2 illustrates the simplest possible network, which displays a conserved moiety, in this case the total mass, $S_1 + S_2$ is constant during the evolution of the network.

Moiety conservation analysis plays a dual role to the analysis of flux balance. Whereas flux balance is concerned with the conservation of mass as it flows into and out of species nodes; moiety conservation analysis is concerned with the conservation of mass as it moves around closed loops in the network. Again we see a direct analogy to electrical engineering since moiety conservation analysis is related to Kirchhoff's second law of potentials. In this case, the potentials correspond to the change in moiety mass as we traverse a closed loop [18,8].

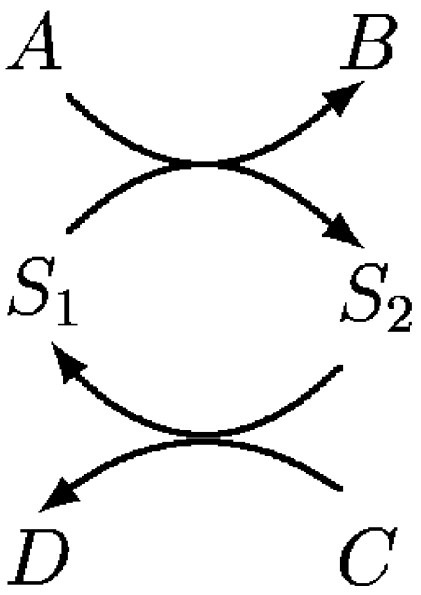


Fig. 2. Simple conserved cycle.

2.4. Practical implications

What are the implications of these structural constraints? There are a number of important practical reasons why conservation relations should be identified as a preliminary step in a system analysis.

Many traditional drugs, for example pesticides and anti-pathogen agents, work by disrupting either flux or metabolite levels to an extent that is harmful to the organism, either by reducing an important flux to unacceptably low levels or increasing the level of a metabolite to toxic proportions. Conservation constraints can impose hard limits to the extent to which a drug can influence a metabolite level. This effect is separate from any kinetic constraints that may exist. Similarly, the flux constraints impose their own limits on the degree to which fluxes can be altered. Thus, stoichiometry analysis is an important initial evaluation of whether manipulating a particular target

[3] There are rare cases when a 'conservation' relationship arises out of a non-moiety cycle. This does not affect the mathematics but only the physical interpretation of the relationship. For example, $A \rightarrow B + C$; $B + C \rightarrow D$ has the conservation, $B - C = T$.

might be effective or not. For an interesting example of these constraints in operation, the reader is referred to the pioneering work of Bakker et al. [2,3] and also [7,8].

Another practical implication of moiety conservation concerns the computational load involved in solving the system equations and thus concerns also the implementation of computer software. Just as the flux balance constraints allowed a subset of the fluxes to be classified as dependent, the mass conservation constraints allow the set of species to be subdivided into independent and dependent species. The dependent species have the property that their concentrations can be calculated from the concentrations of the independent species concentrations. Thus, the number of system equations to be solved is reduced, easing the computational burden. In fact many biochemical simulation packages will automatically check for moiety conservations and perform this simplification before performing any analysis of the system equations. This is especially important for very large models. For example, in the *Escherichia coli* model obtained from Palsson's web site at http://gcrg.ucsd.edu/organisms/ecoli.html, approximately 5% of the differential equations are redundant, and so could be safely eliminated from the model by using moiety conservation constraints.

Moreover, the identification of moiety conserved cycles is critical when computing steady state solutions directly. If the linear constraints on the conserved pools are not taken into account, there will be no unique steady state solution (the system is underdetermined and the solver is liable to 'wander' over the space of possible solutions).

Apart from the practical reasons for determining conserved cycles there are also theoretical reasons. For example, the analysis of biochemical control [14,19,35] is dependent on the identification of conserved moieties.

3. Theory of conservation analysis

Conserved moieties in the network reveal themselves as linear dependencies in the rows of the stoichiometry matrix. The question then arises: how, given a stoichiometry matrix, can we identify linear dependencies and determine the corresponding conserved moieties [18,8]?

If we examine the system equations for the model depicted in Fig. 2, it is easy to see that the rate of appearance of S_1 must equal the rate of disappearance of S_2, in other words $dS_1/dt = -dS_2/dt$. This identity is a direct result of the conservation of mass, namely that the sum $S_1 + S_2$ is constant through out the evolution of the system.

In this case, the stoichiometry matrix has two rows $[1 \quad -1]$ and $[-1 \quad 1]$. Since either row can be derived from the other by multiplication by -1, they are linearly dependent and the rank of the matrix is 1. Whenever the network exhibits conserved moieties, there will be dependencies among the rows of $\boldsymbol{N}$, and so the rank of $\boldsymbol{N}$ ($rank(\boldsymbol{N})$) will be less than m, the number of rows of $\boldsymbol{N}$. The rows of $\boldsymbol{N}$ can be rearranged so that the first $rank(\boldsymbol{N})$ rows are linearly independent. The metabolites, which correspond to these rows, can then be defined as the *independent species* (S_i). The remaining m-$rank(\boldsymbol{N})$ are the *dependent species* (S_d).

In the simple example above, there is one independent species, S_1 and one dependent species, S_2 (or, alternatively, S_2 is independent and S_1 dependent).

Once the matrix $\boldsymbol{N}$ has been rearranged as described above, we can partition it as

$$\boldsymbol{N} = \begin{bmatrix} \boldsymbol{N_R} \\ \boldsymbol{N} \end{bmatrix}$$

where the submatrix $\boldsymbol{N_R}$ is full rank, and each row of the submatrix $\boldsymbol{N}_0$ is a linear combination of the rows of $\boldsymbol{N_R}$. Note that if there are no structural conserved cycles in the network, then m equals the $rank$ ($\boldsymbol{N}$) and $\boldsymbol{N}$ simply equals $\boldsymbol{N_R}$. Following Reder [31], we make the following construction. Since the rows of $\boldsymbol{N}_0$ are linear combinations of the rows of $\boldsymbol{N_R}$ we can define a link-zero matrix $\boldsymbol{L}_0$ which satisfies

$$\boldsymbol{N}_0 = \boldsymbol{L}_0 \boldsymbol{N_R} \tag{1}$$

We can combine $\boldsymbol{L}_0$ with the identity matrix (of dimension $rank(\boldsymbol{N})$) to form the link matrix, $\boldsymbol{L}$,

thus:

$$\boldsymbol{L}=\begin{bmatrix}\boldsymbol{I}\\ \boldsymbol{L}_0\end{bmatrix}$$

Hence, we can write:

$$\boldsymbol{N}=\begin{bmatrix}\boldsymbol{N_R}\\ \boldsymbol{N_0}\end{bmatrix}=\begin{bmatrix}\boldsymbol{I}\\ \boldsymbol{L_0}\end{bmatrix}\boldsymbol{N_R}=\boldsymbol{L}\boldsymbol{N_R}$$

For networks without structural conservations the $\boldsymbol{L}$ matrix reduces to the identity matrix, $\boldsymbol{I}$. By partitioning the stoichiometry matrix into a dependent and independent set we also partition the system equation. The full system equation, which describes the dynamics of the network is thus:

$$\begin{bmatrix}\boldsymbol{I}\\ \boldsymbol{L_0}\end{bmatrix}\boldsymbol{N_R}\upsilon=\frac{\mathbf{dS}}{\mathbf{d}t}=\begin{bmatrix}\mathbf{d}\boldsymbol{S_i}/\boldsymbol{dt}\\ \mathbf{d}\boldsymbol{S_d}/\boldsymbol{dt}\end{bmatrix}$$

where the terms $\mathrm{d}S_i/\mathrm{d}t$ and $\mathrm{d}S_d/\mathrm{d}t$ refer to the independent and dependent rates of change, respectively. From the above equation, we see that

$$\frac{\mathrm{d}S_d}{\mathrm{d}t}=\boldsymbol{L}_0\frac{\mathrm{d}S_i}{\mathrm{d}t}.$$

Integrating, we find

$$S_d(t)-S_d(0)=\boldsymbol{L}_0[S_i(t)-S_i(0)]$$

for all time t. Introducing the constant vector $\boldsymbol{T}=S_d(0)-\boldsymbol{L}_0S_i(0)$, we can write the above equation as

$$[-\boldsymbol{L_0}\quad\boldsymbol{I}]\begin{bmatrix}\boldsymbol{S_i}\\ \boldsymbol{S_d}\end{bmatrix}=\boldsymbol{T}$$

Recalling that $S=(S_i,S_d)$, we can introduce $\boldsymbol{\Gamma}=[-\boldsymbol{L0}\quad\boldsymbol{I}]$, and write this concisely as

$$\boldsymbol{\Gamma}\boldsymbol{S}=\boldsymbol{T}$$

We will call $\boldsymbol{\Gamma}$ the conservation matrix.

Each row of the conservation matrix relates to a particular conserved cycle and thus the number of rows indicates the number of linearly independent conserved moieties in the network. The elements in a particular row indicate which metabolite species contribute to a particular cycle. Algorithms for computing the conservation matrix involve directly or indirectly computing the $\boldsymbol{L}_0$ matrix.

In the example shown in Fig. 2, the conservation matrix, $\boldsymbol{\Gamma}$ is

$$\boldsymbol{\Gamma}=[1\quad 1]$$

4. Computing the conservation matrix

There are a variety of related approaches to computing $\boldsymbol{\Gamma}$, the conservation matrix. The choice of which approach to take depends on the problem and the type of software available to the practitioner. The first criterion to consider is the size of the network. Large networks, such as those generated by genomic studies, require special attention (appropriate methods will be discussed in a later section). For small networks, say up to thirty to sixty molecular species, a variety of 'simple' methods are available that are more than adequate. Some of these simpler methods generate the conservation matrix directly while others generate the $\boldsymbol{L}$ matrix from which $\boldsymbol{\Gamma}$ can be determined. The first direct method, which we will describe is very useful in interactive maths environments such as Matlab or Maple.

One problem that all the methods suffer from, simple or otherwise is that they do not always necessarily generate the most meaningful conservation vectors. This will be discussed further in a later section. Little has been written on the practical issues relating to computing the conservation matrix, however, [20] is of interest in this respect.

4.1. By computing the null space of $\boldsymbol{N}$

We will first describe a simple method for computing the conservation matrix. The method is ideal for interactive computing, though is not necessarily the fastest approach.

Recall that

$$\boldsymbol{L}_0\boldsymbol{N_R}=\boldsymbol{N}_0$$

so that:

$$[-\boldsymbol{L_0} \quad \boldsymbol{I}]\begin{bmatrix}\boldsymbol{N_R}\\ \boldsymbol{N_0}\end{bmatrix}=\boldsymbol{0}$$

and, therefore:

$$[-\boldsymbol{L_0} \quad \boldsymbol{I}]\boldsymbol{N}=0$$

Note that the first term is the $\boldsymbol{\Gamma}$ conservation matrix, thus

$$\boldsymbol{\Gamma N}=0 \tag{2}$$

From this it is clear that the rows of $\boldsymbol{\Gamma}$ lie in the left null space of $\boldsymbol{N}$. Since $\boldsymbol{\Gamma}$ has *m-rank*($\boldsymbol{N}$) rows which are linearly independent (by construction), we conclude that the rows form a basis for the left nullspace of $\boldsymbol{N}$. Since computation of the right nullspace is more conventional we will transpose (Eq. (2)) to yield:

$$\boldsymbol{N}^T\boldsymbol{\Gamma}^T=0$$

and we can thus compute $\boldsymbol{\Gamma}^T$ by determining a basis for the (right) nullspace of $\boldsymbol{N}^T$.

Clearly, computing the conservation matrix by this method can be very easily carried out by interactive math tools such as Matlab, Maple or more specialized systems biology tools such as Jarnac [36]. Under Matlab one would simply enter: transpose (null (transpose (n)).

4.2. By the Gauss–Jordan method

Probably the fastest method to computing the conservation matrix directly is to row reduce the stoichiometry matrix [18,8]. If we take the stoichiometry matrix, $\boldsymbol{N}$ and premultiply by a series of elementary matrices, $\boldsymbol{E_i}$, we can transform $\boldsymbol{N}$ to $\boldsymbol{N_R}$, by row reduction. Let the product of these elementary matrices be given by $\boldsymbol{M}$, so that

$$\boldsymbol{MN}=\begin{bmatrix}\boldsymbol{N_R}\\ \boldsymbol{0}\end{bmatrix}$$

If we partition $\boldsymbol{M}$ into

$$\boldsymbol{M}=\begin{bmatrix}\boldsymbol{M_1}\\ \boldsymbol{M_0}\end{bmatrix}$$

we see that $\boldsymbol{M}_0\boldsymbol{N}=0$. That is, each row of $\boldsymbol{M}_0$ is an element of the nullspace of $\boldsymbol{N}^T$. Furthermore, the rows of $\boldsymbol{M}_0$ are linearly independent, since $\boldsymbol{M}$, being a product of elementary matrices, is invertible. Therefore, the rows of $\boldsymbol{M}_0$ are the same as the rows of $\boldsymbol{\Gamma}$ and thus $\boldsymbol{M}_0=\Gamma$, see Eq. (2).

In practice this method involves augmenting the stoichiometry matrix with an identity matrix, [$\boldsymbol{NI}$]. This augmented matrix is now row reduced. Once the reduction is complete the identity matrix will have been transformed into the $\boldsymbol{M}$ matrix as described above. The conservation matrix is then simply formed from the bottom *m-rank*($\boldsymbol{N}$) rows. Interested readers are referred to the papers [18,8] for a worked through example.

This method can be continued further to generate the $\boldsymbol{L}$ matrix. Since $\boldsymbol{M}$ is invertible, we have

$$\boldsymbol{N}=\boldsymbol{M}^{-1}\begin{bmatrix}\boldsymbol{N_R}\\ \boldsymbol{0}\end{bmatrix}.$$

If we partition $\boldsymbol{M}^{-1}$ into two column blocks, $\boldsymbol{A}$ and $\boldsymbol{B}$, we get

$$\boldsymbol{N}=[\boldsymbol{A} \quad \boldsymbol{B}]\begin{bmatrix}\boldsymbol{N_R}\\ \boldsymbol{0}\end{bmatrix}=\boldsymbol{AN_R}$$

That is, the left submatrix, $\boldsymbol{A}$ equals $\boldsymbol{L}$. Although this method does generate $\boldsymbol{L}$, it is much easier to extract the $\boldsymbol{L}$ matrix directly from conservation matrix $\boldsymbol{\Gamma}$.

4.3. Relationship to reduced row echelon form

From a theoretical point of view it is instructive to consider the relationship between the conservation matrix and the reduced row echelon form. A matrix where the rows have been reordered so that the independent rows are the top rows of the matrix will have the following structure when row reduced to echelon form:

$$\begin{bmatrix}\boldsymbol{I} & \boldsymbol{P}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}$$

If we row reduce N^T to this form, we find

$$\begin{bmatrix} \boldsymbol{I} & \boldsymbol{P} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \boldsymbol{\Gamma}^T = \boldsymbol{0}$$

Since $\boldsymbol{\Gamma} = [-L_0 \quad I]$ then

$$\begin{bmatrix} \boldsymbol{I} & \boldsymbol{P} \\ \boldsymbol{0} & \boldsymbol{0} \end{bmatrix} \begin{bmatrix} -\boldsymbol{L}_0^T \\ \boldsymbol{I} \end{bmatrix} = \boldsymbol{0}$$

Multiplying out the above equation yields the simple result:

$$\boldsymbol{P} = \boldsymbol{L}_0^T$$

from which we conclude that the unknown entry, $\boldsymbol{P}$, in the row reduced matrix is in fact $\boldsymbol{L}_0^T$. This means we can obtain the link-zero matrix, $\boldsymbol{L}_0$ simply by performing a row reduction to echelon form on the transpose of the stoichiometry matrix and extracting the $\boldsymbol{L}_0^T$ partition.

From a practical point of view, this method is not recommended because the full reduction of the stoichiometry matrix requires too many arithmetic operations. The closely-related but much more efficient Gauss–Jordan method described earlier is recommended.

4.4. Generating physically meaningful vectors

An issue that sometimes arises in determining structural conservations is whether the conservations have a simple physical interpretation. Consider the network shown in Fig. 3.

This network has two conserved cycles, $S_1 + S_2 + ES = T_1$ and $E + ES = T_2$. However, if we derive these relations using the row reduction technique we can, depending on the row order in the stoichiometry matrix, obtain the following relations:

$$S_1 + S_2 + ES = T_1$$

$$E - S_1 - S_2 = T_2$$

These relations are correct though the second relation contains negative terms, which may at times be difficult to interpret physically. In this example, T_2 actually equals the difference in the total amount of E and S moieties. A simpler relationship would be $E + ES = T_2$, where T_2 has the more satisfying interpretation of being the total amount of enzyme mass in the network. As a result of these difficulties there have been efforts to design algorithms whose aim is to generate conservation vectors which have only positive entries, in the hope that their physical interpretation is more obvious, e.g. Ref. [32]. Schuster and colleagues [42,40] have been most active in this area and have devised methods based on convex analysis. These methods are dual to those designed to discover elementary modes.

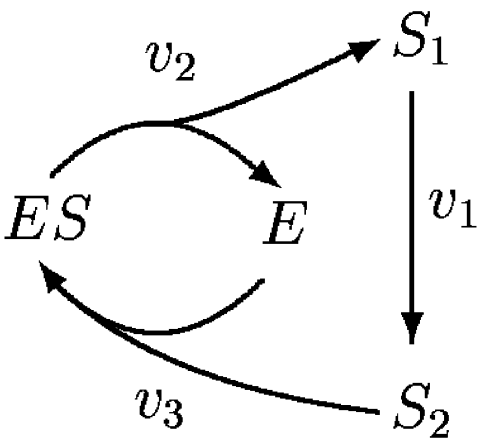


Fig. 3. Linked conserved cycles.

Fortunately, the form of the conservations laws is not always an important consideration, especially when the calculations are carried out internally by a simulator. Nevertheless, problems may arise when one needs to investigate how the total mass in a conserved cycle influences the system [35,24]. A clear understanding of the totals in the conserved relationships might be necessary to gain a satisfactory interpretation.

4.5. Large systems

Small networks are relatively easy to deal with when it comes to computing conservation constraints. Computing the conservation matrix for

large systems, however, posses numerical instability issues. For large systems the currently recommended approach to use singular value decomposition (SVD).

*4.5.1. Computing **L** by SVD*

It is possible to compute the matrix $\boldsymbol{L}$ directly by evaluating the pseudoinverse of $\boldsymbol{N_R}$ [46].

Since $\boldsymbol{N}=\boldsymbol{L}\boldsymbol{N_R}$, postmultiplying on both sides by $\boldsymbol{N_R^T}$ and rearranging, we find

$$\boldsymbol{L}=\boldsymbol{N}\boldsymbol{N_R}^{\mathrm{T}}\,(\boldsymbol{N_R}\boldsymbol{N_R}^{\mathrm{T}})^{-1} \tag{3}$$

The inverse of $\boldsymbol{N_R}\boldsymbol{N_R^T}$ is guaranteed to exist because $\boldsymbol{N_R}$ has full row rank. The matrix $\boldsymbol{N_R^T}(\boldsymbol{N_R}\boldsymbol{N_R^T})^{-1}$ is the pseudoinverse of $\boldsymbol{N_R}$, denoted $\boldsymbol{N_R^+}$. Rather than compute the pseudoinverse by inverting $(\boldsymbol{N_R}\boldsymbol{N_R^T})^{-1}$ by the traditional matrix inversion procedures, we can perform the computation using the much more stable technique of Singular Value Decomposition. SVD is usually associated with least-squares problems, but its uses go beyond this. For example, SVD can be useful in finding the inverse of an ill-condition matrix, i.e. one that is close to being singular.

SVD decomposes a matrix $\boldsymbol{A}$ into the product of three other matrices, usually denoted, $\boldsymbol{U}$, Σ and $\boldsymbol{V}$, that is:

$$\boldsymbol{A}=\boldsymbol{U}\boldsymbol{\Sigma}\boldsymbol{V}^T$$

The matrices $\boldsymbol{U}$ and $\boldsymbol{V}$ are orthogonal (i.e. $\boldsymbol{U}^T\boldsymbol{U}=1$ and $\boldsymbol{V}\boldsymbol{V}^T=1$). The matrix $\boldsymbol{\Sigma}$ has the same dimensions as the original matrix $\boldsymbol{A}$, and has zero elements everywhere but the diagonal, with the *singular values* of $\boldsymbol{A}$ running down the diagonal.

The main advantage of SVD is it's great numerical stability, which makes it ideally suitable for dealing with large systems. The pseudoinverse can be computed using the relation

$$\boldsymbol{V}\boldsymbol{\Sigma}^{+}\boldsymbol{U}^T$$

where $\boldsymbol{\Sigma}^{+}$ is constructed by transposing $\boldsymbol{\Sigma}$ and inverting all non-zero entries. The derivation of this relationship can be found in most standard linear algebra text [45]. Referring to Eq. (3), we can compute $\boldsymbol{L}$ using the relation:

$$\boldsymbol{L}=\boldsymbol{N}\boldsymbol{V}\boldsymbol{\Sigma}^{+}\boldsymbol{U}^T$$

From $\boldsymbol{L}$ we can obtain $\boldsymbol{L}_0$ and hence $\boldsymbol{\Gamma}$.

5. Computing the null space

Nullspace determination is an important computational aspect of biochemical modelling and arises frequently in structural analysis computations. However, detailed descriptions of the algorithms (particularly the implementation details) for this computation are not commonly found in text books on linear algebra. Specific implementation details are also not commonly covered in detail in the research literature, one of the few articles which does describe an implementation can be found in [43]. In this section we wish to discuss several approaches which can be employed to compute the null space. The first is simple to understand and is relatively easy to implement in software.

5.1. By Gaussian elimination

Given an equation:

$$\boldsymbol{A}\boldsymbol{x}=0$$

we wish to find the set of solutions that satisfy this equation. One of the standard techniques for solving this equation is to reduce $\boldsymbol{A}$ to its reduced row echelon form. This operation will not change the null space and hence the problem is simplified to that of finding the null space of the reduced row echelon. Thus, we now wish to solve an equation of the form:

$$\begin{bmatrix}\boldsymbol{I} & \boldsymbol{M}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\boldsymbol{x}=0$$

This form is much easier to solve. A set of solutions is clearly given by

$$\begin{bmatrix}\boldsymbol{I} & \boldsymbol{M}\\ \boldsymbol{0} & \boldsymbol{0}\end{bmatrix}\begin{bmatrix}-\boldsymbol{M}\\ \boldsymbol{I}_k\end{bmatrix}=\boldsymbol{0}$$

where k equals the column dimension of $\boldsymbol{M}$ (which

is the dimension of the nullspace of $\boldsymbol{A}$). Each column vector in the matrix $[-M \quad I_k]^T$ is a solution to $\boldsymbol{Ax}=0$. Moreover, the rows in the non-zero partition of the row echelon are guaranteed to be linear independent, therefore, the columns of $[-M \quad I_k]^T$ are likewise linearly independent, hence they span the null space. In summary we can very easily determine the null space of a matrix by reducing the matrix to row echelon form, the non-zero rows of the echelon will form the required null space.

5.2. By singular value decomposition

For very large matrices the above method may not be suitable unless careful pivoting and/or scaling in carried out; ill-conditioning may render the row echelon extremely error sensitive. For large matrices, the method of choice is to use SVD or singular value decomposition. Indeed, a basis for the null space is automatically generated during the decomposition; they appear as columns of the $\boldsymbol{V}$ matrix. However, the vectors generated by SVD will not be as 'clean' as those generated by row reduction. Rather than integer elements as we might expect, SVD tends to generate odd looking fractions, albeit correct odd looking fractions! In general this is not a problem as in many instances the solutions will be used internally by software and, therefore, need not be viewed by the discerning human eye. However, there will be occasions when a 'clean' set of vectors may be required. In these cases one may form a matrix from the vectors derived through SVD and reduce the transpose of the vectors to a reduced echelon form. In many cases this will generate a set of vectors, which will have a more obvious physical meaning, the process of reduction will not, by its nature, affect the space spanned by the vectors, it only changes the basis, which is the intention.

6. Rational arithmetic

A final topic worth considering is the use of rational arithmetic when manipulating the matrices, in which all entries in the matrix are represented as ratios of integers. This technique is employed frequently by some of the commercial algebra packages, such as Mathematical or Maple. An academic software tool, called emPath (available at http: //bms-mudshark.brookes.ac.uk) was developed by John Woods and took advantage of rational arithmetic to compute elementary modes. The advantage to using rational numbers is that the results of any computation are exact and thus avoids issues such as rounding errors that accumulate during a matrix reduction. The problem of course is that all the matrix manipulation algorithms have to be rewritten to accommodate this type of arithmetic. SBML [22], the biochemical exchange standard has facilities, which allows stoichiometies to be represented as rational numbers.

7. Software

The use of software in stoichiometric analysis is not only useful but necessary. It is simply not possible, except in relatively trivial cases, to carry out the computations by hand. Probably one of the best packages available for stoichiometric analysis is METATOOL [29]. This is a C based packaged originally conceived by Stefan Schuster and developed by Thomas Pfeiffer and more recently by Juan Carlos Nuno and Ferdinand Moldenhauer (http: //www.bioinf.mdc-berlin.de/projects/metabolic/metatool/). It easily runs on Linux or Windows or for that matter any platform, which can compile standard C code. METATOOL generates a multitude of information, including but not exclusively, the null space of the stoichiometry matrix, conservation relations, and what METATOOL was specifically designed to generate, elementary modes. Generating elementary modes is a non-trivial exercise and other packages, such as the interactive simulator, Jarnac [36] employ METATOOL for this task.

Hagen Hoepfner and Matthias Lange at Magdeburg in Germany created a Web site which interfaces to METATOOL, called phpMetatool (http: //www-bm.ipk-gatersleben.de/development/php/phpMetatool).

There was also an effort by Klaus Mauch at Stuttgart to implement the algorithms used by METATOOL using Web based Java but the work appears no longer to be available at the University

site and instead is now part of a commercial enterprise (http://www.insilico-biotechnology.com/insilico_en.html). This package is extremely sophisticated and offers probably the most stable robust implementation for computing structural properties, especially for large systems.

FluxAnalyzer [25] is a recently developed MATLAB based application which offers extensive capabilities for computing the structural capabilities of reaction networks. Although the main emphasis of FluxAnalyzer is on elementary and extreme network analysis, the program will also generate conservation laws. FluxAnalyser also offers a friendly user interface which should encourage it's widespread use.

Another package, recently released, which can calculate a variety of metrics associated with the stoichiometry matrix is ScrumPy, available at http://mudshark.brookes.ac.uk/ScrumPy. This package, developed by Mark Poolman and David Fell takes an interesting approach; ScrumPy (the name being a derivation of one of the earliest PC based simulators, SCAMP, [37]) is a module for the highly popular scripting tool, Python. This means that the analysis can be carried out interactively making 'what-if' type questions much easier to carry out. At the time of writing, ScrumPy only runs under Linux, however, a Windows version is promised at some point in the future. ScrumPy also has facilities to carry out dynamic simulation and curve fitting. As with METATOOL, full source code is supplied with ScrumPy.

A related tool to ScrumPy is Pysces developed by Brett Olivier and Jannie Hofmeyr (http://www.rmsb.u-bordeaux2.fr/BTK/abstracts/long-47-Olivier.pdf). Although still under development this is also a simulation and analysis tool build around Python.

Jarnac [36], a windows package, also has extensive support for stoichiometric analysis and uses METATOOL to carry out the elementary modes calculations. Jarnac also has built in support for computing all the matrices discussed in this review and includes an extensive library of matrix functions (http://www.sys-bio.org). In addition, it has SVD support for dealing with large models. Like ScrumPy, Jarnac is an interactive tool suitable for quick what-if type queries. The package also has extensive simulation and analysis capabilities. Full source is also available.

Gepasi [26], a popular windows based package also has capabilities for compute conservation cycles and elementary modes. Unlike the other software packages, Gepasi is a pure GUI based package and as a result is ideal for casual users.

JigCell (http://gnida.cs.vt.edu/cellcyclepse/) is a recently developed Java based systems biology simulator. Like Gepasi, jigCell uses a GUI based interface but unlike Gepasi, it uses a spreadsheet metaphor for inputting and managing models. JigCell is able to compute conservation laws and attempts to generate the most physically sensible laws from the reaction scheme.

A commercial package well-worth mentioning is SimPheny [30]. This package, developed by Genomatica, offers an integrated environment for analyzing genome data and incorporates many analyses that are focused on the analysis of network topology.

All the packages except SimPheny, generate textual output in one form or another. However, it would be desirable to be able to display the results of stoichiometric analyses visually on a network diagram. With the development of SBW (Systems Biology Workbench) this is now easier to achieve. The SBW/SBML team lead by Hiroaki Kitano has developed a means by which software tools for the systems biology community can be combined in a modular like fashion. For example, the SBW team has developed a stoichiometric visualizer tool. This is made up of three components, METATOOL, an SBW wrapper application around METATOOL and JDesigner, (a SBW network visualization tool). A user 'draws' the network on a display canvas using JDesigner. When ready the user instructs JDesigner to generate SBML [22] for the network, which is sent automatically to the METATOOL interface module. The METATOOL module in turn generates the appropriate input files for METATOOL and executes METATOOL. The output generated from METATOOL is then passed back via the METATOOL interface module to JDesigner so that the individual elementary modes or other stoichiometric information can be visualized. Fig. 4 illustrates a conservation law on display by JDesigner. A selection list is also

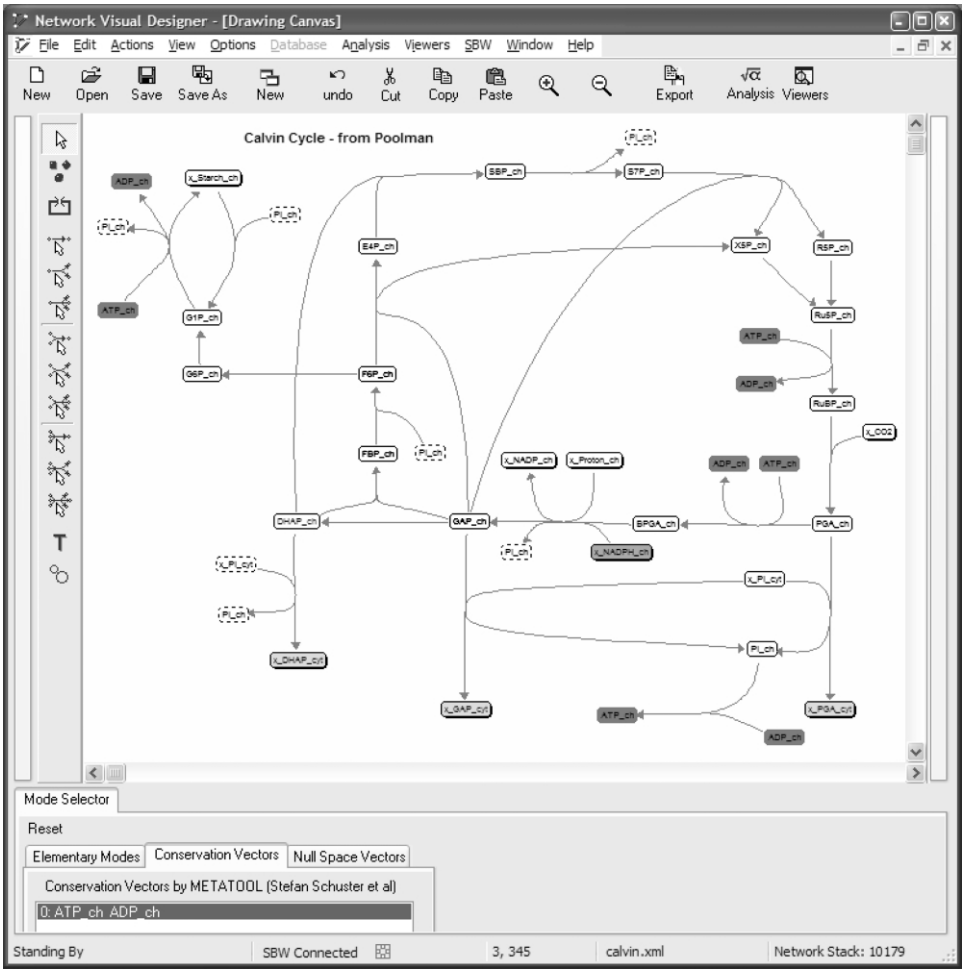


Fig. 4. JDesigner with METATOOL Integration via SBW, with conservation species identified and highlighted in dark shade.

generated to allow the user to scroll through each conservation law one at a time. The package also permits users to highlight elementary modes.

7.1. Software issues relating to large systems

Large systems pose a significant problem in stoichiometric analysis. Large stoichiometric matrices are sparse, that is only a small percentage of the matrix has non-zero entries. A stoichiometry matrix for an *E. coli* model comprising of 544 species and 739 reactions was obtained from Palsson's web site at http://gcrg.ucsd.edu/organisms by translating the Microsoft Excel format model into SBML [23] using Jarnac [36]. This matrix has 402,016 elements and occupies over 3 MB of memory. The interesting statistic is that less than 1% of the matrix has non-zero elements. If we were to store only the non-zero elements, the matrix would only use approximately 20 K of space, a considerable saving. Matrices, which have very few non-zero elements are termed sparse and considerable expertise now exists in the physics, numerical analysis and related communities for dealing with large sparse matrices [33]. In future, software development of biochemical simulators should make a serious effort to incorporate sparse matrix support since the need to analyze large

models will undoubtedly increase as larger network maps become more readily available through genomic studies. This need for greater efficiency is becoming acute. Consider that the authors of FluxAnalyzer reported the elementary mode analysis of a medium size model of *E. coli* comprising of 89 metabolites and 110 reactions. They reported the generation of 507,632 elementary modes, which took roughly 50 h on a 1.0 Hz Pentium PC. However, the authors do not indicate how the algorithms were implemented, whether in raw MATLAB m-files (which are notoriously slow) or as compiled m-files or some external C library. A recent proceedings paper by Samatova et al. [34] describe a novel [34] approach involving the parallelization of the calculations which should greatly improve performance.

8. Conclusion

This review highlights some of the issues facing developers of simulation and analysis software for systems biology. In particular the problem of detecting and determining the conservation laws in an arbitrary biochemical network is discussed. For small models with only a few variables, determining the conservation laws by hand is a practical proposition. For large systems, the same computation is simply not practical. Depending on the type of question being asked, deriving the conservation laws can be crucial. Most if not all modern simulation packages available to the systems biology community can carry out the necessary derivation, moreover, almost all the packages will also reduce the differential equation set according the number of conservation laws detected.

In this review we highlight only one small area of analysis, the detection and derivation of conservation laws. While the theory behind conservation analysis is reasonably understood, implementation issues in computer software is rarely discussed and we hope that this review has highlighted some of the issues. Much still remains to be done, for example more efficient and robust algorithms for determining physically interpretable conservation laws and particularly acute is the development of software and algorithms to deal with very large systems, including the use of sparse matrices and novel visualization approaches.

Acknowledgments

The authors would like to thank David Fell, Jannie Hofmeyr and Stefan Schuster for numerous discussions over the years in relation to the computation of conservation laws in reaction networks. The authors would also like to acknowledge the reviewers who comments greatly improved this article.

References

[1] R. Aris, Prolegomena to the rational analysis of systems of chemical reactions, Arch. Rational Mech. Anal. 19 (1965) 81–99.
[2] B.M. Bakker, P.A.M. Michels, F.R. Opperdoes, H. Westerhoff, What controls glycloysis in bloodstream form trypanosoma brucei, J. Biol. Chem. 274 (1999) 14 551–14 559.
[3] B.M. Bakker, H.V. Westerhoff, F.R. Opperdoes, P.A.M. Michels, Metabolic control analysis of glycolysis in trypanosomes as an approach to improve selectivity and effectiveness of drugs, Mol. Biochem. Parasitol. 106 (2000) 1–10.
[4] BioSPICE (2001), Home page. The BioSPICE Development Project, http://www.biospice.org/.
[5] C. Chassagnole, B. Ras, E. Quentin, D.A. Fell, J.-P. Mazat, An integrated study of threonine-pathway enzyme kinetics in *Escherichia coli*, Biochem. J. 356 (2001) 415–423.
[6] B.L. Clarke, Stability of Complex Reaction Networks, Vol. 42, Wiley, New York, 1980, of Adv. Chem. Phys.
[7] A. Cornish-Bowden, R. Eisenthal, Computer simulation as a tool for studying metabolism and drug design, in: A. Cornish-Bowden, M.L. Cardenas (Eds.), Technological and Medical Implications of Metabolic Control Analysis, Kluwer Academic Publishers, The Netherlands, Dordrecht, 2000, pp. 165–172.
[8] A. Cornish-Bowden, J.-H.S. Hofmeyr, The role of stoichiometric analysis in studies of metabolism: an example, J. Theor. Biol. 216 (2002) 179–191.
[9] J. Edwards, B. Palsson, The *Escherichia coli* mg1655 in silico metabolic genotype: its definition, characteristics and capabilities, PNAS 97 (2000) 5528–5533.
[10] Entelos, Company page (2001), Biosimulation for in silico drug discovery and development, http://www.entelos.com/.
[11] I. Famili, B.O. Palsson, The convex basis of the left null space of the stoichiometric matrix leads to the definition of metabolically meaningful pools, Biophys. J. 85 (2003) 16–26.

[12] M. Feinberg, Necessary and suffficient conditions for detailed balancing in mass action systems of arbitrary complexity, Chem. Eng. Sci. 44 (1989) 1819–1827.
[13] D. Fell, Understanding the Control of Metabolism, Portland Press, London, 1997.
[14] D.A. Fell, H.M. Sauro, Metabolic control analysis: additional relationships between elasticities and control coefficients, Eur. J. Biochem. 148 (1985) 555–561.
[15] GNS (2003), Gene network sciences: company web page. Gene Network Sciences accelerates the drug discovery process by creating dynamic computer models of living cells, vww.gnsbiotech.com.
[16] W.J. Hedley, N.R. Melanie, D. Bullivant, A. Cuellar, Y. Ge, M. Grehlinger, et al. (2001), CellML specification. Available via the World Wide Web at http: //www. cellml. org.
[17] R. Heinrich, S. Schuster, The Regulation of Cellular Systems, Chapman and Hall, 1996.
[18] J.-H.S. Hofmeyr, Steady state modelling of metabolic pathways: a guide for the prospective simulator, Comp. Appl. Biosci. 2 (1986) 5–11.
[19] J.-H.S. Hofmeyr, H. Kacser, K.J. van der Merwe, Metabolic control analysis of moiety-conserved cycles, Eur. J. Biochem. 155 (1986) 631–641.
[20] H. Holstein, C. Greenshaw, A numerical treatment of metabolic control models, in: H.V. Westerhoff (Ed.), Biothermokinetics, Intercept Ltd, Andover, 1994, pp. 293–299.
[21] F. Horn, R. Jackson, General mass action kinetics, Arch. Rational Mech. Anal. 47 (1972) 81–116.
[22] M. Hucka, A. Finney, H.M. Sauro, H. Bolouri (2001), Systems Biology Markup Language (SBML) Level 1: structures and facilities for basic model definitions. Available via the world wide web at http://www.cds-caltech.edu/erato.
[23] M. Hucka, A. Finney, H.M. Sauro, H. Bolouri, J.C. Doyle, H. Kitano, et al., The systems biology markup language (SBML): a medium for representation and exchange of biochemical network models, Bioinformatics 19 (2003) 524–531.
[24] B.N. Kholodenko, H.M. Sauro, H.V. Westerhoff, Control by enzymes, coenzymes and conserved moieties, a generalisation of the connectivity theorem of metabolic control analysis, Eur. J. Biochem. 225 (1994) 179–186.
[25] S. Klamt, J. Stelling, M.G. M, E.D. Gilles, Fluxanalyzer: exploring structure, pathways and flux distributions in metabolic networks on interactive flux maps, Bioinformatics 19 (2003) 261–269.
[26] P. Mendes, Gepasi: a software package for modelling the dynamics, steady states and control of biochemical and other systems, Comput. Appl. Biosci. 9 (1993) 563–571.
[27] J.A. Papin, N.D. Price, S.J. Wiback, D.A. Fell, B.O. Palsson, Metabolic pathways in the post-genome era, Trends Biochem. Sci. 28 (2003) 250–258.
[28] D.J.M. Park, The hierarchical structure of metabolic networks and the construction of efficient metabolic simulators, J. Theor. Biol. 46 (1974) 31–74.
[29] T. Pfeiffer, I. Sanchez-Valdenebro, J.C. Nuno, F. Montero, S. Schuster, Metatool: for studying metabolic networks, Bioinformatics 15 (1999) 251–257.
[30] N.D. Price, J.A. Papin, C.H. Schilling, B.O. Palsson, Genome-scale microbial in silico models: the constraints-based approach, Trends Biotechnol. 21 (2003) 162–169.
[31] C. Reder, Metabolic control theory: a structural approach, J. Theor. Biol. 135 (1988) 175–201.
[32] J.G. Reich, E.E. Selkov, Energy Metabolism of the Cell, Academic Press, London, 1981.
[33] Y. Saad, Iterative Methods for Sparse Linear Systems, 2nd ed, Society for Industrial and Applied Mathematics, 2003.
[34] N.F. Samatova, A. Geist, G. Ostrouchov, A. Melechko (2003), Parallel out-of-core algorithm for genome-scale enumeration of metabolic systemic pathways. in: S. Alurur, D.A. Bader (Eds.), Second IEEE International Workshop on High Performance Computational Biology, HiCOMB 2003.
[35] H.M. Sauro, Moiety-conserved cycles and metabolic control analysis: problems in sequestration and metabolic channelling, Biosystems 33 (1994) 15–28.
[36] H.M. Sauro, Jarnac: a system for interactive metabolic analysis. in: J.-H.S. Hofmeyr, J.M. Rohwer, J.L. Snoep (Eds.), Animating the Cellular Map: Proceedings of the 9th International Meeting on BioThermoKinetics, Stellenbosch University Press, 2000.
[37] H.M. Sauro, D.A. Fell, Scamp: a metabolic simulator and control analysis program, Math. Comput. Modelling 15 (1991) 15–28.
[38] C.H. Schilling, D. Letscher, B.O. Palsson, Theory for the systemic definition of metabolic pathways and their use in interpreting metabolic function from a pathway-oriented perspective, J. Theor. Biol. 203 (2000) 229–248.
[39] S. Schuster, D.A. Fell, T. Dandekar, A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks, Nature Biotechnol. 18 (2000) 326–332.
[40] S. Schuster, C. Hilgetag, What information about the conserved-moiety structure of chemical reaction systems can be derived from their stoichiometry?, J. Phys. Chem. 99 (1995) 8017–8023.
[41] S. Schuster, C. Hilgetag, J. Woods, D. Fell, Reaction routes in biochemical reaction systems: algebraic properties, validated calculation procedure and example from nucleotide metabolism, J. Math. Biol. 45 (2002) 153–181.
[42] S. Schuster, T. Hofer, Determining all extreme semi-positive conservation relations in chemical reaction systems: a test criterion for conservativity, J. Chem. Soc. Faraday Trans. 87 (1991) 2561–2566.

[43] S. Schuster, R. Schuster, Detecting strictly detailed balanced subnetworks in open chemical reaction networks, J. Math. Chem. 6 (1991) 17–40.
[44] G. Stephanopoulos, A. Aristidou, J. Nielsen, Metabolic Engineering, Academic Press, 1998.
[45] G. Strang, Linear Algebra and its Applications, 3rd ed, International Thomson Publishing, 1988.
[46] D. Visser, J. Heijnen, The mathematics of metabolic control analysis revisited, Metabolic Eng. 4 (2002) 114–123.
[47] E.O. Voit, Computational Analysis of Biochemical Systems, Cambridge University Press, 2000.