FluxSimulator: An R package to simulate isotopomer distributions in metabolic networks

Binsl, T.W.; Mullen, K.M.; van Stokkum, I.H.M.; Heringa, J.; van Beek, J.H.G.M.

published in
Journal of Statistical Software
2007

DOI (link to publisher)
10.18637/jss.v018.i07

document version
Publisher's PDF, also known as Version of record

Journal of Statistical Software

January 2007, Volume 18, Issue 7. http://www.jstatsoft.org/

FluxSimulator: An R Package to Simulate Isotopomer Distributions in Metabolic Networks

Thomas W. Binsl
Vrije Universiteit Amsterdam

Katharine M. Mullen
Vrije Universiteit Amsterdam

Ivo H. M. van Stokkum
Vrije Universiteit Amsterdam

Jaap Heringa
Vrije Universiteit Amsterdam

Johannes H. G. M. van Beek
Vrije Universiteit Medical Centre Amsterdam

Abstract

The representation of biochemical knowledge in terms of fluxes (transformation rates) in a metabolic network is often a crucial step in the development of new drugs and efficient bioreactors. Mass spectroscopy (MS) and nuclear magnetic resonance spectroscopy (NMRS) in combination with ^{13}C labeled substrates are experimental techniques resulting in data that may be used to quantify fluxes in the metabolic network underlying a process. The massive amount of data generated by spectroscopic experiments increasingly requires software which models the dynamics of the underlying biological system.

In this work we present an approach to handle isotopomer distributions in metabolic networks using an object-oriented programming approach, implemented using S4 classes in R. The developed package is called **FluxSimulator** and provides a user friendly interface to specify the topological information of the metabolic network as well as carbon atom transitions in plain text files. The package automatically derives the mathematical representation of the formulated network, and assembles a set of ordinary differential equations (ODEs) describing the change of each isotopomer pool over time. These ODEs are subsequently solved numerically.

In a case study **FluxSimulator** was applied to an example network. Our results indicate that the package is able to reproduce exact changes in isotopomer compositions of the metabolite pools over time at given flux rates.

Keywords: metabolism, flux, isotopomer labeling, ODE, R, MS, NMRS, systems biology.

1. Introduction

Dedicated drug development (Böhm *et al.* 2002; Leach and Gillet 2003) and the design of new efficient bioreactors (Dunn *et al.* 2003) is a great challenge for the pharmaceutical and biotechnological industries. Progress often depends on the exact knowledge of biochemical reactions between chemical compounds. As these reactions occur in the metabolism of patients or microorganisms the reactions are termed metabolic reactions and the chemical compounds metabolites. Both can be represented via so called metabolic network models which allow for their quantitative investigation. We will provide two case scenarios to highlight this, based on the small sample network given in Figure 1.

In the example network each reaction is catalyzed by enzyme $E1$ or $E2$, which transform metabolite A into metabolite B or C, respectively. Such a network may model a scenario in which metabolite B is a desired product to be accumulated during a fermentation. Then an inhibition of $E2$ or knocking out the gene expressing it would cause a limitation in the production of C and could lead to a higher production rate of B.

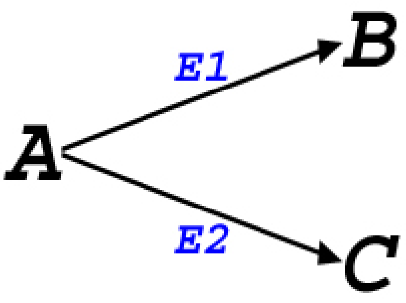


Figure 1: A simple part of an hypothetical metabolic network. Metabolite A is transformed via two enzymes into two different metabolites named B and C. The reaction from A to B is catalyzed by enzyme $E1$ and the reaction from A to C by enzyme $E2$.

Alternatively, such a network model may represent a scenario in which it is known that the absence of metabolite C causes a disease. Again the knowledge provided by the metabolic network helps to understand why this metabolite is produced at a lesser rate or not at all. For instance, this could be caused by the absence or the malfunction of enzyme $E2$. Thus, an artificial supply of enzyme $E2$ or, even better, a supply with an optimized enzyme $E2'$ are two possible solutions.

Much research effort has been directed at experimental techniques providing insight into metabolic networks and the quantification of the exchange fluxes between metabolites, since, as the example just considered demonstrates, such models may provide insight into the underlying system. All developed techniques base on the same approach, that is the labeling of one or several metabolites. This is achieved by replacing atoms within the metabolites by one of their isotopes. These isotopes can be detected easily by spectroscopic measuring techniques due to their different number of neutrons. Hence, the metabolites can be observed during the experiment.

Early experiments (Sauer *et al.* 1970) included radioactive ^{14}C atom labeling to determine the flux of intracellular metabolites. These experiments provide information of interest but suffered from several disadvantages, including the need for isolation of the intermediate metabolites, their purification, chemical degradation, and the risk entailed by radioactive radiation. In order to address such problems, labeling techniques using ^{13}C isotopes in combination with nuclear magnetic resonance spectroscopy (NMRS) have been developed over the course of the last thirty years (Chance *et al.* 1983). The technique provides similar information as yielded by radioactive labeling with ^{14}C but does not suffer from its disadvantages.

In both experimental techniques, labeled and unlabeled carbon atoms move from a particular position in a substrate molecule to a particular position in the product of the reaction. The set of all transitions between carbon atoms in substrates and products is termed the carbon transition network (CTN). Metabolic networks consists of several metabolite pools, where each particular pool is the sum of all metabolites belonging to the same chemical species. Within a carbon transition network these pools are additionally subdivided into particular fractions. This is due to the 2^n different labeling states that can occur in a metabolite consisting of n carbons. Each of these labeling states is termed an isotopomer and forms a subpool within its particular metabolite pool.

Hence, to analyze ^{13}C NMRS experiments, software to simulate and interpret these carbon transition networks is necessary. Several approaches have been implemented previously. For instance, van Beek *et al.* (1998) and van Beek *et al.* (1999), developed and implemented computer programs in FORTRAN to simulate the dynamics of various metabolic network models. Such FORTRAN programs achieve accurate simulation results but suffer from limited flexibility and are error-prone during the manual implementation of the ODEs. This is due to the fact that each model has to be hard coded in a subroutine, which is linked to a simulation computer program.

Another approach is using existing software packages like **Berkeley Madonna** (Macey *et al.* 2000) or **JSIM** (Raymond *et al.* 2003) to simulate dynamic systems. Here again, the programs require hard-coding of the model in a specially defined mathematical language.

Yet another approach by Wiechert *et al.* (1995); Wiechert and de Graaf (1997); Wiechert *et al.* (1997, 1999); Wiechert (2001) consists of the algorithms to analyze and simulate ^{13}C labeling experiments. Wiechert *et al.* (2001) incorporated these developments into a software package (Wiechert *et al.* 2001).

This package considers the underlying network to be in metabolic and isotopomeric steady state, i.e., it assumes that the concentrations of the particular metabolites and their isotopomers do not change over time. As new experimental approaches go beyond these assumptions, programs must be developed to handle non-steady state systems. Furthermore, the package is written in C++ and must be used on a Linux platform.

To address the above limitations we have developed a package in the high level language R (R Development Core Team 2006). The package is designed to simulate the carbon atom transitions occurring in a metabolic network (Carbon Transition Network, CTN) over time. In addition to algorithms solving ODEs efficiently, R provides features for parameter estimation and statistics that will facilitate the extension of the package in the future. Furthermore, R is cross-platform and can be run on all major operating systems like Linux, Windows and MacOS. The developed R package is called **FluxSimulator** and can be used without any skills in programming or handling ordinary differential equations (ODEs).

An example of a CTN, its mathematical representation, the derivation from the input files as well as an exact specification of the input files is given in the Section 2. Additionally, we performed an example simulation and compared the results to a simulation done with **Berkeley Madonna**. This comparison is analyzed and discussed in Section 3. Final conclusions and an outlook to future work are given in Section 4.

2. Methods

This section introduces an example CTN. It is used to explain the automatic derivation of the mathematical representation from the input files which are also specified in detail. Additionally, an application scenario is presented.

2.1. CTNs And their mathematical representation

A CTN is a special type of metabolic network. Metabolic networks are usually given by their metabolites and the exchange fluxes that occur between them (Figure 2, upper network). The CTN has an additional property. For each metabolite present in the network, the transitions of its carbon atoms to the produced metabolites are specified (Figure 2, lower network). Both networks shown in Figure 2 are artificially designed, but are inspired by the network given in Wiechert *et al.* (1999), which includes almost all basic situations that can occur in a metabolic or carbon transition network.

Due to the specified carbon transitions the exact path of a labeled carbon through the network is traceable. However, this causes problems with the calculation of the dynamic behavior.

While ordinary metabolic networks can be represented by a number of ODEs that is equal to the number of changing metabolites occurring in the network, this is not possible for CTNs. This is due to the large number of different isotopomers that can occur within a single metabolite pool, each of which need to be represented by a pool themselves. Thus, the number of ODEs necessary to represent a single metabolite is equal to 2^n, where n is the number of its carbon atoms (Figure 3).

Accordingly, the metabolic network given on the top in Figure 2 can be represented by the following seven ODEs specified in Equations 1 to 7.

$$\frac{d[B]}{dt} = v_1 + v_{3b} + v_{7f} - (v_2 + v_{3f} + v_{7b}) \quad (1)$$

$$\frac{d[C]}{dt} = v_2 - v_8 \quad (2)$$

$$\frac{d[D]}{dt} = v_{3f} - (v_{3b} + v_4) \quad (3)$$

$$\frac{d[E]}{dt} = v_4 + v_5 - v_9 \quad (4)$$

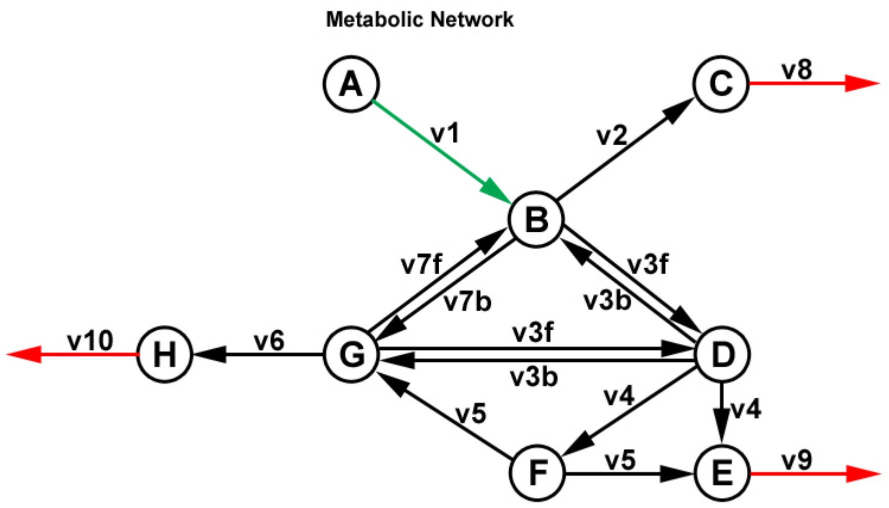
$$\frac{d[F]}{dt} = v_4 - v_5 \quad (5)$$

$$\frac{d[G]}{dt} = v_{7b} + v_{3b} + v_5 - (v_{3f} + v_6 + v_{7f}) \quad (6)$$

$$\frac{d[H]}{dt} = v_6 - v_{10} \quad (7)$$

Since metabolite A represents the entry point of the network, its concentration derives from the experimental protocol. To simplify matters, we consider it to be constant after it has been changed from natural abundance ^{13}C levels at $t = 0$. All remaining changes of metabolite concentrations are given as the sum of the incoming fluxes minus the sum of outgoing fluxes. Metabolite concentrations are represented by the metabolite names in squared brackets. This reflects the mass balance of the metabolites.

Each flux is given in $\left[\frac{\mu mol}{g*s}\right]$ but how a particular flux is defined depends on the underlying reaction. Biochemical reactions not catalyzed by enzymes are given as the product of a rate



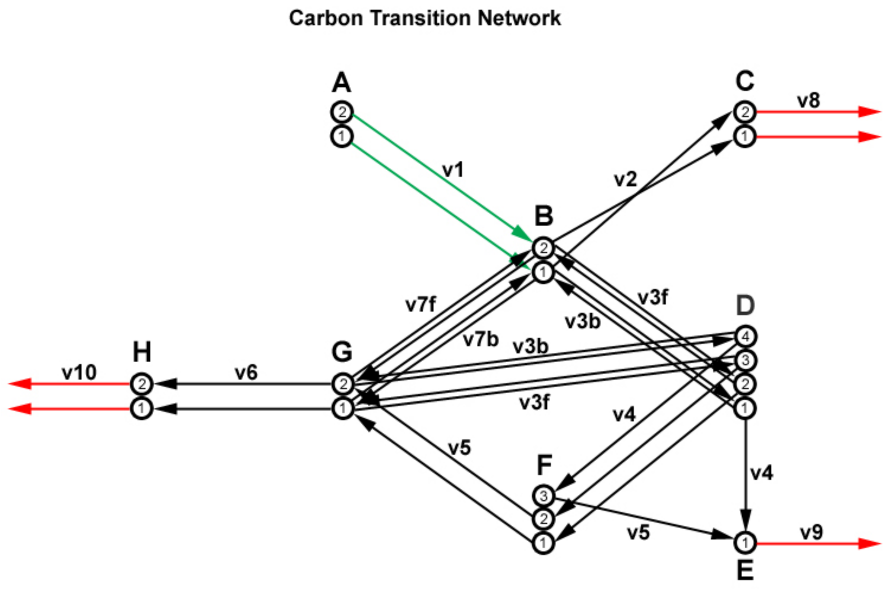
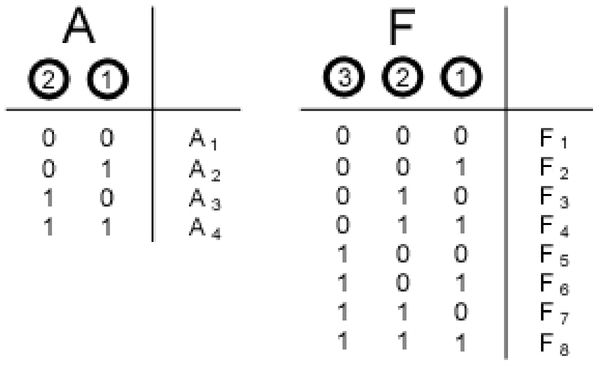


Figure 2: A metabolic network (top) with its corresponding carbon transition network (bottom). Green colored arrows represent fluxes entering the network, while red colored arrows indicate exiting fluxes. Black colored arrows refer to internal fluxes. Note metabolite A which represents the entry point of the network and therefore requires special handling throughout the following simulation. Flux v_{3f} indicates a bimolecular reaction on the substrate side that combines the two metabolites B and G into a new metabolite D. Flux v_{3b} indicates a bimolecular reaction on the product side, splitting metabolite D into the metabolites B and G. The same holds for flux v_4 splitting metabolite D into metabolites E and F, and flux v_5 splitting metabolite F into E and G. Hence, v_{3f}, v_{3b}, v_4 and v_5 are given twice to denote the coupled reactions.



A ②	A ①	
0	0	A_1
0	1	A_2
1	0	A_3
1	1	A_4

F ③	F ②	F ①	
0	0	0	F_1
0	0	1	F_2
0	1	0	F_3
0	1	1	F_4
1	0	0	F_5
1	0	1	F_6
1	1	0	F_7
1	1	1	F_8

Figure 3: Representation of the isotopomer pools occurring in a two carbon metabolite A and a three carbon metabolite F. The numbers highlighted by a circle represent the particular carbon atoms one, two and three. The binary encoding represents the possible labeling states. A "0" represents a non labeled carbon while a "1" represents a labeled carbon. Finally, the indices at the metabolite names to the right of the binary encoding are created by the decimal representation of the binary code plus one. The indices represent the particular isotopomers.

constant k times the concentration of the metabolite at the origin of the reaction (Equation 8). In contrast, assuming an enzyme-catalyzed reaction, flux v_1 could for instance be modeled by Michaelis-Menten kinetics (Michaelis and Menten 1913) as given in Equation 9, where v_{max} indicates the maximum reaction rate and K_M the Michaelis-Menten constant.

$$v_1 = k_1 \left[\frac{1}{s}\right] * [A] \left[\frac{\mu mol}{g}\right] \tag{8}$$

$$v_1 = v_{max} \left[\frac{\mu mol}{g * s}\right] * \frac{[A]\left[\frac{\mu mol}{g}\right]}{K_M \left[\frac{\mu mol}{g}\right] + [A]\left[\frac{\mu mol}{g}\right]} \tag{9}$$

In contrast to the metabolic network the CTN does not deal with simple metabolites but with all isotopomer fractions of these metabolites. Hence, the example CTN has to be represented by 42 ODEs, each referring to an isotopomer fraction. The ODEs are given in indexed mode by Equation 10 to 16. The explanation of the indices is given in Figure 3.

$$\begin{aligned}[B] \cdot \frac{db_i}{dt} &= (v_1 \cdot a_i + v_{3b} \cdot (d_i + d_{i+4} + d_{i+8} + d_{i+12}) + v_{7f} \cdot g_i \\ &\quad - (v_2 + v_{3f} + v_{7b}) \cdot b_i) \quad (i = 1, ..., 4)\end{aligned} \tag{10}$$

$$[C] \cdot \frac{dc_i}{dt} = \begin{cases} v_2 \cdot b_i - v_8 \cdot c_i & \text{if } (i = 1, 4) \\ v_2 \cdot b_{i+1} - v_8 \cdot c_i & \text{if } (i = 2) \\ v_2 \cdot b_{i-1} - v_8 \cdot c_i & \text{if } (i = 3) \end{cases} \tag{11}$$

$$[D] \cdot \frac{dd_i}{dt} = \begin{cases} (v_{3f} \cdot b_i \cdot g_1 - (v_{3b} + v_4) \cdot d_i) & \text{if } (i = 1, ..., 4) \\ (v_{3f} \cdot b_{i-4} \cdot g_2 - (v_{3b} + v_4) \cdot d_i) & \text{if } (i = 5, ..., 8) \\ (v_{3f} \cdot b_{i-8} \cdot g_3 - (v_{3b} + v_4) \cdot d_i) & \text{if } (i = 9, ..., 12) \\ (v_{3f} \cdot b_{i-12} \cdot g_4 - (v_{3b} + v_4) \cdot d_i) & \text{if } (i = 13, ..., 16) \end{cases} \tag{12}$$

$$[E] \cdot \frac{de_i}{dt} = \begin{cases} (v_4 \cdot \sum_{j=1}^{8} d_{2j-1} + v_5 \cdot \sum_{j=1}^{4} f_j - v_9 \cdot e_i) & \text{if } (i = 1) \\ (v_4 \cdot \sum_{j=1}^{8} d_{2j} + v_5 \cdot \sum_{j=4}^{8} f_j - v_9 \cdot e_i) & \text{if } (i = 2) \end{cases} \tag{13}$$

$$[F] \cdot \frac{df_i}{dt} = (v_4 \cdot \sum_{j=2i-1}^{2i} d_j - v_5 \cdot f_i) \quad (i = 1, ..., 8) \tag{14}$$

$$[G] \cdot \frac{dg_i}{dt} = \begin{cases} (v_{7b} \cdot b_i + \sum_{j=1}^{4} v_{3b} \cdot d_j + v_5 \cdot (f_i + f_{i+4}) & \\ \quad -(v_{3f} + v_6 + v_{7f}) \cdot g_i) & \text{if } (i = 1) \\ (v_{7b} \cdot b_i + \sum_{j=5}^{8} v_{3b} \cdot d_j + v_5 \cdot (f_i + f_{i+4}) & \\ \quad -(v_{3f} + v_6 + v_{7f}) \cdot g_i) & \text{if } (i = 2) \\ (v_{7b} \cdot b_i + \sum_{j=9}^{12} v_{3b} \cdot d_j + v_5 \cdot (f_i + f_{i+4}) & \\ \quad -(v_{3f} + v_6 + v_{7f}) \cdot g_i) & \text{if } (i = 3) \\ (v_{7b} \cdot b_i + \sum_{j=13}^{16} v_{3b} \cdot d_j + v_5 \cdot (f_i + f_{i+4}) & \\ \quad -(v_{3f} + v_6 + v_{7f}) \cdot g_i) & \text{if } (i = 4) \end{cases} \tag{15}$$

$$[H] \cdot \frac{dh_i}{dt} = (v_6 \cdot g_i - v_{10} \cdot h_i) \quad (i = 1, ..., 4) \tag{16}$$

Each isotopomer fraction is represented by the metabolite name in lower case letter combined with an index, e.g. d_i or de_i. This index corresponds to the carbon atoms labeled in the particular isotopomer fraction (see explanation in Figure 3). The fluxes are equal to those used in the previous Equations 1 to 7. From Equations 10 to 16 one can see that the time scale of change of the isotopomer fractions can be derived by dividing the incoming fluxes and outgoing fluxes (right side of the equations) by the concentration of the receiving metabolite pool (concentration on the left side of the equations). As metabolite A is the entry of the network its isotopomers are prescribed and are again not considered in the equations above.

With large numbers of ODEs two problems occur. The first problem is the increased chance of possible mistakes in the ODEs if they have to be hard coded in the computer program. This is clearly illustrated by the complexity of the previous equations. An example for such an encoding is given in the by the input file used for **Berkeley Madonna** which is available together with the paper. The second problem is the increasing computational power necessary to solve the ODEs numerically. To address the first problem, **FluxSimulator** is designed to derive the ODEs automatically from three simple text files described in Section 2.3.

2.2. Core algorithm

Initially, all input files are parsed and the data structures necessary to represent the CTN are constructed within the R environment. Afterwards, all ODEs representing the dynamic behavior of the CTN are automatically derived, as illustrated in the subsequent flowchart in Nassi-Shneiderman style (Nassi and Shneiderman 1973).

Core Algorithm — Deriving ODEs From Input Data

```
NOT ALL METABOLIC POOLS CONSIDERED
    GET NEXT METABOLIC POOL
    NOT ALL ISOTOPOMERS CONSIDERED
        GET NEXT ISOTOPOMER
        NOT ALL INFLUXES CONSIDERED
            GET NEXT INFLUX
            MULTIMOLECULAR REACTION
                Y: COMPUTE MULTIMOLECULAR INFLUX CONTRIBUTION
                N: COMPUTE SIMPLE INFLUX CONTRIBUTION
        NOT ALL OUTFLUXES CONSIDERED
            GET NEXT OUTFLUX
            COMPUTE OUTFLUX CONTRIBUTION
        SUM UP INFLUX CONTRIBUTIONS
        SUM UP OUTFLUX CONTRIBUTIONS
        SUBTRACT OUTFLUX FROM INFLUX CONTRIBUTIONS
        SET ODE OF ISOTOPOMER
```

During the derivation of the ODEs all isotopomers of all different metabolites have to be considered. In the first step all incoming fluxes of a particular isotopomer are checked for taking part in a multimolecular reaction on the substrate side, i.e. whether the reaction between two or more metabolites causes the production of the considered metabolite or not.

The contribution caused by an ordinary unimolecular influx to a particular isotopomer fraction is the product of the flux size times the sum of all fractions of isotopomers, within the feeding metabolite pool, feeding the receiving isotopomer fraction. However, the multimolecular contribution differs from that. Because the formation depends on the reaction of at least two different metabolites, the multimolecular contribution is the product of the flux size times the sum of the products of all isotopomer fractions that are able to produce the isotopomer under consideration. Examples for an equation representing a bimolecular reaction are given in Equation 12. All remaining equations include only unimolecular incoming and outgoing fluxes.

The second step of the algorithm considers all exiting fluxes whose particular contributions to the change of the isotopomer pool under consideration are given by the product of the flux size times the fraction of the receiving isotopomer.

For all inflowing and outflowing fluxes all distinct contributions to the change of the currently regarded isotopomer are calculated. All contributions caused by inflowing fluxes are summed and all contributions of outflowing fluxes are subtracted. The resulting value is divided by the size of the metabolic pool the isotopomer belongs to. This is due to the fact that the ODEs (Equations 10 to 16) describe the time derivative of the isotopomer fractions and do not describe the derivative of the absolute amounts of isotopomers.

After having calculated the ODEs for all isotopomer fractions of all metabolites the ODEs are passed to the *lsoda()* function of R. This function solves the ODEs numerically over a given sequence in time. It starts with a set of initial values that were also given in the input files. The *lsoda* algorithm was designed by Petzold (1983); Hindmarsh (1983). It is able to switch between algorithms for stiff and non-stiff systems of first order ODEs for optimal numerical integration.

2.3. Input files

The information necessary to encode a CTN for **FluxSimulator** is distributed over three different input files. This is done for reasons of simplicity and clarity since each file includes another type of information about the network.

The first input file contains the topology of the network and is therefore called the *topology file*. The network is represented in matrix notation. The rows are labeled by the metabolite names and the columns by the flux names. Whenever a flux is an inflowing flux to a particular metabolite pool this is indicated by a '1' at the particular position of the matrix, while in contrast '−1' indicates an outflowing flux and '0' designates a flux that is not connected to a particular metabolite. The structure of the *topology file* is illustrated below on the basis of the example CTN in Figure 2. Note that the matrix in the *topology file* is an adjacency matrix and not equal to a stoichiometric matrix, which is not applicable in CTNs.

```
   v1  v2  v3f v3b v4  v5  v6  v7f v7b v8  v9  v10
A  -1   0   0   0   0   0   0   0   0   0   0   0
B   1  -1  -1   1   0   0   0   1  -1   0   0   0
C   0   1   0   0   0   0   0   0   0  -1   0   0
D   0   0   1  -1  -1   0   0   0   0   0   0   0
E   0   0   0   0   1   1   0   0   0   0  -1   0
F   0   0   0   0   1  -1   0   0   0   0   0   0
G   0   0  -1   1   0   1  -1  -1   1   0   0   0
H   0   0   0   0   0   0   1   0   0   0   0  -1
```

The second input file, the *transition file*, contains information about the individual carbon transitions occurring between the different metabolites and hence between their isotopomer pools. For each metabolite pool that receives carbon atoms from another pool a particular entry is included in the *transition file*. The structure of the entries is illustrated using metabolite D of the example CTN. The complete *transition file* is available with the paper.

```
@transition D 2
   C1  C2  C3  C4
B   1   2  NA  NA
G  NA  NA   1   2
end
```

Each single entry starts with the keyword '`@transition`' followed by the name of the carbon-receiving metabolite pool and the number of metabolites that transfer carbons to the receiving pool. The next lines contain the actual transition information and are specified in matrix notation. The rows are labeled by the name of the carbon-supplying metabolite pools. The columns are labeled by the particular carbon atoms of the carbon-receiving metabolite and range from $C1$ to Cn, where n is the number of carbon atoms the receiving metabolite consists of. The matrix entries are constructed as follows. Each entry corresponds to the position of the carbon in the supplying pool before it was transferred to the receiving pool. If there is no carbon atom transferred, the entry of the matrix is '*NA*', which indicates *not available* or *missing value* in R.

For instance, the transition entry for the carbon-receiving metabolite pool D shows a '1' and a '2' in the row of carbon transferring metabolite B, followed by two times '*NA*'. This means that the first and the second carbon of B are transferred to the first and second carbon position of metabolite C, respectively, and no carbons are transferred from B to the third and fourth position. Each single transition entry ends with the keyword '`end`'. Specifying the transition entries the entire transition file is built.

The third input file is called *parameter file* and contains the flux and pool sizes as well as the number of carbon atoms for each particular metabolite. The *parameter file* of the example CTN is given below. The entry containing the parameters for the metabolite pools starts with the keyword '`@poolparameter`' followed by a matrix wise representation containing the pool sizes and the number of atoms. The first row with keyword '`size`' contains the particular pool sizes. The second row, labeled by '`atoms`', contains the number of carbon atoms for each metabolite pool. The columns are labeled by the pool names and the whole entry ends with the keyword '`end`'.

```
@poolparameter [micromol/g]
        A    B    C    D    E    F    G     H
size   100  100  100  100  100  100  100   100
atoms  2    2    2    4    1    3    2     2
end

@fluxparameter [micromol/(g * s)]
        v1   v2  v3f  v3b    v4   v5   v6  v7f  v7b  v8  v9  v10
size   24.5  9   17   15.5  1.5  1.5  14    13   27   9   3   14
end
```

The subsequent entry contains the flux sizes and is encoded in the same manner as for the pool sizes, although no number of atoms is needed and the labeling of the columns is done by the flux names. Additionally, the entry starts with the keyword '`@fluxparameter`'. A complete *parameter file* should contain all pool and flux descriptions.

2.4. Implementation and application

According to the object-oriented programming style the entire package was designed based on S4 classes, which are the backbone of object-oriented programming in R (Chambers 2003). They provide all usual object-oriented features such as instantiation, polymorphism and inheritance. Object oriented programming is very intuitive and structured. Additionally, it offers advantages when dealing with biological systems, whose modular, hierarchical structure is often naturally representable in terms of objects.

The **FluxSimulator** implementation was validated using the CTN given in Figure 2. This CTN was specified in the three previously described input files. For comparison the ODE representation was also encoded in a file applicable in **Berkeley Madonna** 8.3.8. It mainly consists of the 42 ODEs represented by Equations 10 to 16.

In order to specify the input files correctly, one additional constraint has to be considered. The CTN is assumed to be in metabolic steady state, which means that the pool size of each metabolite is not allowed to change over time, although the fractions of its isotopomers are changing. Hence, the flux sizes have to be solutions of the linear equation system in Equation 17 to 23, which results from Equations 1 to 7 with the time derivative set to zero:

$$\begin{aligned} v_1 + v_{3b} + v_{7f} &= v_2 + v_{3f} + v_{7b} && (17)\\ v_2 &= v_8 && (18)\\ v_{3f} &= v_4 + v_{3b} && (19)\\ v_4 + v_5 &= v_9 && (20)\\ v_4 &= v_5 && (21)\\ v_5 + v_{3b} + v_{7b} &= v_{3f} + v_6 + v_{7f} && (22)\\ v_6 &= v_{10} && (23) \end{aligned}$$

This under-determined system of equations guarantees the metabolic steady state. The amount of all inflowing metabolites is equal to the amount of outflowing metabolites for each particular metabolite pool. The system consists of seven equations and twelve unknown parameters, the flux sizes. Thus, five flux sizes may be chosen freely and are arbitrarily set to $v_1 = 24.5 \left[\frac{\mu mol}{g \cdot s}\right]$, $v_{3f} = 17 \left[\frac{\mu mol}{g \cdot s}\right]$, $v_{7f} = 13 \left[\frac{\mu mol}{g \cdot s}\right]$, $v_8 = 9 \left[\frac{\mu mol}{g \cdot s}\right]$ and $v_9 = 3 \left[\frac{\mu mol}{g \cdot s}\right]$ in the example. The remaining seven flux sizes were then computed by solving the system of equations and resulted in $v_2 = 9 \left[\frac{\mu mol}{g \cdot s}\right]$, $v_{3b} = 15.5 \left[\frac{\mu mol}{g \cdot s}\right]$, $v_4 = 1.5 \left[\frac{\mu mol}{g \cdot s}\right]$, $v_5 = 1.5 \left[\frac{\mu mol}{g \cdot s}\right]$, $v_6 = 14 \left[\frac{\mu mol}{g \cdot s}\right]$, $v_{7b} = 27 \left[\frac{\mu mol}{g \cdot s}\right]$ and $v_{10} = 14 \left[\frac{\mu mol}{g \cdot s}\right]$.

After the specification of the input files the dynamic behavior of the CTN was simulated in both packages for a time period of 1000 seconds. The usage of **FluxSimulator** is described below while the usage of **Berkeley Madonna** can be found elsewhere (Macey *et al.* 2000).

The initial step getting started with **FluxSimulator** is to install and load the **FluxSimulator** package. Additional packages necessary to install are *wle*, *circular*, *boot* and *odesolve* which are freely available on the Comprehensive R Archive Network (CRAN). Once the **FluxSimulator** package is loaded the function call *fluxsim()* starts an interactive dialog that guides the user through all necessary specifications. This dialog is subsequently explained on the base of the input necessary to perform the simulation of the example CTN.

```
Please enter the path to the topology file!
1: Data/topology.txt
Please enter the path to the transition file!
1: Data/transition.txt
Please enter the path to the parameter file!
1: Data/parameter.txt
Please enter the start time!
1: 0
Please enter the end time!
1: 1000
Please enter the size of the time steps!
1: 0.1
Please enter the name of the initially labeled pool!
1: A
Please enter the 4 different isotopomer fractions!
1. isotopomer fraction:
1: 0.01
2. isotopomer fraction:
1: 0.01
3. isotopomer fraction:
1: 0.97
4. isotopomer fraction:
1: 0.01
```

The first three input parameters (lines two, four and six) represent the relative or absolute locations of the input files on the system. Note that the name of the input files has to include the file extension. The subsequent input parameters specify the time period that has to be simulated (start and end time) and the size of the time steps (lines seven to twelve). The last two input parameters (lines 13 to 19) specify the name of the initially labeled pool and its isotopomer fractions. Note that the number of different isotopomer fractions depends on the number of carbon atoms present in the initially labeled metabolite and can be lower or higher than in this example. Additionally, all given isotopomer fractions have to sum up to '1'. After entering the last isotopomer fraction a confirmation by the enter button starts the simulation. Finally, the dynamic behavior of each metabolite's isotopomer fractions is plotted.

An additional possibility to start the simulation is suited for users who do not want to be guided through the specifications. Then the function call should include all arguments necessary for the computation as for the example simulation below:

```
fluxsim(topologyFile = "topology.txt",
        transitionFile = "transition.txt",
        parameterFile = "parameter.txt",
        simTime = c(0, 1000, 0.1),
        labeledPool = "A",
        isotopomerFractions = c(0.01, 0.01, 0.97, 0.01))
```

3. Results and discussion

In the following the simulation results of **FluxSimulator** and **Berkeley Madonna** achieved during the simulation of the dynamics of the CTN are compared. Additionally, the handling and performance of the packages is discussed.

As mentioned previously the numerical integration method chosen for **FluxSimulator** is the *lsoda* algorithm of Petzold et al. (Petzold 1983). The algorithm chosen in **Berkeley Madonna** was developed by Runge and Kutta (Albrecht 1977). All results achieved during the numerical integration with **Berkeley Madonna** were given to the seventh decimal place behind the decimal point. Those of **FluxSimulator** up to the tenth decimal place. It turned out that both simulations computed exactly the same results in each iteration step when the values of **FluxSimulator** were rounded to the seventh decimal position. Hence, both simulations computed equal dynamic behavior for all isotopomer fractions. This confirmed the adequate performance of **FluxSimulator**, for which three example plots are given in Figure 4.

Berkeley Madonna receives one input file in plain text format. Within this input file each ODE describing the behavior of a single isotopomer has to be encoded by hand, leading to a total of 42 ODEs for the current example. This leads to the following issues:

1. All network properties have to be considered manually, e.g. multimolecular reactions or the fact that one isotopomer could be fed by more than one isotopomer of another metabolite.

2. Small changes in the network can cause various changes in several ODEs that have to be recoded carefully by hand.

3. Unexperienced users prefer the specification of a more intuitive network representation, above hard-coded equations.

It is clear that manual specification of the ODEs is a potential source of error and leads to limited flexibility and low biologist-friendliness. This is addressed by **FluxSimulator** and its automatic derivation of the ODEs from the three intuitive input files. These files can be edited easily. Hence, changes in the network can be made very fast and consistently, leading to a high flexibility of the package. Moreover, the generation of the input files is feasible for a biologist without mathematical and computer science training.

4. Conclusions and outlook

In this work we presented a new R package called **FluxSimulator** to simulate isotopomer distributions in metabolic networks over time. The package was developed using object-oriented S4 classes. It was specifically designed to derive the mathematical representation underlying the dynamics of the network automatically using intuitive input files. **FluxSimulator** and **Berkeley Madonna** computed an identical behavior for the example CTN. However, in contrast to **Berkeley Madonna** or other available simulation programs, **FluxSimulator** is suitable for users with a weak mathematical background. This is due to the fact that the enormous potential for errors during direct encoding of all ODEs representing a CTN is circumvented. Additionally, the user can more easily and consistently experiment with the model system,

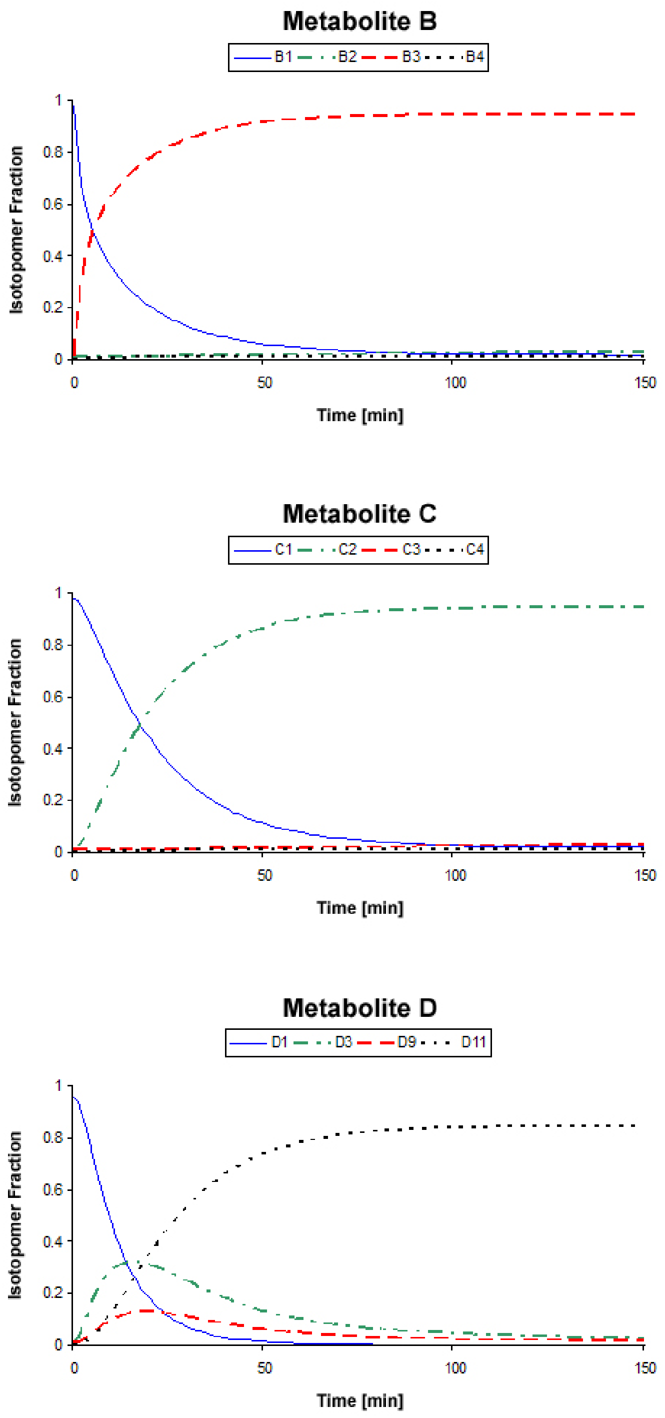


Figure 4: Dynamic change of the isotopomer fractions of metabolite B, C and D (four of the sixteen isotopomers) over time. Before $t = 0$ all metabolites isotopomer fractions were constant at the natural abundance level of 1.1% of carbon isotope. After $t = 0$ the isotopomer fractions of A were fixed at $a_1 = 0.01$, $a_2 = 0.01$, $a_3 = 0.97$ and $a_4 = 0.01$.

e. g. changing the network topology. A web interface for users that want to simulate their CTNs without installing R is planned and will enhance the user-friendliness.

Hence, **FluxSimulator** is an appropriate alternative to simulate isotopomer distributions over time in metabolic networks. Future work will improve and extend **FluxSimulator** further with special focus on the following properties:

1. Incorporation of algorithms to eliminate ODEs not necessary for the computation of the desired simulation, hence increasing computational speed.
2. Analysis of experimental NMRS data to estimate the fluxes.
3. Application of the statistical facilities of R to simulate hypothetical NMRS data, e.g. by adding measurement noise.
4. Application of the optimization routines of R to estimate unknown flux parameters from the experimental NMRS data.
5. Incorporation of algorithms to improve the speed of solving the ODEs.
6. Increase of flexibility with respect to the input of labeled isotopomers, e.g. usage of several metabolite pools as entry point for labeled substrate, or flexible functions of time.
7. Extension to other isotopes like ^{2}H, ^{15}N, ^{17}O, ^{18}O and ^{33}S, or a combination of them.
8. Development of a concise and easy to handle graphical user interface to enhance the flexibility and the comfort.
9. Analysis of MS data in combination with NMRS data to estimate the flux parameters.

All these improvements can make **FluxSimulator** a powerful, flexible and easy to handle tool to analyze and simulate isotope labeling NMRS and MS experiments.

Acknowledgments

This work is part of the BioRange program (project number SP 2.2.1) of the Netherlands Bioinformatics Centre (NBIC), which is supported by a BSIK grant through the Netherlands Genomics Initiative (NGI). The work is also part of the Center for Medical Systems Biology which is a Genomics Center of Excellence funded by the Dutch Government via the NGI. K.M. was supported by Computational Science grant #635.000.014 from the Netherlands Organization for Scientific Research (NWO).

References

Albrecht P (1977). "The Runge-Kutta Theory in a Nutshell." *SIAM Journal on Numerical Analysis*, **14**, 1006–1021.

Böhm H, Klebe G, Kubinyi H (2002). *Wirkstoffdesign - Der Weg zum Arzneimittel.* Spektrum Akademischer Verlag, Heidelberg.

Chambers J (2003). *Programming with Data: A Guide to the S Language.* Springer, Berlin.

Chance E, Seeholzer S, Kobayashi K, Williamson J (1983). "Mathematical Analysis of Isotope Labeling in the Citric Acid Cycle with Applications to 13C NMR Studies in Perfused Rat Hearts." *Journal of Biological Chemistry*, **258**, 13785–13794.

Dunn I, Heinzle E, Ingham J, Jiri E (2003). *Biological Reaction Engineering. Dynamic Modelling Fundamentals with Simulation Examples.* Wiley-VCH Verlag GmbH & Co. KGaA, Weinheim.

Hindmarsh A (1983). "**ODEPACK**, A Systematized Collection of ODE Solvers." *Scientific Computing*, pp. 55–64.

Leach A, Gillet V (2003). *An Introduction to Chemoinformatics.* Springer, Berlin.

Macey R, Oster G, Zahnley T (2000). **Berkeley Madonna** *User's Guide.* University of California.

Michaelis L, Menten ML (1913). "Die Kinetik der Invertinwirkung." *Biochemische Zeitschrift*, **49**, 333–369.

Nassi I, Shneiderman B (1973). "Flowchart Techniques for Structured Programming." *SIGPLAN Notices*, **8**, 12–26.

Petzold L (1983). "Automatic Selection of Methods for Solving Stiff and Nonstiff Systems of Ordinary Differential Equations." *SIAM Journal on Scientific Computing*, **4**, 136–148.

Raymond G, Butterworth E, Bassingthwaighte J (2003). "**JSIM**: Free Software Package for Teaching Phyiological Modeling and Research." *Experimental Biology*, **280.5**, 102.

R Development Core Team (2006). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL `http://www.R-project.org/`.

Sauer F, Erfle J, Binns M (1970). "Turnover Rates and Intracellular Pool Size Distribution of Citrate Cycle Intermediates in Normal, Diabetic and Fat-Fet Rats Estimated by Computer Analysis from Specific Activity Decay Data of 14C-Labeled Citrate Cycle Acids." *European Journal of Biochemistry*, **17**, 350–363.

van Beek JM, Csont T, de Kanter F, Bussemaker J (1998). "Simple Model Analysis of 13C NMR Spectra to Measure Oxygen Consumption Using Frozen Tissue Samples." *Advances in Experimental Medicine and Biology*, **454**, 475–485.

van Beek JM, van Mil H, King R, de Kanter F, Alders D, Bussemaker J (1999). "A 13C NMR Double-Labeling Method to Quantitate Local Myocardial O2 Consumption Using Frozen Tissue Samples." *American Journal of Physiology*, **277**, H1630–H1640.

Wiechert W (2001). "13C Metabolic Flux Analysis." *Metabolic Engineering*, **3**, 195–206.

Wiechert W, de Graaf A (1997). "Bidirectional Reaction Steps in Metabolic Networks: I. Modeling and Simulation of Carbon Isotope Labeling Experiments." *Biotechnology and Bioengineering*, **55**, 101–117.

Wiechert W, de Graaf A, Marx A (1995). "In Vivo Stationary Flux Determination Using 13C NMR Isotope Labelling Experiments." *Advances in Biochemical Engineering/Biotechnology*, **54**, 109–154.

Wiechert W, Möllney M, Isermann N, Wurzel M, de Graaf A (1999). "Bidirectional Reaction Steps in Metabolic Networks: III. Explicit Solution and Analysis of Isotopomer Labeling Systems." *Biotechnology and Bioengineering*, **66**, 71–85.

Wiechert W, Möllney M, Petersen S, de Graaf A (2001). "A Universal Framework for 13C Metabolic Flux Analysis." *Metabolic Engineering*, **3**, 265–283.

Wiechert W, Siefke C, de Graaf A, Marx A (1997). "Bidirectional Reaction Steps in Metabolic Networks: II. Flux Estimation and Statistical Analysis." *Biotechnology and Bioengineering*, **55**, 118–135.

Affiliation:

Thomas W. Binsl
Faculteit der Exacte Wetenschappen
Vrije Universiteit Amsterdam
De Boelelaan 1083a
1081HV Amsterdam, The Netherlands
Telephone: +31/02/5987734 E-mail: tbinsl@few.vu.nl
URL: http://www.few.vu.nl/~tbinsl/

Journal of Statistical Software http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 18, Issue 7 *Submitted:* 2006-09-28
January 2007 *Accepted:* 2007-01-10