Bioinformatics, 36(14), 2020, 4163–4170
doi: 10.1093/bioinformatics/btaa280
Advance Access Publication Date: 10 July 2020
Original Paper

OXFORD

Systems biology

Boosting the extraction of elementary flux modes in genome-scale metabolic networks using the linear programming approach

Francisco Guil*, José F. Hidalgo and José M. García

Departamento de Ingeniería y Tecnología de Computadores, Universidad de Murcia, Murcia 30080, Spain

*To whom correspondence should be addressed.

Associate Editor: Pier Luigi Martelli

Received on December 19, 2019; revised on April 16, 2020; editorial decision on April 17, 2020; accepted on April 22, 2020

Abstract

Motivation: Elementary flux modes (EFMs) are a key tool for analyzing genome-scale metabolic networks, and several methods have been proposed to compute them. Among them, those based on solving linear programming (LP) problems are known to be very efficient if the main interest lies in computing large enough sets of EFMs.
Results: Here, we propose a new method called EFM-Ta that boosts the efficiency rate by analyzing the information provided by the LP solver. We base our method on a further study of the final tableau of the simplex method. By performing additional elementary steps and avoiding trivial solutions consisting of two cycles, we obtain many more EFMs for each LP problem posed, improving the efficiency rate of previously proposed methods by more than one order of magnitude.
Availability and implementation: Software is freely available at https://github.com/biogacop/Boost_LP_EFM.
Contact: fguil@um.es
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Elementary flux modes (EFMs) are one widely known tool in computational systems biology for studying genome-scale metabolic networks (GSMNs) reconstruction. EFMs represent a finite set of possible states that can generate all the possible states of the network by using convex combinations (see, e.g. De Figueiredo *et al.*, 2009; Gagneur and Klamt, 2004; Klamt *et al.*, 2005; Rezola *et al.*, 2011; Röhl *et al.*, 2019; Schuster *et al.*, 1999, 2000; Schuster and Hilgetag, 1994). Unfortunately, the cardinality of this set is typically very large so it can only be completely computed in a few cases (Hunt *et al.*, 2014). Methods to compute sets of EFMs can be roughly divided into two families (Planes and Beasley, 2008), according to if they rely on properties of the associated graph (path-finding methods) (Arabzadeh *et al.*, 2018; Hidalgo *et al.*, 2015) or just on the study of the stoichiometric matrix of the network.

Regarding the second approach, different methods have been proposed to solve the system of linear equations proposed by the stoichiometric matrix. The most used ones are the double description method (DDM) (Fukuda and Prodon, 1995), the mixed integer linear programming (MILP) method (De Figueiredo *et al.*, 2009; Rezola *et al.*, 2011; Röhl and Bockmayr, 2017) and the linear programming (LP) method (Machado *et al.*, 2012; Tabe-Bordbar and Marashi, 2013). The main advantage of the methods based on DDM and MILP is that they can, theoretically, produce the full set of EFMs. On the other hand, LP methods are faster and can more efficiently produce big sets of EFMs, but they cannot assure when all of the sets of EFMs have been computed.

Methods based on LP techniques are capable of producing sets of EFMs at a better efficiency rate, both in time and computer resources. Several efforts have been made to propose efficient algorithms that can produce large enough sets of EFMs in GSMNs (Kaleta *et al.*, 2009; Quek and Nielsen, 2014; Pey *et al.*, 2015). The critical point of these techniques is to use different additional constraints and objective functions to transform the stoichiometric and thermodynamic feasibility constraints into an optimization problem. In this way, for any such LP problem, a solution can be an EFM under certain hypothesis (Pey and Planes, 2014). Different strategies try to minimize the associated issues that can appear (usually, infeasible problems or repeated solutions). The efficiency rate of the LP method is defined by how many LPs are needed to run to find a new EFM. It is known that the 'ideal' rate is to find a new EFM for each LP problem run. As far as we know, the best efficiency rate was obtained by Pey *et al.* (2015), where they achieved an efficiency rate of 1.3 (i.e. a new EFM was obtained for each 1.3 LP problems solved). Perhaps due to the fact that it is quite close to the ideal rate, the interest in this topic has slowly decreased.

This article presents a new LP approach in which this ideal rate is overcome. We have developed our proposal based on two ideas

extracted from the LP problem-solving procedure. First, the objective function (and so the optimal solution obtained) is just a tool to produce a vertex of the (restricted) cone of solutions given by our constraints. As previously noted (Pey and Planes, 2014) just using one additional positive constraint (in a sense explained below) gives that all the vertices of the restricted cone correspond to the subset of EFMs of the network that fulfills the added constraint. Second, the simplex algorithm (Taha, 2016) is based on a series of steps, and each of them can be viewed, after obtaining a first feasible solution, as a way to pass from one vertex to another.

Putting these two ideas together, our approach (named EFM-Ta, which stands for EFMs using the tableau) uses the LP simplex algorithm to produce an initial optimal solution that is an EFM and, after that, some simple steps are performed to obtain new vertices, that is, new EFMs. To limit the possible rounding problems arising from the use of floating point numbers, EFM-Ta performs just one possible step from the initial vertex obtained. However, having plenty of possible ways of doing this step leads to the possibility of obtaining lots of new vertices from the initial one. In the article, we call these vertices the *adjacent* vertices of the first computed solution.

As observed in Gerstl *et al.* (2015), it is desirable to avoid the appearance of two cycles. In the current context, this is especially important for two reasons. First, these solutions are sometimes false EFMs, in the sense of consisting of pairs of irreversible reactions coming from the splitting of a reversible one (Klamt *et al.*, 2005). In general, the output returned by the simplex method cannot be of this unwanted type, but the use of an additional constraint can produce its appearance. In the experiments performed, we have seen that this is a common behavior that dramatically decreases the efficiency of the methods used. Moreover, these two-length vertices (in the sense of corresponding to EFMs with support of cardinality two) tend to have adjacent vertices that also are two length. So, EFM-Ta has been extended to avoid this kind of undesired solution.

EFM-Ta obtains efficiency rates that go far beyond those achieved by previous methods by a factor of 10× and so breaking the ideal rate of one EFM by LP problem solved. Our approach has been tested in previously studied GSMNs models and compared with well-known tools (as EFMEvolver or treeEFM), obtaining an increase in the efficiency rate of more than one order of magnitude.

The main contributions presented in this article are the following:

- A further study of the final tableau of the simplex method. For each LP problem posed, new vertices (EFMs) from the original one are calculated, invoking additional elementary simplex steps.
- The importance of avoiding two-cycle EFMs as solutions. To do that, reactions that come from reversible ones are avoided, and also those that appear in previous EFMs of length 2.
- The significant improvement by a factor of 10× of the efficiency rate in the extraction of sets of EFMs.

We have not used any heuristic rule for this increment (setting apart the avoiding of two cycles), so we expect that this efficiency rate can be greatly improved in subsequent works. We also want to point out that this method of obtaining new vertices from the initial one can also be easily added to other approaches based on LP methods.

The associated matrix to any metabolic network is called its stoichiometric matrix, and it represents the processes that can take place on the network. Rows and columns of S represent the internal metabolites and reactions, respectively. So, if the network has m metabolites and n reactions, the associated matrix S has m rows and n columns. Each value of the matrix is the stoichiometric coefficient of the corresponding column (reaction) in the metabolic equation represented by its row.

Any given state of the network is characterized by a vector of variables of length equal to the number of reactions. The corresponding values of the variables give the rate at which each reaction is performing in this state. This vector is called a flux rate.

If the time interval considered is small enough, it is normal to assume that the concentrations (internal metabolites) are stable. This leads to the so-called steady-state constraint:

$$S \cdot v = 0. \tag{1}$$

Let R and Irr be the sets of reactions and irreversible reactions of our network. Irreversible reactions are those that can only occur in one direction, while reversible ones are those that are not irreversible. The usual method to deal with a reversible reactions is to replace it with two irreversible ones, representing its two possible directions.

For Irr reactions, the flux rates must be non-negative, what is known as the thermodynamic constraint:

$$v[j] \geq 0, \quad \forall j \in Irr. \tag{2}$$

A flux vector is called a *mode* if it fulfills both the steady-state and thermodynamic constraints.

If v is a mode, its support $supp(v)$ is defined as the set of those reactions r that appear in v with a non-zero rate. A mode v is called an elementary mode, or EFM, if its support is minimal (i.e. there is no other non-zero mode v' with $supp(v') \subsetneq supp(v)$) (see Schuster and Hilgetag, 1994).

For any non-trivial network, its set of modes is infinite. However, the set of EFMs of the network is finite and any non-elementary mode can be written (in a non-unique way) as a linear combination of EFMs using non-negative coefficients (see, e.g. Klamt and Stelling, 2003). In this way, the problem of getting all the modes is translated to the problem of computing the set of all EFMs.

To tackle the problem of whether a given mode is elementary or not, the well-known characterization of EFMs in terms of the stoichiometric matrix is used. For any mode v, the submatrix S_v of S is constructed by taking just those columns corresponding to reactions in $supp(v)$. If all the reactions of the network are irreversible, linear algebra methods can prove that a mode v is an EFM if and only if $rank(S_v) = k - 1$ for $k = |supp(v)|$ (Klamt *et al.*, 2005; Terzer and Stelling, 2008).

Linear programming techniques

As stated in Section 1, one approach to find EFMs is the use of linear optimization techniques. In this way, it is common to use any libraries that are publicly available to handle this kind of problems. To do so, the procedure starts from the stoichiometric and thermodynamic constraints that define a (polyhedral) cone of solutions and introduce an *ad hoc* objective function to convert them into a linear program. To guarantee that this problem is bounded, the objective function is minimized along with the use of non-negative coefficients.

The direct translation of a stoichiometric matrix into a linear program defines a *clean linear program* [Equation (3)].

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{n} a_i \cdot v[i] \\ \text{subject to} \quad & S \cdot v = 0 \\ & v[i] \geq 0 \quad \forall r_i \in Irr \end{aligned} \tag{3}$$

But, in this case, one minimal solution is obtained by setting all variables to zero. So, the problem must be additionally restricted to get non-trivial solutions. Different conditions must be used to modify this clean linear program in order to constrain it. It can be imposed that certain set of reactions must appear with non-zero flux (at least one of them). This kind of constraints are called positive constraints (see Acuña *et al.*, 2009 or Hidalgo *et al.*, 2017). Other times, the imposition is that a set of reactions must be absent in the final solution (negative constraints). If the set of reactions of interest is $J = \{r_{i_1}, \ldots, r_{i_k}\}$, the constraints can be written as

$$\sum_J v[i_j] = 1 \text{ (for positive constraints)}, \tag{4}$$

$$\sum_J v[i_j] = 0 \text{ (for negative constraints)}. \tag{5}$$

So, different modes (solutions) can be obtained by choosing different sets of positive and negative constraints.

However, in order to compute EFMs, only certain types of constraints can be used. To avoid possible infeasible LP problems, just one positive constraint is used. The LP problem posed with just one positive constraint and any number of negative ones gives a solution that is an EFM, whenever all reactions are irreversible and the simplex algorithm is used to solve the LP problem (Pey and Planes, 2014).

2 Materials and methods

2.1 Posing the LP problem: the influence of reversible reactions on the positive constraint

In order to avoid possible infeasible LP problems, in addition to just use one positive constraint, the blocked reactions are removed from the network (Hidalgo *et al.*, 2017).

As stated before, the conventional way to obtain EFMs using the simplex method is by taking into account the stoichiometric and thermodynamic constraints, and additionally constraining the problem with one positive constraint. Then, the following LP problem has to be posed:

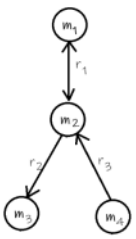
$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{n} a_i \cdot v[i] \\ \text{subject to} \quad & S \cdot v = 0 \\ & \sum_J v[i_j] = 1 \\ & v[i] \geq 0 \quad \forall r_i \in Irr \end{aligned}, \tag{6}$$

where the objective function includes a random number of reactions with non-negative coefficients, and only one positive constraint is chosen from a set of reactions $J = \{r_{i_1} \cdots, r_{i_k}\}$ using Equation (4). Then, every solution returned by the solver is always an EFM.

However, we have found that the combination of one positive constraint together with reversible reactions can produce undesired artifacts in the LP problem, leading to the solution found is not always an EFM. Remind that reversible reactions are replaced with two irreversible ones, representing its two possible directions.

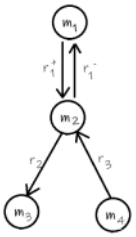
This anomalous behavior can be observed in the following example.

Consider the metabolic network given by the following graph:



in which the reaction r_1 is reversible and all stoichiometric values are set to 1. This network has three EFMs with supports $\{r_1, r_2\}$ (in this case r_1 acts in the direction from m_1 to m_2), $\{r_1, r_3\}$ (r_1 in the opposite direction) and $\{r_2, r_3\}$.

If the reaction r_1 is replaced with two irreversible reactions r_1^+ and r_1^-, the following graph is derived:



For the following order of variables (r_1^+, r_1^-, r_2, r_3) (only considering internal metabolites), this stoichiometric matrix is derived

$$S = \begin{pmatrix} -1 & 1 & 0 & 0 \\ 1 & -1 & -1 & 1 \end{pmatrix}.$$

Then, the restricted LP problem obtained by adding the constraint $r_1 = 1$ yields as solution the tuple $(1, 1, 0, 0)$ that does not correspond to any EFM.

The problem in the previous example is due to a solution is not allowed to include together the two reactions obtained from the same reversible one. But, as shown, this cannot be assured by just posing LP problems and one positive reaction.

Our next result shows how to avoid this undesired behavior:

Theorem 1. *Let M be a metabolic network. Without loss of generality, the reactions can be reordered so that $\{r_1, \ldots, r_k\}$ are irreversible and $r_{k+1}, \ldots, r_n$ are reversible. For each reversible reaction r_i, a pair of irreversible reactions $\{r_i^+, r_i^-\}$ representing the two possible signs for the flux of the reaction r_i are introduced. Let us take a list of non-negative numbers $\{a_1, \ldots, a_k\}$ such that at least one of the a_i is non-zero. Consider the subset $V \subset \mathbb{R}^n$ defined by the following constraints*

$$\begin{aligned} & S \cdot v = 0 \\ & v \geq 0 \\ & \sum_{i=1}^{k} a_i \cdot v[i] = 1 \end{aligned}. \tag{7}$$

If $V \neq \varnothing$ then any extreme point of V corresponds to an EFM of the network.

Observe that, in this result, the positive constraint only includes reactions that does not come from reversible ones. This theorem improves the known result from Pey and Planes (2014) in order to avoid situations as in our previous example. A proof of this result can be found in Supplementary Material of this article.

Remark: It is easy to prove that the only situation in which a vertex can include both reactions, coming from a reversible one, is in the trivial case that only includes those two reactions with the same flux value. Even so, in practical situations, it is better to avoid them (see Supplementary Material). To conclude this section, LP problems with a positive constraint must only include reactions that do not come from reversible ones to assure the obtention of EFMs.

2.2 Getting more information from the solution

The main drawback of the LP methods is that, for every posed LP problem, a unique EFM is obtained, even though the proposed minimization problem can have a non-unique solution, which is a very common case. Moreover, some of the obtained EFMs are repeated, so the efficiency rate is not quite good.

EFM-Ta focuses on obtaining, for any LP problem posed, as many solutions as possible, that is, as many vertices as possible from the cone of solutions. To do so, we leverage the *tableau* information that the simplex method handles along the process of finding the LP

problem solution. Let us start by re-examining the steps performed by the simplex method.

As commented in Taha (2016), the simplex algorithm is developed in two phases. The first phase is devoted to obtain a first feasible solution that is a vertex of the polyhedron, introducing some artificial variables. Once a vertex is found, all artificial variables should have zero value and be removed so the process continues in the second phase using just the original ones (and slack variables if needed). The second phase uses the found vertex and the gradient of the function to try to obtain another vertex with a lower function value (if it is minimizing the function). Let us briefly examine how this is done.

Associated to the function $f(x_1,\dots,x_n)=\sum a_i x_i$ the stoichiometric matrix $S=\begin{pmatrix} s_{11} & \cdots & s_{1n} \\ s_{21} & \cdots & s_{2n} \\ \cdots & \cdots & \cdots \\ s_{m1} & \cdots & s_{mn} \end{pmatrix}$ and the positive constraint $\sum d_j x_j = 1$ the following tableau is started:

$-a_1$	$-a_2$	$\cdots$	$-a_n$	0
s_{11}	s_{12}	$\cdots$	s_{1n}	0
s_{21}	s_{22}	$\cdots$	s_{2n}	0
$\cdots$	$\cdots$	$\cdots$	$\cdots$	0
s_{m1}	s_{m2}	$\cdots$	s_{mn}	0
d_1	d_2	$\cdots$	d_n	1

Suppose that, in one step of the second phase, a vertex $P=(x_1,\dots x_n)$ is set. At this point, the simplex method splits the set of reactions into two disjoint subsets. The reactions of these two subsets are called basic and non-basic variables (respectively). The columns of the tableau (putting aside the first row) in the original one corresponding to the basic variables form an invertible matrix B. In this step, another tableau can be obtained from the original one by suitably multiplying by the matrix B^{-1} and, in this new tableau, the columns corresponding to the basic variables form the identity matrix (except for the order).

z_1-c_1	z_2-c_2	$\cdots$	z_n-c_n	c
s'_{11}	s'_{12}	$\cdots$	s'_{1n}	b'_1
s'_{21}	s'_{22}	$\cdots$	s'_{2n}	b'_2
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
s'_{m1}	s'_{m2}	$\cdots$	s'_{mn}	b'_m
d'_1	d'_2	$\cdots$	d'_n	b'_{m+1}

The numbers $b'_1,\dots,b'_{m+1}$ are non-negative numbers. The vertex P from this tableau is calculated by assigning a zero value to the non-basic variables and solving the trivial associated system.

The process of trying to obtain a new vertex is started by selecting a non-basic variable x_j. The first element of the associated column (i.e. z_j-c_j) tells how the proposed change is going to affect the value of the function. If this number is 0, then the function will have the same value in the new vertex that it had in P. If the number is negative (or positive), the function value will decrease (or increase) in the new vertex.

To obtain the new vertex, the values assigned to the variables must be carefully chosen so that they are all non-negative and fulfill all the constraints of the problem. To do so, a column j is chosen such that there are positive values s'_{ij} with the corresponding b'_i also positive. To proceed, all the values of the jth column with this property are taken and the minimum of the corresponding quotients $\frac{b'_i}{s'_{ij}}$ is calculated. Then, the row i, such that $\frac{b'_i}{s'_{ij}}>0$ is minimal is selected and a pivoting step is done that converts the j column into the column vector $e_i=(0,\dots,0,1,\dots,0,0)^T$ (having the 1 in the ith position).

This process affects the values of the variables in the following way:

- The variable x_j is now considered basic and its value is set to $\frac{b'_i}{s'_{ij}}>0$.
- Let x_{i_0} be the basic variable such that its column in the previous step is e_i. This variable is considered non-basic (its value is 0).
- The remaining variables remain basic or non-basic. For the basic ones, their values under the pivoting step must be updated.

Suppose a basic variable x_{i_0} such that the only non-zero entry in its column appears in the row i_1. Then, the following tableau is formed, in which only the columns corresponding to the reaction x_{i_0} and the new basic variable x_j and the rows i_1 and i are shown.

$\cdots$	$z_{i_0}-c_{i_0}$	$\cdots$	z_j-c_j	$\cdots$	c
$\cdots$	0	$\cdots$	s'_{1j}	$\cdots$	b'_1
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\cdots$	1	$\cdots$	$s'_{i_1 j}$	$\cdots$	b'_{i_1}
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$
$\cdots$	0	$\cdots$	s'_{ij}	$\cdots$	b'_i
$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$	$\cdots$

In order to transform the i_1 entry of the column j into zero, the ith row multiplied by $\frac{s'_{i_1 j}}{s'_{ij}}$ must be added to the i_1th row. Therefore, the value of this variable changes from b'_{i_1} to $b'_{i_1}-b'_i\cdot\frac{s'_{i_1 j}}{s'_{ij}}$.

Most solvers use the revised simplex method in which, in order to speed up the calculations, the tableau is not calculated in intermediate steps. Instead, only the original tableau is kept in memory and only the matrix B^{-1} is updated in each step, so the tableau is only available in the final step of the process.

Therefore, EFM-Ta only calculates the vertices that can be easily computed from the final tableau. Our approach consists of the following steps:

1. Get the final solution (vertex) and final tableau from the solver
2. Obtain the list of non-basic variables x_j
3. For each one, check if there are positive values s'_{ij} with the corresponding b'_i value also positive
4. If this is the case, obtain a new vertex (EFM) by manually doing the pivoting step as previously explained

2.3 Tuning the computing of the adjacent vertices in the EFM extraction

Our proposed approach has still some drawbacks and requires a fine-tuning optimization.

First, in some commercial optimization packages, for example in the IBM package CPLEX used in this article (http://www-01.ibm.com/software/integration/optimization/cplex-optim), not all artificial variables are removed at the end of the first phase of the simplex method if the problem is degenerated (as in this case). Instead, the remaining artificial variables are constrained to have zero value during the rest of the optimization process. When the final tableau is retrieved, a list of basic variables is obtained, but not all the variables that are not in that list are non-basic ones.

Second, pivoting steps without having the full list of non-basic variables can produce false extreme points (modes that are not EFMs), preventing some other adjacent vertices from being reached by another pivoting step.

Finally, the pivoting steps require the use of the S matrix in dense form, which produces a slowdown of the computation process.

Therefore, we have elaborated more on our approach to overcoming these drawbacks. We have tuned our proposal for steps 2–4, replacing the pivoting step by invoking another *restricted* LP

problem. Then, EFM-Ta searches for new vertices invoking another LP problem with the same constraints but starting with the previously obtained solution, changing the objective function and limiting to one the number of steps performed by the solver. EFM-Ta uses as new objective functions those defined from subsets of reactions contained in the support of the solution, and do so while the number of repeated adjacent vertices computed is below a certain threshold. Algorithm 1 summarizes our EFM-Ta proposal.In the Algorithm, the *randomFunction* builds a function based on randomly chosen reactions from the given set; the *randomReaction* selects a random reaction from the given set; the *poseLinearProgram* builds the LP problem with objective function f, stoichiometric matrix S, and positive additional constraint *cons*; and finally, the *poseRestrictedLinearProg* builds the LP problem similarly to *poseLinearProg* but using *sol* as a starting vertex and limiting the number of steps to one.

2.4 Avoiding repeated EFMs

As previously stated, one of the main problems of LP methods is the EFM found in every LP call is repeated. This is a common behavior that significantly decreases the efficiency of the methods used. This problem is exacerbated when using EFMs of length 2 (in the sense of corresponding to EFMs with a support of cardinality two), as due to their reduced size is highly probably to be repeated, and also they tend to have adjacent vertices that also are two length.

Therefore, we decided not to include a solution in our set of computed EFMs unless it has a length >2 (this excludes such solutions from the set of computed EFMs, but they can be easily recovered after the process is over). EFM-Ta favors this behavior using two measures: (i) only includes in the positive constraint those reactions that had not previously appeared in any computed solution of length 2 and, (ii) reactions coming from reversible ones are also put in the objective function, so the minimizing process would also try to exclude them.

Algorithm 2 describes our final EFM-Ta proposal.From Algorithm 2, EFM-Ta works in the following manner:

- Use the stoichiometric matrix in which all blocked reactions are removed.
- Start with an initially empty set of forbidden reactions.
- Choose a positive constraint to extract EFMs from this matrix, by randomly choosing a non-forbidden reaction and imposing that this must be equal to 1 (i.e. the obtained solution must include this reaction from this set).
- Use a similar method to construct the objective function by randomly choosing a set of not forbidden reactions. Additionally, all

Algorithm 1 Computing extra EFMs from the final simplex tableau

Data: Matrix S
Data: Integer *nEFMs*
Result: set *EFMs*

Function(*runExperiment(S)*)
$EFMs \leftarrow \emptyset$;
while *(|EFMs| < nEFMs)* **do**
  $f \leftarrow randomFunction(R)$;
  $cons \leftarrow randomReaction(Irr)$;
  $lp \leftarrow poseLinearProgram(f, S, cons)$;
  $\{sol\} \leftarrow simplex(lp)$;
  if *isNew(sol)* **then**
    $EFMs \leftarrow EFMs \cup \{sol\}$;
    $sop = supp(sol)$;
    $repeated = 0$;
    while *(repeated < threshold)* **do**
      $f \leftarrow randomFunction(sop)$;
      $lp \leftarrow poseRestrictedLinearProgram(f, S, cons, sol)$;
      $\{newSol\} \leftarrow simplex(lp)$;
      if *isNew(newSol)* **then**
        $EFMs \leftarrow EFMs \cup \{newSol\}$;
        $repeated = 0$;
      end
      else
        $repeated \leftarrow repeated + 1$;
      end
    end
  end
end
return *EFMs*;

Algorithm 2 Improvement of efficiency by discarding two-cycle EFMs

Data: Matrix S
Data: Integer *nEFMs*
Result: set *EFMs*

Function(*2cycleEFM(sol)*)
if $(length(supp(sol)) < 3)$ **then**
  return *TRUE*;
end
return *FALSE*;
Function(*runExperiment(S)*) $EFMs \leftarrow \emptyset$;
$forbidden \leftarrow \emptyset$;
while *(|EFMs| < nEFMs)* **do**
  $f \leftarrow randomFunction(R)$;
  $cons \leftarrow randomReaction(Irr \backslash forbidden)$;
  $lp \leftarrow poseLinearProg(f, S, cons)$;
  $\{sol\} \leftarrow simplex(lp)$;
  if $2cycleEFM(sol)$ **then**
    $forbidden \leftarrow forbidden \cup supp(sol)$;
  end
  else
    if *isNew(sol)* **then**
      $EFMs \leftarrow EFMs \cup \{sol\}$;
      $sop = supp(sol)$;
      $repeated = 0$;
      while *(repeated < threshold)* **do**
        $f \leftarrow randomFunction(sop)$;
        $lp \leftarrow poseRestrictedLinearProg(f, S, cons, sol)$;
        $\{newSol\} \leftarrow simplex(lp)$;
        if *isNew(newSol)* **then**
          $EFMs \leftarrow EFMs \cup \{newSol\}$;
          $repeated = 0$;
        end
        else
          $repeated \leftarrow repeated + 1$;
        end
      end
    end
  end
end
return *EFMs*;

the forbidden reactions have to be included in this function so if possible, the LP method will try to push these reactions out of the solution. To increase its randomness, also multiply each coefficient of this objective function by a random number (currently set to between 0 and 1).
- Solve the LP problem associated with this constraint and objective function.
- Use the techniques explained in Section 3.2 to find adjacent vertices to this solution. Save to a list those that are different from previously obtained ones. This process will stop if the number of consecutive repeated vertices obtained is greater than a given threshold.
- Each time a solution of length 2 is obtained, store the reactions involved in the set of forbidden reactions and discard this solution.

Remind that our approach can be used in any previously proposed method to obtain sets of EFMs by posing LP problems.

3 Results

This section shows the evaluation results of EFM-Ta using a specific case study.

3.1 Experimental framework

As suggested in Pey *et al.* (2015), a good unit to measure the efficiency of LP methods is the number of LP problems solved for one EFM obtained. This makes efficiency as independent as possible of the software, hardware and model chosen. However, as EFM-Ta does more things than just solving LP problems (it has to run the Algorithm 2), this section also shows the number of LPs, restricted LPs (RLPs), and the total time (s) of every experiment. To be able to compare our proposal with previous approaches, we define the efficiency rate as the number of solutions obtained by a unit of time defined as the (mean) time required to solve an LP problem.

Regarding our evaluation platform, this is equipped with a double socket Cascade Lake Xeon Gold 6238 (44 cores) @ 2.2 GHz with 384 GB of RAM. The system runs on a CentOS Linux 7.5, running CPLEX 12.10 version from IBM and Python 3.6.8 version from Intel.

As a case study, we have chosen three different network models available from BIGG models (Schellenberger *et al.*, 2010), ranging from small to medium-large sizes. The main model used is the reconstruction model for *Escherichia coli iAF*1260 (Feist *et al.*, 2007) with 2382 reactions and 1668 metabolites, being this model previously used in Pey *et al.* (2015). The two other models are the model for *Cricetulus griseus iCHOv*1 (Hefzi *et al.*, 2016) with 6663 reactions and 4456 metabolites and the Homo sapiens *Recon3D* (Brunk *et al.*, 2018) with 10 600 reactions and 5385 metabolites.

Finally, in the following experiments blocked reactions have been removed from the stoichiometric matrix.

3.2 Characterization of EFM-Ta

First, this section gives the results for our three models of the case study while extracting 100 000 different EFMs. Table 1 shows the number of LPs, restricted LPs (RLPs), total time (s) and efficiency rate (in number of LP per EFM) needed to compute 100 000 different EFMs in our three network models.

Next, we give a detailed information for the *iAF*1260 network model while extracting 1 000 000 different EFMs. Table 2 shows the total number of LPs, RLPs, total time (s) and efficiency rate (LP/EFM) at every 200 000 EFMs found, up to a 1 000 000 different EFMs.

For each iteration with a starting solution of length >2, EFM-Ta found between no new EFMs and a maximum of 24 859 (in just one iteration). The mean number of new EFMs obtained for iteration is 853.71 (with standard deviation of 3632.74).

Table 1. Characterization of EFM-Ta for computing 100 000 different EFMs in three network models

Model used	LPs	RLPs	Time (s)	Ef. rate (LP/EFM)
iAF1260	68	101 825	355	0.099
iCHOv1	84	103 462	1220	0.062
Recon3D	384	106 979	3879	0.067

Table 2. EFM-Ta characterization for *iAF*1260 network model for 1 000 000 different EFMs

No. of EFMs	LPs	RLPs	Time (s)	Eff. rate (LP/EFM)
200 000	100	205 781	672	0.098
400 000	537	414 107	1360	0.096
600 000	981	621 460	2045	0.095
800 000	1962	831 779	2763	0.095
1 000 000	2535	1 038 022	3445	0.094

Regarding the times reported, we have run EFM-Ta in a sequential way, that is, only one copy of the code is run. We expect that a parallel version of EFM-Ta would be faster and could take advantage of the number of cores that our testbed computer has, but this is out of the goal of this article and is postponed as future work.

From Tables 1 and 2, we can conclude that most of the computing effort needed to compute those sets of EFMs rely on solving RLPs (i.e. in computing adjacent vertices), and not in full LPs (just 0.24% of the problems solved). Note that solving an RLP is quite faster that doing so for a full LP.

One of the main problems when using random sampling as an extraction method is the number of repeated solutions obtained (Pey *et al.*, 2015). In this case, the rate of repeated solutions is <4.1%, as the number of LP and RLP problems needed to obtain 1 000 000 EFMs is of 1 040 557.

This leads to a very effective efficiency rate for EFM-Ta that is also very stable in time (as can be seen in Fig. 1).

Different experiments with this model show that this efficiency rate tends to remain stable in a value between 0.094 and 0.095 with mean of 0.948 and standard deviation of 0.01. Similar behavior has been observed in the other models used.

3.3 Comparing the efficiency with other tools

As pointed out in Pey *et al.* (2015), an exciting problem consists of obtaining EFMs that contains a target reaction in its support. As other tools have reported the number of EFMs obtained including target reactions, in this section we also do in this way to compare EFM-Ta with them.

First, EFM-Ta has to be relaxed to compute EFMs containing a specific target reaction. To do so, we fix the positive constraint to be the flux through the desired reaction and use just the objective function to avoid undesired solutions.

We have followed this approach to compute 2000 EFMs in the network model *iAF*1260 passing through reactions with lysine, threonine and arginine. Table 3 shows the final efficiency rate for different reactions in the network and compare them with those obtained with EFMEvolver (Kaleta *et al.*, 2009) and treeEFM (Pey *et al.*, 2015). EFM-Ta obtained slightly worse efficiency rates than in the general case, but still much higher than in previous approaches.

The efficiency rate highly depends on the reaction chosen. In all the studied cases, the efficiency rate tends to grow, stabilized at a specific rate that differs depending on the reaction. Graphs showing the evolution of efficiency while computing 20 000 EFMs passing through reactions with lysine, threonine and arginine can be found in Supplementary Material.

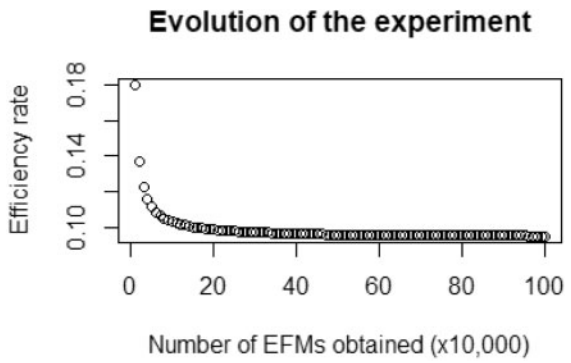


Fig. 1. Evolution of the efficiency rate during the whole experiment

Table 3. Comparison of efficiency rates (LP/EFM) for extracting 2000 EFMs including different target reactions

Eff. rate	EFMEvolver	treeEFM	EFM-Ta
Lysine	2.23	1.38	0.19
Threonine	1.90	1.64	0.16
Arginine	1.80	1.67	0.16

4 Conclusions and future work

We have presented a new method to obtain sets of EFMs called EFM-Ta. Its main difference with previous LP-based algorithms is found in the analysis of the final tableau, which enables us to obtain several solutions for each LP problem by performing additional simplex steps. EFM-Ta searches for new vertices invoking additional restricted LP problems with the same constraints but starting with the previously obtained solution, changing the objective function and limiting to one of the number of steps performed by the solver.

We have also analyzed the importance of two-cycle EFMs (that usually represent false EFMs and are caused by the imposition of positive constraints), and the negative impact in our method. This impact can be mitigated by using very simple heuristics combining both the positive constraint and objective function.

By extending EFM-Ta with the heuristics to avoid two cycles, we have implemented an algorithm that greatly breaks the ideal efficiency rate of 1 by a factor of more than 10×. We have also shown that this highly improved rate is very stable along the time.

Finally, we have compared EFM-Ta with other previously used tools when obtaining the sets of EFMs passing through a given target reaction. EFM-Ta manages it by using just the objective function and using the positive constraint to get the desired target. As expected, this produces a slight decrease in the efficiency rate (at least for some reactions), but the obtained rates remain high, and in all the studied cases, EFM-Ta obtains efficiency rates that are better than the previous ones in one order of magnitude.

As future work, we plan to explore the changes in the efficiency of EFM-Ta when applied to bigger models. The experiments performed indicate that the efficiency rates increase with the size of the model considered. We think that this can be a consequence of several characteristics of the model such as having a large number of reactions or the difference between this number and the rank of the stoichiometric matrix. It would be interesting to get a better understanding of this behavior so we can have an initial estimation of the efficiency of our method in different networks.

In the end, as for any extraction method based on random sampling, it is usually recommended a statistic analysis of the solutions obtained in order to avoid any bias produced by that method (see, e.g. Hidalgo *et al.*, 2018; Tabe-Bordbar and Marashi, 2013). Therefore, another interesting future work is to carry out that statistic analysis to check the diversity and quality of the EFMs found by EFM-Ta.

Funding

This work was partially funded by the AEI (State Research Agency, Spain) and the ERDF (European Regional Development Fund, EU) [RTI2018-098156-B-C53].

Conflict of Interest: none declared.

References

Acuña,V. *et al.* (2009) Modes and cuts in metabolic networks: complexity and algorithms. *Biosystems*, **95**, 51–60.

Arabzadeh,M. *et al.* (2018) A graph-based approach to analyze flux-balanced pathways in metabolic networks. *Biosystems*, **165**, 40–51.

Brunk,E. *et al.* (2018) Recon3d enables a three-dimensional view of gene variation in human metabolism. *Nat. Biotechnol.*, **36**, 272–281.

De Figueiredo,L.F. *et al.* (2009) Computing the shortest elementary flux modes in genome-scale metabolic networks. *Bioinformatics*, **25**, 3158–3165.

Feist,A.M. *et al.* (2007) A genome-scale metabolic reconstruction for *Escherichia coli* k-12 mg1655 that accounts for 1260 ORFs and thermodynamic information. *Mol. Syst. Biol.*, vol. **3**, no. 121, pp. 1-18

Fukuda,K. and Prodon,A. (1995) Double description method revisited. In: Deza,M. *et al.* (eds) *Combinatorics and Computer Science, Volume 1120 of Lecture Notes in Computer Science*. Springer, Berlin, Heidelberg, pp. 91–111.

Gagneur,J. and Klamt,S. (2004) Computation of elementary modes: a unifying framework and the new binary approach. *BMC Bioinformatics*, **5**: 175.

Gerstl,M. *et al.* (2015) Metabolomics integrated elementary flux mode analysis in large metabolic networks. *Sci. Rep*, **5**, 8930.

Hefzi,H. *et al.* (2016) A consensus genome-scale reconstruction of Chinese hamster ovary cell metabolism. *Cell Syst.*, **23**, 434–443.

Hidalgo,J.F. *et al.* (2015) A new approach to obtain EFMs using graph methods based on the shortest path between end nodes. In: Ortuño, F. and Rojas, I. (eds.) *Bioinformatics and Biomedical Engineering*, Vol. **9043** of *lnbi*. Springer International Publishing, Granada, Spain, pp. 641–649.

Hidalgo,J.F. *et al.* (2017) Representativeness of a set of metabolic pathways. In: Rojas, I. and Ortuño, F. (eds) *Bioinformatics and Biomedical Engineering*, Vol. **10208**. Springer International Publishing, Granada, Spain, pp. 659–667.

Hidalgo,J.F. *et al.* (2018) Improving the EFMs quality by augmenting their representativeness in LP methods. *BMC Syst. Biol.*, **12** (**Suppl. 5**), 101.

Hunt,K. *et al.* (2014) Complete enumeration of elementary flux modes through scalable demand-based subnetwork definition. *Bioinformatics*, **30**, 1569–1578.

Kaleta,C. *et al.* (2009) Efmevolver: Computing elementary flux modes in genome-scale metabolic networks. *Lect. Notes Inf.* P-157.

Klamt,S. and Stelling,J. (2003) Two approaches for metabolic pathway analysis? *Trends Biotechnol*, **21**, 64–69.

Klamt,S. *et al.* (2005) Algorithmic approaches for computing elementary modes in large biochemical reaction networks. *IEE Proc. Syst. Biol.*, **152**, 249–255.

Machado,D. *et al.* (2012) Random sampling of elementary flux modes in large-scale metabolic networks. *Bioinformatics*, **28**, i515–i521.

Pey,J. and Planes,F. (2014) Direct calculation of elementary flux modes satisfying several biological constraints in genome-scale metabolic networks. *Bioinformatics*, **30**, 2197.

Pey,J. *et al.* (2015) Treeefm: calculating elementary flux modes using linear optimization in a tree-based algorithm. *Bioinformatics*, **31**, 897–904.

Planes,F. and Beasley,F. (2008) A critical examination of stoichiometric and path-finding approaches to metabolic pathways. *Brief. Bioinform.*, **9**, 422–436.

Quek,L.-E. and Nielsen,L.K. (2014) A depth-first search algorithm to compute elementary flux modes by linear programming. *BMC Syst. Biol.*,**8**, **94**.

Rezola,A. *et al.* (2011) Exploring metabolic pathways in genome-scale networks via generating flux modes. *Bioinformatics*, **27**, 534–540.

Röhl,A. and Bockmayr,A. (2017) A mixed-integer linear programming approach to the reduction of genome-scale metabolic networks. *BMC Bioinformatics*, **18**, 2.

Röhl,A. *et al*. (2019) Computing irreversible minimal cut sets in genome-scale metabolic networks via flux cone projection. *Bioinformatics*, **35**, 2618–2625.

Schellenberger,J. *et al*. (2010) Bigg: a biochemical genetic and genomic knowledgebase of large scale metabolic reconstructions. *BMC Bioinformatics*, **11**, 213.

Schuster,S. and Hilgetag,C. (1994) On elementary flux modes in biochemical reaction systems at steady state. *J. Biol. Syst*., **2**, 165–182.

Schuster,S. *et al*. (1999) Detection of elementary flux modes in biochemical networks: a promising tool for pathway analysis and metabolic engineering. *Trends Biotechnol*., **17**, 53–60.

Schuster,S. *et al*. (2000) A general definition of metabolic pathways useful for systematic organization and analysis of complex metabolic networks. *Nat. Biotechnol*., **18**, 326–332.

Tabe-Bordbar,S. and Marashi,S. (2013) Finding elementary flux modes in metabolic networks based on flux balance analysis and flux coupling analysis: application to the analysis of *Escherichia coli* metabolism. *Biotechnol. Lett*., **35**, 2039–2044.

Taha,H. (2016) *Operations Research: An Introduction*, 10th edn. Prentice Hall, Upper Saddle River, NJ

Terzer,M. and Stelling,J. (2008) Large-scale computation of elementary flux modes with bit pattern trees. *Bioinformatics*, **24**, 2229–2235.