Model Reduction of Chemical Reaction Systems using Elimination

François Boulier, Marc Lefranc, François Lemaire, Pierre-Emmanuel Morant

Model Reduction of Chemical Reaction Systems using Elimination

François Boulier, Marc Lefranc, François Lemaire and Pierre-Emmanuel Morant

Abstract. There exist different schemes of model reduction for parametric ordinary differential systems arising from chemical reaction systems. In this paper, we focus on some schemes which rely on quasi-steady states approximations. We show that these schemes can be formulated by means of differential and algebraic elimination. Our formulation is simpler than the classical ones. It permitted us to obtain an approximation of the basic enzymatic reaction system which is different from those of Henri-Michaëlis-Menten and Briggs-Haldane.

1. Introduction

Chemical reaction systems provide a formalism to describe dynamical systems in contexts more general than the traditional chemically reacting mixtures. The dynamics of these systems is usually studied by translating them to systems of parametric ordinary differential equations by means of the mass action law. Observe that the strict setting of chemical reactions is usually not sufficient to describe many behaviours and needs to be generalized [15]. For instance, in the process of modeling gene regulatory networks, it is common to allow unbalanced reactions, or reactions which do not conserve the mass of the reactants to describe, among other phenomenons, mRNA translation or protein degradation. See e.g. [28].

The parametric ODE systems derived from these generalized chemical reaction systems are often too complicated for further analysis and need to be reduced. They are often overparameterized, which makes fitting methods to determine the parameters values heavy to carry out and unreliable. Moreover, for the important purpose of understanding which parts of the system contribute the most to some property of interest [17], it is preferable to deal with smaller systems.

For these reasons, model reduction of chemical reaction systems have become a central problem in their study, as explained in the survey [24]. Among all the existing model reduction methods (lumping, sensitivity analysis, ...) this paper is only concerned by the quasi-steady state approximation, which relies on the assumption that some of the chemical reactions are much faster than the other ones.

The idea of quasi-steady state approximation is simple: study the dynamics of the slow reactions, assuming that the fast ones are at quasi-equilibrium, thereby removing from the ODE system, the differential equations which describe the evolution of the variables at quasi-equilibrium. Many authors [27, 24, 29, 2] state that carrying out rigorously this approximation is far from straightforward. We would rather say that there are different ways to perform this approximation and that it is the problem of ascertaining the domain of validity of each kind of approximation which is not straightforward.

A basic quasi-steady state approximation may very well apply if one is only interested by a basic property of the model under study (e.g. the presence of a Poincaré-Andronov-Hopf bifurcation) possibly under some extra assumptions [28, 4, 8]. A more subtle one may be needed if one is interested by the timescales over which the system equilibrates or by the period and the amplitude of the oscillations, in the case of an oscillating system [2].

This paper focuses on the quasi-steady state approximations considered in [27, 29, 2], which are equivalent and which assume that reactions can be separated in only two categories: the "fast" ones and the "slow" ones (there are no, say, "very fast" or "medium" reactions). It shows how differential elimination methods [5, 6, 16] can be used to perform the model reduction. The presented method is brute force. A more subtle use of non differential elimination methods, based on [20], was implemented by the third author and will be described in a future paper. Observe however that this paper is not concerned by the problem of checking some validity conditions such as conditions C3 and C4 of [27].

We stress the fact that using automatic elimination methods makes the task of the modeller much easier. In particular, [27, 29] do not simplify their reduced dynamical systems by taking into account the conservation laws. Simplifying a dynamical system modulo algebraic equations is not easy to perform by hand while it is very important since it sometimes permits to decrease the number of variables of the resulting dynamical system.

As all model reduction methods, the quasi-steady state approximation permits to investigate the stability of biological systems, by symbolic methods, over a larger range of systems [31, 23]. The quasi-steady state approximation is however, specifically, a very interesting preparation process for further numerical integration, since it permits to avoid the presence of different timescales in ODE systems. As pointed out by Robertson in 1966, "*When the equations represent the behaviour of a system containing a number of fast and slow reactions, a forward integration of these equations becomes difficult*" cited in [13, sect. IV.1, Examples of Stiff Equations].

2. The method over the basic enzymatic reaction system

The following classical system of chemical reactions (1) and (2) describes the transformation of a substrate S into a product P under the action of the enzyme E. An intermediate complex C is produced.

$$E + S \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} C \tag{1}$$

$$C \xrightarrow{k_2} E + P \tag{2}$$

There is a straightforward way to transform this chemical system into a system of four parametric ordinary differential equations. The so obtained model (3) is often called the *initial model* of the chemical system:

$$\dot{E} = -k_1\,E\,S + k_{-1}\,C + k_2\,C \tag{3a}$$

$$\dot{S} = -k_1\,E\,S + k_{-1}\,C \tag{3b}$$

$$\dot{C} = k_1\,E\,S - k_{-1}\,C - k_2\,C \tag{3c}$$

$$\dot{P} = k_2\,C \tag{3d}$$

The hypothesis which permits to perform all the model reductions is that the reversible reaction (1) is much faster than the reaction (2). With other words, one assumes that $k_1,\, k_{-1} \gg k_2$.

Two different reductions of this chemical system were given in the early twentieth century by Henri, Michaëlis and Menten [14, 21] on the one hand, Briggs and Haldane [11] on the other hand.

2.1. The Henri-Michaëlis-Menten and Briggs-Haldane reductions

In all cases, the reduced model, which describes the evolution of the product concentration from the substrate concentration, writes:

$$\dot{P}(t) = \frac{V_m\,S(t)}{K + S(t)}, \qquad \dot{S}(t) = -\frac{V_m\,S(t)}{K + S(t)} \tag{4}$$

but with different assumptions and different values for the parameter K.

Both reductions rely on a few extra assumptions i.e. (the 0 subscript indicating initial concentrations) $S_0 \gg E_0$, $P \simeq 0$, $C_0 = 0$ [1, page 9]. In the case of the Henri-Michaëlis-Menten

reduction, one assumes equation (1) is constantly at equilibrium (the pre-equilibrium condition), which means $k_1\,C \simeq k_{-1}\,E\,S$. Thus $\dot{S}$ is assumed to be small. Performing some extra computations, one is led to $V_m = k_2\,E_0$ and $K = \frac{k_{-1}}{k_1}$. In the case of the Briggs-Haldane reduction, one assumes that the intermediate complex does not vary much i.e. that $\dot{C} \simeq 0$ which implies $k_1\,E\,S \simeq (k_{-1} + k_2)\,C$ (the quasi-stationary state hypothesis). Doing some further computations, one is led to $V_m = k_2\,E_0$ (as in the Henri-Michaëlis-Menten case) and $K = \frac{k_{-1}+k_2}{k_1}$.

2.2. Reduction using differential elimination

The first idea that a casual reader would have for performing such reductions could consist in translating the "$\simeq$" into a "=". In the Briggs-Haldane case, the hypothesis $\dot{C} \simeq 0$ would be treated by enlarging the initial model equations with the new equation $\dot{C} = 0$. However, this equation implies that C is a constant, which implies that S is a also a constant. The reduced model so obtained can certainly be considered as useless.

The right way to proceed consists in separating the "fast" variables from the "slow" ones and in transforming the ODE defining the evolution of each fast variable $\dot{x} = rhs$ by the algebraic equation $0 = rhs$. But what is a "fast" variable ? An intuitive definition would be: a variable is fast if it is involved in a fast chemical reaction. Over some examples [28] this definition is sufficient. However, it would clearly lead to a useless reduced model over our example since E, S and C would then be considered as fast variables and one would end up with the reduced model $\dot{P} = \dot{S} = 0$. As explained in [2, page 3502], a variable x such that $\dot{x}$ depends on a mixture of slow and fast reactions should not be considered as slow. The fast and slow variables of a system are not necessarily the model variables but functions of them. They can be computed by a rigorous two timescales analysis, as shown by [27] but, as stated by [29, 2], this is not always necessary.

The reduction that we present here is equivalent to those of [27, 29, 2] but differs in its presentation. As for the Henri-Michaëlis-Menten reduction, one uses the pre-equilibrium approximation on the first reaction but this is (for the moment) the only assumption we make:

$$k_1\,E\,S = k_{-1}\,C \tag{5}$$

As for the initial model, one translates the two chemical reactions as parametric ordinary differential equations expressing the evolution of the concentrations of E, S, C and P. However one handles the reactions which are assumed to be at equilibrium (the fast reactions) in a different manner than the other reactions (the slow reactions): *the variation of the concentration of each chemical species is the sum of the contributions of each reaction. The contribution of each slow reaction is derived from the law of mass action as usual. The contribution of each fast reaction is represented by a new variable.*

The underlying idea is that the dynamic of the fast reaction is (for the moment) considered as unknown. This means that one adds a degree of freedom for each fast reaction. This freedom in fact allows the system to stay on the equilibrium conditions. Performing elimination on the extra unwanted variables, one gets a set of differential equations in the reactants only. Over the example, the contribution of the fast reaction is denoted F_1 (for fast reaction 1) and the contribution of the slow reaction is $k_2\,C$:

$$\dot{E} = -F_1 + k_2\,C \tag{6a}$$

$$\dot{S} = -F_1 \tag{6b}$$

$$\dot{C} = F_1 - k_2\,C \tag{6c}$$

$$\dot{P} = k_2\,C \tag{6d}$$

Using elimination on (5) and (6) with the *ranking* $\mathscr{R} = [F_1] \gg [P, C, E, S]$ [19, chap. I, §8], one eliminates the unknown F_1. Elimination is performed below using the MAPLE *diffalg* package. The parameters are put in the base field $\mathscr{F}$ of the equations to avoid discussions over their values (they are then considered as algebraically independent). An inequation is added to avoid considering the degenerate case of $C(t)$ being the zero function. The output is pretty printed.

```
with(diffalg):
sys := [ E[t] - (-F1 + k2*C), S[t] - (-F1), C[t] - (F1 - k2*C), P[t] - k2*C,
         k1*E*S - km1*C ]:
F := field_extension (transcendental_elements=[k2,k1,km1]):
R := differential_ring (ranking=[ [F1], [C,E,P,S] ],
                        derivations=[t], field_of_constants=F):
Ineqs := [ C ]:
Ids := Rosenfeld_Groebner (sys,Ineqs,R);
                             Ids := [characterizable]

rules := rewrite_rules( Ids[1] );
```

$$\left[F_1 = \frac{k_2\,k_1\,E\,S\,(k_1\,S + k_{-1})}{k_{-1}\,(k_1\,S + k_1\,E + k_{-1})},\quad \dot{E} = \frac{{k_1}^2 E^2 k_2\,S}{k_{-1}\,(k_1\,S + k_1\,E + k_{-1})},\quad \dot{P} = \frac{k_2\,k_1\,E\,S}{k_{-1}},\right.$$

$$\left.\dot{S} = -\frac{k_2\,k_1\,E\,S\,(k_1\,S + k_{-1})}{k_{-1}\,(k_1\,S + k_1\,E + k_{-1})},\quad C = \frac{k_1\,E\,S}{k_{-1}}\right]$$

In the above system, one gets the value of F_1. The reduced model is made of three parametric ordinary differential equations. It is exactly that obtained by [29, page 2327, (33), denoting A, B, C, D, κ for S, E, C, P, k_1/k_{-1}].

2.3. Refinements

One can simplify further the reduced model by replacing the two parameters k_1 and k_{-1} by a single parameter $K = k_{-1}/k_1$. Indeed, k_1 and k_{-1} only appear in the equation (5) for which one can substitute $K = k_{-1}/k_1$. This simplification makes sense: the values of k_{-1} and k_1 describe the speed of the fast reaction. Since this reaction is assumed to be constantly at the equilibrium, only the ratio of the two constants matters. Continuing the example, one gets:

```
map (normal, subs (km1=K*k1, rules));
```

$$\left[F_1 = \frac{k_2\,E\,S\,(S+K)}{K\,(S+E+K)},\quad \dot{E} = \frac{k_2\,E^2\,S}{K\,(S+E+K)},\quad \dot{P} = \frac{k_2\,E\,S}{K},\quad \dot{S} = -\frac{k_2\,E\,S\,(S+K)}{K\,(S+E+K)},\quad C = \frac{E\,S}{K}\right].$$

The second simplification consists in using the two conservations laws below (that one can automatically deduce from the system (6) by means of linear algebra):

$$E + C = E_0 + C_0 \tag{7a}$$

$$S + C + P = S_0 + C_0 + P_0 \tag{7b}$$

plus the hypothesis $C_0 = P_0 = 0$ (as for the classical reductions). One can then eliminate E from the ODE describing the evolution of S. This can be done by means of elimination. For instance, one can rerun the above commands, enlarging the field $\mathscr{F}$ with the two new symbols E_0 and S_0 and the list *sys* with the two equations $E + C = E_0$ and $S + C + P = S_0$. Performing the same elimination as above and denoting $k_2\,E_0 = V_m$ as usual, one eventually gets:

$$\dot{S} = -\frac{V_m\,S\,(K+S)}{K\,E_0 + (K+S)^2}. \tag{8}$$

As far as we know, this formula is new. If one neglects the term $K\,E_0$ in equation (8), one gets equation (4). Thus equation (8) and (4) are almost equal when the term $K\,E_0$ is negligible, for example when $S \gg E_0$. What happens when S and E_0 are in the same range, a phenomenon likely to occur *in vivo* according to [12, §2.1.1] ? Equation (8) turns out to be more precise, Figure 1 shows.

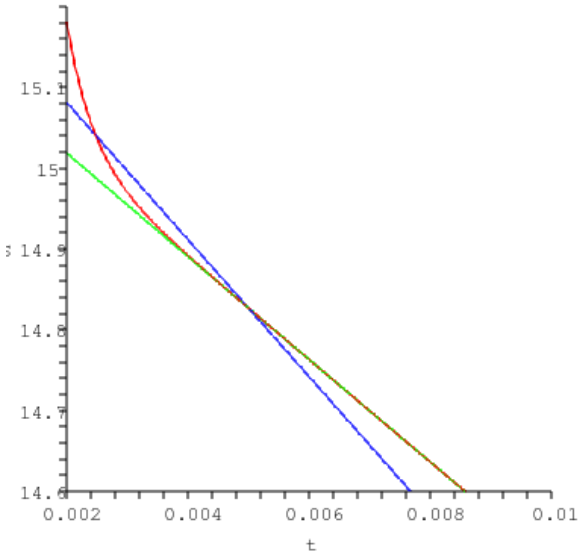

FIGURE 1. When the initial enzyme concentration is close to that of the substrate, the curve $S(t)$ computed from the initial model (3) (red, coming from the top of the diagram) is better approximated by formula (8) (green) than by formula (4) (blue). The two reduced formulas were integrated for an initial value picked on the non reduced curve, after the transient step, at $t = 0.005$. Parameters values were borrowed from [29].

3. The algorithm

The main algorithm DifferentialModelReduction is composed of two main steps: the transformation of the list of reactions into a dynamical system, and the elimination part. Its main interest is that it is fully automatic and performs automatic case splitting when needed.

The DifferentialModelReduction algorithm described in section 3.2 relies on a few auxiliary functions described in section 3.1. In all algorithms presented in this section, the parameter L denotes a list of chemical reactions, and X denotes a vector of all chemical species of L (i.e. all the reactants and products involved in L). Except in the function DifferentialModelReduction, the order of the elements of X is not important. As explained later in section 3.2, the order of X encodes the chemical species to be eliminated.

Each reaction is represented by a list of reactants, a list of products, a boolean flag indicating if the reaction is considered fast, a boolean flag indicating if the reaction is reversible, and one (or two) rate constant(s) (depending on whether the reaction is reversible or not). A fast reaction can be as well reversible as irreversible. Reactants and products are represented by lists with two elements: a symbol and a stoichiometric coefficient. For example, the two reactions (1) and (2) in section 2 could be coded in MAPLE, in the humanly readable format:

```
[ reactants=[ [E,1], [S,1] ], products=[ [C,1] ],
  rateConstants=[k1, km1], fast=true, reversible=true ];

[ reactants=[ [C,1] ], products=[ [E,1], [P,1] ],
  rateConstants=[k2], fast=false, reversible=false ]
```

Observe that, in our algorithm, a reversible reaction is not equivalent to two irreversible reactions. The task of indicating that a reaction is reversible is left to the user. It could be made automatic by computing the rank of a stoichiometry matrix as [29, page 2323].

A basic way to define slow and fast reactions consists in splitting the set of rate constants into two sets: the large ones and the small ones. A reaction is considered as fast if it involves at least one large rate constant.

3.1. The auxiliary functions

In this section, one denotes a reaction as

$$\sum a_j X_j \xrightarrow{k_1} \sum b_j X_j \text{ if } R \text{ is irreversible, otherwise } \sum a_j X_j \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \sum b_j X_j.$$

This formulation involves all the chemical species present in the list of reactions, by setting $a_j = 0$ (resp. b_j=0) when X_j is not a reactant (resp. a product).

The function StoichiometricMatrix**.** It computes the stoichiometric matrix associated to a list of reactions L (see [18]).

Algorithm 1 StoichiometricMatrix(L, X)

Require: a list of reactions $L = [l_1, \ldots, l_p]$, a vector $X = {}^t(X_1, \ldots, X_n)$ of all reactants and products involved in L
Ensure: the stoichiometric matrix associated to L
1: let $M = (m_{ji})_{1 \leq j \leq n,\ 1 \leq i \leq p,}$
2: **for** each reaction l_i **do**
3: denote l_i as $\sum a_j X_j \xrightarrow{k_1} \sum b_j X_j$ if R is irreversible, otherwise $\sum a_j X_j \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \sum b_j X_j$
4: $m_{ji} := b_j - a_j$ for each $1 \leq j \leq n$
5: **end for**
6: **return** M

The function Equilibria**.** It returns a list of algebraic relations ensuring that the fast reactions are at pre-equilibrium. On our example in section 2, it simply returns $[k_1 ES - k_{-1} C]$, which is the equation (5).

Algorithm 2 Equilibria(L, X)

Require: a list of reactions L, a vector $X = {}^t(X_1, \ldots, X_n)$ of all reactants and products involved in L
Ensure: the list of equilibrium relations for the fast reactions
1: let $L_f = [l_1, , \ldots, l_p]$ be the list of the fast reactions of L
2: ▷ let us first construct the rate vector V_f for the fast reactions
3: let $V_f = {}^t(v_1, \ldots, v_p)$
4: **for** each reaction l_i of L_f **do**
5: denote l_i as $\sum a_j X_j \xrightarrow{k_1} \sum b_j X_j$ if R is irreversible, otherwise $\sum a_j X_j \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \sum b_j X_j$
6: **if** l_i is irreversible **then**
7: $v_i := k_1 \prod {X_j}^{a_j}$
8: **else**
9: $v_i := k_1 \prod {X_j}^{a_j} - k_2 \prod {X_j}^{b_j}$
10: **end if**
11: **end for**
12: ▷ let us compute the stoichiometric matrix for the fast reactions
13: $M_f :=$ StoichiometricMatrix(L_f, X)
14: $res := M_f \cdot V_f$
15: convert res into a list and remove duplicate entries
16: **return** res

Algorithm 3 ListReactionsToODE(L, X)

Require: $L = [l_1, \ldots, l_p]$ is a list of reactions, $X = {}^t(X_1, \ldots, X_n)$ is a vector of all reactants and products involved in L

Ensure: a dynamical system $\dot{X} = H(X, \Theta, F)$ describing L, where Θ is the set of the rate constants, and F is the set of the fast variables F_i's.

▷ let us first construct the vector of rate V
let $V = {}^t(v_1, \ldots, v_p)$
for each reaction l_i **do**
    denote l_i as $\sum a_j X_j \xrightarrow{k_1} \sum b_j X_j$ if R is irreversible, otherwise $\sum a_j X_j \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \sum b_j X_j$
    if l_i is a fast reaction **then**
        $v_i := F_i$
    else
        if l_i is irreversible **then**
            $v_i := k_1 \prod {X_j}^{a_j}$
        else
            $v_i := k_1 \prod {X_j}^{a_j} - k_2 \prod {X_j}^{b_j}$
        end if
    end if
end for
▷ let us compute the stoichiometric matrix
$M :=$ StoichiometricMatrix(L, X)
return $\dot{X} = M \cdot V$

The function ListReactionsToODE**.** It returns a dynamical system $\dot{X} = H(X, \Theta, F)$, where Θ denotes the set of all rate constants in the list of reactions, and F denotes the set of the extra variables F_i introduced for the fast reactions. On our example in section 2, if L is the list composed of the two reactions (1) and (2) (in that order), and if $X = {}^t(E, S, C, P)$, then the vector V and the matrix M computed inside ListReactionsToODE are

$$V = \begin{pmatrix} F_1 \\ k_2 C \end{pmatrix} \quad M = \begin{pmatrix} -1 & 1 \\ -1 & 0 \\ 1 & -1 \\ 0 & 1 \end{pmatrix}$$

Thus ListReactionsToODE returns

$$\dot{X} = \begin{pmatrix} \dot{E} \\ \dot{S} \\ \dot{C} \\ \dot{P} \end{pmatrix} = \begin{pmatrix} -1 & 1 \\ -1 & 0 \\ 1 & -1 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} F_1 \\ k_2 C \end{pmatrix} = \begin{pmatrix} -F_1 + k_2 C \\ -F_1 \\ F_1 - k_2 C \\ k_2 C \end{pmatrix}$$

which is exactly the system (6).

The ConservationLaws **function.** It aims at finding linear combinations of the X_i which are constant along solutions. To compute them, one just needs to compute a basis of the kernel of ${}^tM$ (i.e. a basis of the solutions of ${}^tM \cdot Y = 0$). This is a linear algebra problem over $\mathbb{Q}$. The argument is straightforward: if y satisfies ${}^tM \cdot y = 0$, then ${}^t y \cdot \dot{X} = {}^t({}^tM \cdot y) \cdot V = 0$, so ${}^t y \cdot X$ is constant on any solution, which implies ${}^t y \cdot X$ is a conservation law. See also [18].

On the example of section 2, a basis of the kernel of ${}^tM$ is for example ${}^t(-1, 1, 0, 1)$ and ${}^t(1, 0, 1, 0)$. This leads to $-E + S + P = -E_0 + S_0 + P_0$ and $E + C = E_0 + C_0$. Although different to the two conservation laws (7), one easily checks that they are equivalent. If instead, one takes the other basis ${}^t(0, 1, 1, 1)$ and ${}^t(1, 0, 1, 0)$, one exactly obtains the conservation laws (7).

Algorithm 4 ConservationLaws(L, X)

Require: a list of reactions L, a vector X of all reactants and products involved in L
Ensure: a list of (linear) conservation laws
1: $M :=$ StoichiometricMatrix(L, X)
2: compute a basis $[y_1, \ldots, y_s]$ of ${}^tM \cdot Y = 0$
3: **return** the list $[{}^ty_1 \cdot X, \ldots, {}^ty_s \cdot X]$

3.2. The DifferentialModelReduction algorithm

The function DifferentialModelReduction**.** It transforms a list of reactions into a list of dynamical systems encoding the reduced models. The chemical species to be eliminated are to be put first in the vector X. The number of chemical species that can be eliminated depends on several criteria: the more fast reactions and conservation laws there are, the more chemical species get eliminated. However, this number also depends on the form of the equations themselves, as shown in section 4.1.

Algorithm 5 DifferentialModelReduction(L, X)

Require: a list of reactions L, a vector $X = {}^t(X_1, \ldots, X_n)$ of all reactants and products involved in L
Ensure: a list of reduced dynamical systems in the variables X
1: $Sys :=$ ListReactionsToODE(L, X)
2: $Cons :=$ ConservationLaws(L, X)
3: $Equi :=$ Equilibria(L, X)
4: $S := Sys \cup Cons \cup Equi$
5: $\mathscr{R}$ the ranking $[F] \gg [X]$ where F is the set of the unknowns F_i introduced for each fast reaction
6: $[C_1, \ldots, C_t] :=$ Rosenfeld-Gröbner$(S, \mathscr{R})$
7: if NormalForm(F_i, C_k) is not a rational fraction in the variables X (for all F_i and all $1 \leq k \leq t$) then *FAIL*.
8: **return** the list $[S_1, \ldots, S_t]$
9: where each S_i is equal to $[\dot{X}_1 =$ NormalForm$(\dot{X}_1, C_i), \ldots, \dot{X}_n =$ NormalForm$(\dot{X}_n, C_i)]$

After building the system S of equations, DifferentialModelReduction builds the *ranking* $\mathscr{R}$ to be used by Rosenfeld-Gröbner over the system S. The ranking $[F] \gg [X]$ means that all derivatives of F are greater than those of X, the derivatives of F are ordered w.r.t. an *orderly* ranking, and the derivatives of X also.

The algorithm Rosenfeld-Gröbner performs the so called differential elimination. It returns a list of systems of polynomial differential equations called *characteristic sets*. Each characteristic set C_i defines some *differential ideal* $\mathfrak{A}_i$ in the *differential polynomial ring* R which contains the model equations. Using the set C_i, one is able to perform computations (such as the normal form computation) modulo the ideal $\mathfrak{A}_i$. The italicized words above come from the *differential algebra* [25, 19, 30].

Applied to any differential polynomial p and C_i, the NormalForm algorithm returns a rational fraction which is a canonical representative of the residue class of p in the factor ring $R/\mathfrak{A}$. See [9] or [10, Figure 2] for a presentation of the NormalForm algorithm[1]. In our case, the goal of NormalForm is straightforward: it consists in expressing each derivative $\dot{X}_i$ or each unknown F_j as a rational fraction in the variables X only. If this goal is reachable, the normal form function performs it, thanks to the rankings properties.

Rosenfeld-Gröbner returns a list of characteristic sets because it is a case splitting algorithm, so it may need to consider different cases (say) $X_j = 0$ and $X_j \neq 0$ for some j. Such splittings

[1] In very rare cases, the NormalForm algorithm may split the characteristic set C_i as finitely many smaller characteristic sets, refining thereby the list returned by Rosenfeld-Gröbner.

might introduce some spurious outputs, since the case $X_j = 0$ is usually not very interesting. Indeed in section 2, the case $C = P = E = S = 0$ is not interesting. However, those splittings are needed when fast irreversible reactions are involved. For example, if one assumes that the reaction $A + B \xrightarrow{k} C$ is fast, then the pre-equilibrium hypothesis yields $kAB = 0$, which implies $A = 0$ or $B = 0$. Those two cases need to be treated differently since the dynamic in each case is different.

Over our example in section 2, the function DifferentialModelReduction (up to the renaming $K = k_{-1}/k_1$) performs the same computations as in sections 3.2 and 2.3.

3.3. Failures

Strictly speaking, it may happen that some unknowns F_i cannot be expressed as rational fractions of the variables X. In this case, our algorithm fails. Examples for which our method fails include many examples which make no sense (e.g. fast unbalanced reactions such as $A \to A + B$). A meaningful example for which our method does not readily apply is given by the Robertson's example [13, chap. IV.1] when the fast and the very fast reactions are considered both fast.

Translating our technical condition in terms of properties of the chemical reaction systems is an interesting problem, left for a future paper. Observe that our condition plays a role equivalent to the *"kinetic" linear independence* condition combined to the singularity testing for algebraic equations described in [29, pages 2324-2326] but our method seems to be more straightforward: it can be directly formulated in the setting of algebraic geometry (ideal theory) instead of a matrix inversion problem over a residue class ring [29, eq. (27)].

Sufficient conditions for the two methods to apply are described in [29, page 2326] (e.g. case of fast reactions with pairwise disjoint sets of reactants and products).

3.4. Optimisations

First, as said in section 2, one may decrease the number of rate constants in the reduced model. Indeed, the system returned by the Equilibria can usually be simplified by replacing some constants by their ratio or their sum. This can be done automatically using symmetry technics as in [26]. As a particular case, if all the fast reactions are reversible, one can introduce a new symbol $K = k_2/k_1$ for each fast reaction $\sum a_j X_j \underset{k_2}{\overset{k_1}{\rightleftharpoons}} \sum b_j X_j$. To handle this algorithmically, one would need to modify the function Equilibria so that it also returns a description of the new K symbols.

Second, although the differential elimination is conceptually simpler, it appears that the elimination process is mainly a linear algebra problem over some residue class ring. This can be achieved by using regular chains and regularity tests, as can be done with the RegularChains [20] package (and its subpackage MatrixTools) in MAPLE. Here is some insight of this claim (another paper will discuss that point).

Over the example of section 2, differentiate (5):

$$k_1 \dot{E} S + k_1 E \dot{S} = k_{-1} \dot{C} \tag{9}$$

Putting (6) and (9) together, one gets a system of five linear equations in F_1, $\dot{E}$, $\dot{S}$, $\dot{C}$ and $\dot{P}$, over the residue class of $K[E, S, C, P, k_1, k_{-1}, k_2]$ by (5). This system can be solved w.r.t. F_1. Taking conservation laws into account preserves the linearity of the problem.

4. Examples

In section 4.1, we carry out a basic polymer degradation system. We checked that our method applies to more complicated systems such as a polymerisation cascade or a variant of the ozone decomposition system (dropping temperature considerations) [29, page 2328].

4.1. Polymer degradation

One considers a simple dimerisation between a protein P and itself. One also assumes that the protein is degraded. The dimerisation is assumed to be fast, the degradation being slow.

$$2P \quad \underset{k_{-1}}{\overset{k_1}{\rightleftharpoons}} \quad P_2 \tag{10a}$$

$$P \quad \xrightarrow{k_2} \quad \varnothing \tag{10b}$$

Let us introduce $K = k_1/k_{-1}$ and take $X = {}^t(P, P_2)$. If L represents the two reactions (10a) and (10b), the call to DifferentialModelReduction(L, X) performs the following computations. The list Sys simply consists of

$$\dot{P} = -2F_1 - k_2 P \tag{11a}$$

$$\dot{P}_2 = F_1 \tag{11b}$$

The list $Cons$ is empty, and $Equi$ is $[KP^2 - P_2]$. Let us detail the computations using *diffalg*:

```
Sys := [ P[t] - (-2*F1 - k2*P), P2[t] - F1 ];
Cons := [];
Equi := [ K*P^2-P2];
F := field_extension (transcendental_elements=[K,k2]):
R := differential_ring (ranking=[ [F1], [P,P2] ],
                        derivations=[t], field_of_constants=F):
Ineqs := [ P ]:
Ids := Rosenfeld_Groebner ( [op(Sys),op(Cons),op(Equi)] ,Ineqs,R);
rules := rewrite_rules( Ids[1] );
```

$$\left[F_1 = -2\,\frac{k_2\,P_2\,(4\,KP - 1)}{16\,KP_2 - 1},\quad \dot{P}_2 = -2\,\frac{k_2\,P_2\,(4\,KP - 1)}{16\,KP_2 - 1},\quad P^2 = \frac{P_2}{K}\right]$$

Computing the normal forms of $\dot{P}$ and $\dot{P}_2$, one gets:

$$\dot{P} = -\frac{k_2\,P_2\,(4\,KP - 1)}{KP\,(16\,KP_2 - 1)} \tag{12a}$$

$$\dot{P}_2 = -2\,\frac{k_2\,P_2(4\,KP - 1)}{16\,KP_2 - 1} \tag{12b}$$

In this case, one could not eliminate P since it still appears in the equation (12b). However, over this simple example, one could eliminate P by using the (non algebraic) relation $P = \sqrt{P2/K}$. This would lead to:

$$\dot{P}_2 = -\frac{2k_2P_2}{4\sqrt{KP_2} + 1} \tag{13}$$

However, this cannot be generalised easily (since an algebraic equation cannot be symbolically solved when its degree is greater than 5). If one changes the order and takes $X = {}^t(P_2, P)$, one obtains a different system:

$$\dot{P}_2 = -\frac{2k_2KP^2}{4KP + 1} \tag{14a}$$

$$\dot{P} = -\frac{k_2P}{4KP + 1} \tag{14b}$$

In that case, P_2 is eliminated.

4.2. A single gene regulated by a polymer of its own protein

A model of a single gene regulated by an order n polymer of its own protein is provided in [8]. The model is composed of $n + 3$ chemical species and $2\,n - 5$ parameters. The polymerisations are supposed to be fast. A quasi-steady state approximation was performed over this model, yielding a reduced model made of 3 parametric ODE. A result concerning the absence of oscillations was then obtained over the reduced model, by means of symbolic computations.

The algorithm presented in this paper was afterwards applied over this generic model for increasing values of n. The naive DifferentialModelReduction implementation, based on the MAPLE *diffalg* package, works for any $n \leq 6$. The improved implementation, based on the MAPLE *RegularChains* package (see section 3.4) achieves the same task in a few seconds for each $n \leq 20$. The obtained reduced model [7] was more complicated than that of [8] but simple enough to be studied by means of symbolic computations: the results obtained in [8] still hold. The new reduced model is a more general approximation than the old one.

This example proves that the awful worst case complexity [3, chapter 9, Proposition 44] of differential elimination methods does not hold in the context addressed in this paper and that one may expect to process accurate reduced models by means of symbolic computations methods.

4.3. Equivalence of our method with [29, 27, 2]

Our method is essentially the same as that of [29]. The main differences are: we do not introduce any ϵ but the unknown F_i instead, we perform differential elimination instead of linear algebra and we directly handle reversible reactions instead of splitting them into two irreversible reactions (this affects the rate vector in the ListReactionsToODE function).

The above comments also apply to [27] since this paper is a basis for [29]. The paper [27] considers more general systems with external supply and withdrawal of species. For this reason, they do not obtain our formula (8) which is a consequence of conservation laws which do not exist in this general setting. They provide a strong theoretical basis for all methods. They transform the initial system into the so called two-time scale standard form by expliciting the true slow variables in order to apply the Tikhonov theorem. Our method (as well as that of [29]) is equivalent but more direct since it avoids this change of coordinates.

Our method is also essentially the same as that of [2]. The paper [2] presents an approximation scheme using the so called *prefactors*. Though we do not introduce this concept, our algorithm DifferentialModelReduction gives *exactly* the same reduced systems as [2] for the examples listed in [2, tables 1 and 2]. A major difference is that our method is fully algorithmic: it does not need to explicit the true slow variables and relies on elimination to perfom the computations. On the opposite, [2] does need to explicit the true slow variables (which might actually be computed automatically, following [29] for example). Computations are then performed by hand.

In order to illustrate the equivalence of our reduction method with that of [2], let us consider a simplified version of the example provided in [2, tables 1]: the case of the simple dimerisation given in section 4.1.

The argumentation of [2] introduces n_P the total number of proteins P, which is simply $n_P = P + 2\,P_2$ (since a dimer P_2 contains two proteins P). This new variable is in fact a slow variable, because this quantity does not change when the fast reaction (the dimerisation) occurs. One easily deduces the three equations:

$$P_2 = KP^2 \tag{15a}$$

$$n_P = P + 2\,P_2 \tag{15b}$$

$$\dot{n}_P = -k_2 P \tag{15c}$$

whereas our approach introduces:

$$P_2 = KP^2 \tag{16a}$$

$$\dot{P} = -2F_1 - k_2 P \tag{16b}$$

$$\dot{P}_2 = F_1 \tag{16c}$$

Although based on different variables, systems (15) and (16) are in fact very similar since they both imply:

$$P_2 = KP^2 \tag{17a}$$

$$\dot{P} + 2\dot{P}_2 = -k_2 P \tag{17b}$$

Indeed equation (17b) is obtained either by combining (16b) and (16c), or by combining (15b) and (15c). Since system (17) implies $\dot{P} = -\frac{k_2 P}{4KP+1}$, both systems (15) and (16) also do.

5. Conclusion

We have presented an algorithmic method for reducing systems of chemical reactions using elimination and we have shown the relationship between the approaches of [29, 27, 2] and ours.

The main interest of our method is that it is fully automated (once the set of fast reactions has been chosen). It is not restricted to academic examples [7]. Moreover the setting of our algorithm could easily be integrated in software based on the SBML format since recent versions of SBML (level 2 version 3 for example) include the flags `reversible` and `fast` that we need in order to apply our method. Still in this computational context, the reduction of the number of rate constants based on symmetries [26] could also be used.

The replacement of the differential elimination step by linear algebra methods over residue class rings is left for a future paper as well as a formulation of the failure conditions in chemical terms.

The application of DifferentialModelReduction to the systems of [28, 22] and the analysis of the reduced models are also left for a future paper.

References

[1] Bellon, B. Biochimie Structurale: introduction à la cinétique enzymatique. `http://www.up.univ-mrs.fr/wabim/d_agora/d_biochimie/cinetique.pdf`, 2006. in French.

[2] Bennet, M. R., Volfson, D., Tsimring, L., and Hasty, J. Transient Dynamics of Genetic Regulatory Networks. *Biophysical Journal 92* (May 2007), 3501–3512.

[3] Boulier, F. Réécriture algébrique dans les systèmes d'équations différentielles polynomiales en vue d'applications dans les Sciences du Vivant, May 2006. Mémoire d'habilitation à diriger des recherches. Université Lille I, LIFL, 59655 Villeneuve d'Ascq, France. `http://tel.archives-ouvertes.fr/tel-00137153`.

[4] Boulier, F. Differential Elimination and Biological Modelling. *Radon Series on Computational and Applied Mathematics (Gröbner Bases in Symbolic Analysis) 2* (October 2007), 111–139. `http://hal.archives-ouvertes.fr/hal-00139364`.

[5] Boulier, F., Lazard, D., Ollivier, F., and Petitot, M. Representation for the radical of a finitely generated differential ideal. In *ISSAC'95: Proceedings of the 1995 international symposium on Symbolic and algebraic computation* (New York, NY, USA, 1995), ACM Press, pp. 158–166. `http://hal.archives-ouvertes.fr/hal-00138020`.

[6] Boulier, F., Lazard, D., Ollivier, F., and Petitot, M. Computing representations for radicals of finitely generated differential ideals. *Applicable Algebra in Engineering, Communication and Computing 20*, 1 (2009), 73–121. (1997 Techrep. IT306 of the LIFL).

[7] Boulier, F., Lefranc, M., Lemaire, F., and Morant, P.-E. Applying a rigorous quasi-steady state approximation method for proving the absence of oscillations in models of genetic circuits. In *Proceedings of Algebraic Biology 2008* (2008), K. H. et al., Ed., no. 5147 in LNCS, Springer Verlag Berlin Heidelberg, pp. 56–64.

[8] Boulier, F., Lefranc, M., Lemaire, F., Morant, P.-E., and Ürgüplü, A. On proving the absence of oscillations in models of genetic circuits. In *Proceedings of Algebraic Biology 2007* (2007), K. H. H. Anai and T. Kutsia, Eds., vol. 4545 of *LNCS*, Springer Verlag Berlin Heidelberg, pp. 66–80. `http://hal.archives-ouvertes.fr/hal-00139667`.

[9] Boulier, F., and Lemaire, F. Computing canonical representatives of regular differential ideals. In *ISSAC'00: Proceedings of the 2000 international symposium on Symbolic and algebraic computation* (New York, NY, USA, 2000), ACM Press, pp. 38–47. `http://hal.archives-ouvertes.fr/hal-00139177`.

[10] Boulier, F., and Lemaire, F. A Normal Form Algorithm for Regular Differential Chains. *Mathematics in Computer Science 4*, 2 (2010), 185–201. 10.1007/s11786-010-0060-3.

[11] Briggs, G. E., and Haldane, J. B. S. A note on the kinetics of enzyme action. *Biochemical Journal 19* (1925), 338–339. available on `http://www.biochemj.org/bj/019/0338/bj0190338_browse.htm`.

[12] Crampin, E. J., Schnell, S., and Mac Sharry, P. E. Mathematical and computational techniques to deduce complex biochemical reaction mechanisms. *Progress in Biophysics & Molecular Biology 86* (2004), 77–112.

[13] Hairer, E., and Wanner, G. *Solving ordinary differential equations II. Stiff and Differential–Algebraic Problems*, 2nd ed., vol. 14 of *Springer Series in Computational Mathematics*. Springer–Verlag, New York, 1996.

[14] Henri, V. *Lois générales de l'Action des Diastases*. Hermann, Paris, 1903.

[15] Horn, F., and Jackson, R. General mass action kinetics. *Archive for Rational Mechanics and Analysis 47* (1972), 81–116.

[16] Hubert, É. Factorization free decomposition algorithms in differential algebra. *Journal of Symbolic Computation 29*, 4,5 (2000), 641–662.

[17] Kell, D. B., and Knowles, J. D. The Role of Modeling in Systems Biology. In *System Modeling in Cellular Biology: From Concepts to Nuts and Bolts* (2006), Z. Szallasi, J. Stelling, and V. Periwal, Eds., Cambridge, Massachussets: The MIT Press, pp. 3–18.

[18] Klamt, S., and Stelling, J. Stoichiometric and Constraint-based Modeling. In *System Modeling in Cellular Biology: From Concepts to Nuts and Bolts* (2006), Z. Szallasi, J. Stelling, and V. Periwal, Eds., Cambridge, Massachussets: The MIT Press, pp. 73–96.

[19] Kolchin, E. R. *Differential Algebra and Algebraic Groups*. Academic Press, New York, 1973.

[20] Lemaire, F., Moreno Maza, M., and Xie, Y. The RegularChains library in MAPLE 10. In *The MAPLE conference* (2005), I. S. Kotsireas, Ed., pp. 355–368.

[21] Michaelis, L., and Menten, M. Die kinetik der invertinwirkung. *Biochemische Zeitschrift 49* (1973), 333–369. Partial translation in english on `http://web.lemoyne.edu/~giunta/menten.html`.

[22] Morant, P.-E., Vandermoere, C., Parent, B., Lemaire, F., Corellou, F., Schwartz, C., Bouget, F.-Y., and Lefranc, M. Oscillateurs génétiques simples. Applications à l'horloge circadienne d'une algue unicellulaire. In *proceedings of the* Rencontre du non linéaire (Paris, 2007). `http://nonlineaire.univ-lille1.fr`.

[23] Niu, W. *Qualitative Analysis of Biological Systems Using Algebraic Methods*. PhD thesis, Université Paris VI, Paris, June 2011.

[24] Okino, M. S., and Mavrovouniotis, M. L. Simplification of Mathematical Models of Chemical Reaction Systems. *Chemical Reviews 98*, 2 (1998), 391–408.

[25] Ritt, J. F. *Differential Algebra*. Dover Publications Inc., New York, 1950.

[26] Sedoglavic, A. Reduction of Algebraic Parametric Systems by Rectification of their Affine Expanded Lie Symmetries. In *Proceedings of Algebraic Biology 2007* (2007), K. H. H. Anai and T. Kutsia, Eds., vol. 4545 of *LNCS*, pp. 277–291.

[27] Van Breusegem, V., and Bastin, G. Reduced order dynamical modelling of reaction systems: a singular perturbation approach. In *Proceedings of the 30th IEEE Conference on Decision and Control* (Brighton, England, December 1991), pp. 1049–1054.

[28] Vilar, J. M. G., Kueh, H. Y., Barkai, N., and Leibler, S. Mechanisms of noise-resistance in genetic oscillators. *Proceedings of the National Academy of Science of the USA 99*, 9 (2002), 5988–5992.

[29] Vora, N., and Daoutidis, P. Nonlinear model reduction of chemical reaction systems. *AIChE Journal 47*, 10 (2001), 2320–2332.

[30] Wang, D. *Elimination Practice: Software Tools and Applications*. Imperial College Press, London, 2003.

[31] Wang, D., and Xia, B. Stability Analysis of Biological Systems with Real Solution Classification. In *proceedings of ISSAC 2005* (Beijing, China, 2005), pp. 354–361.

François Boulier
Université Lille I, LIFL
e-mail: `Francois.Boulier@lifl.fr`

Marc Lefranc
Université Lille I, PhLAM
e-mail: `Marc.Lefranc@univ-lille1.fr`

François Lemaire
Corresponding Author
Université Lille I, LIFL
e-mail: `Francois.Lemaire@lifl.fr`

Pierre-Emmanuel Morant
Université Lille I, PhLAM
e-mail: Pierre-Emmanuel.Morant@phlam.univ-lille1.fr