Approximate Constrained Lumping of Polynomial Differential Equations

Alexander Leguizamon-Robayo[1(✉)], Antonio Jiménez-Pastor[1], Micro Tribastone[2], Max Tschaikowski[1], and Andrea Vandin[3,4]

[1] Aalborg University, Aalborg, Denmark
alexanderlr@cs.aau.dk
[2] IMT School for Advanced Studies Lucca, Lucca, Italy
[3] Sant'Anna School of Advanced Studies, Pisa, Italy
[4] DTU Technical University of Denmark, Lyngby, Denmark

Abstract. In life sciences, deriving insights from dynamic models can be challenging due to the large number of state variables involved. To address this, model reduction techniques can be used to project the system onto a lower-dimensional state space. Constrained lumping can reduce systems of ordinary differential equations with polynomial derivatives up to linear combinations of the original variables while preserving specific output variables of interest. Exact reductions may be too restrictive in practice for biological systems since quantitative information is often uncertain or subject to estimations and measurement errors. This might come at the cost of limiting the actual aggregation power of exact reduction techniques. We propose an extension of exact constrained lumping which relaxes the exactness requirements up to a given tolerance parameter ε. We prove that the accuracy, i.e., the difference between the output variables in the original and reduced model, is in the order of ε. Furthermore, we provide a heuristic algorithm to find the smallest ε for a given maximal approximation error. Finally, we demonstrate the approach in biological models from the literature by providing coarser aggregations than exact lumping while accurately capturing the original system dynamics.

Keywords: Approximate reduction · Dynamical systems · Constrained lumping

1 Introduction

Dynamical models of biochemical systems help discover mechanistic principles in living organisms and predict their behavior under unseen circumstances. Realistic and accurate models, however, often require considerable detail that inevitably leads to large state spaces. This hinders both human intelligibility and numerical/computational analysis. For example, even the relatively primary mechanism of protein phosphorylation yields, in the worst case, a combinatorial state space as a function of the number of phosphorylation sites [37].

As a general way to cope with large state spaces, model reduction aims at providing a lower-dimensional representation of the system under study that

J. Pang and J. Niehren (Eds.): CMSB 2023, LNBI 14137, pp. 106–123, 2023.
https://doi.org/10.1007/978-3-031-42697-1_8

retains some dynamical properties of interest to the modeler. In applications to systems biology, it is beneficial that the reduced state space keeps physical interpretability, mainly when the model is used to validate mechanistic hypotheses [3,38,41]. There is a variety of approaches in this context, such as those exploiting time-scale separation properties [32,51], quasi-steady-state approximation [35,39], heuristic fitness functions [42], spatial regularity [48], sensitivity analysis [40] and, at last, conservation analysis, which detects linear combinations of variables that are constant at all times [49].

By lumping, one generally refers to the latter class, with a self-consistent system of dynamical equations comprised of a set of macro-variables, each given in terms of combination of the original ones [1,6,22,32,40]. In linear lumping, this reduction is expressed as a linear transformation of the original state variables. Since this can destroy physical intelligibility in general, *constrained lumping* allows restricting to only part of the state, allowing defining linear combinations of state variables that ought to be preserved in the reduction [30]. Lumping techniques of Markov chains date back to early nineties [27], were later extended to stochastic process algebra [11,24,44,46,52] and have expanded recently to efficient algorithmic approaches for stochastic chemical reaction networks [12] and deterministic models of biological systems [15,17,19]. In these work, reduction mappings are linear mappings induced by a partition (i.e., an equivalence relation) of state variables; in the aggregated system each macro-variable represents the sum of the original variables of a partition block.

In this paper we are concerned with systems of ordinary differential equations (ODEs) with polynomial derivatives, subsuming mass-action kinetic models (e.g., [45,50]) which is however also rich enough to cover electric circuits [13]. For these, CLUE has been presented as an algorithm that efficiently computes constrained linear lumping [33], avoiding the computational shortcomings of previous work due to the symbolic computation of the eigenvalues of a nonconstant matrix [28,30], enabling the analysis of models with several thousands of equations on standard hardware. In particular, CLUE can compute *the smallest* linear dimensional reduction that preserves the dynamics of arbitrary linear combinations of original state variables given by the user.

The aforementioned lumping approaches are *exact*, in that the reduced model does not incur any approximation error (but only loss of information because, in general, the aggregation map is not invertible). Approximate lumping is a natural extension that has been studied for a long time (e.g., [29]). Indeed, although exact reduction methods have been experimentally proved successful in a large variety of biological systems (e.g., [34]), approximate reductions can be more robust to parametric uncertainty—which notoriously affects systems biology models (e.g., [4,7])—and offer a flexible trade-off between the aggressiveness of the reduction and its precision. For partition-based lumping algorithms, this has been recently explored. In [18], the authors present an algorithm for approximate aggregation parameterized by a tolerance parameter ε, which, informally, relaxes an underlying criterion of equality for two variables to be exactly lumped into the same block.

In this paper, we present for the first time a polynomial time algorithm for the computation of approximate constrained lumpings. It is an extension of CLUE that relaxes its conditions for exact lumping using a *lumping tolerance* parameter that is roughly related to how close a lumping matrix is to an exact lumping. Moreover, we show that the actual error is proportional to the lumping tolerance. Using a prototype implementation, we evaluate our approximate constrained lumping on three representative case studies from the literature. Overall, the numerical results show that our approximate extension can lead to substantially smaller reduced models while introducing limited errors in the dynamics.

Further Related Work. We are complementary to classic works [29,30] which do not address the efficient algorithmic computation of lumping. Moreover, we are more general than [18,26,47] which consider so-called "proper" lumping (i.e., [40]) where each state variable appears in exactly one aggregated variable. Likewise, we are complementary to rule-based reduction techniques [21] which are independent of kinetic parameters [14]. As less closely related abstraction techniques, we mention (bisimulation) distance approaches for the approximate reduction of Markov chains [5,20] and proper orthogonal projection [2], which, however, apply to linear systems. Abstraction of chemical reaction networks by means of learning [10,36] and simulation [23] are complementary to lumping methods.

Outline. Section 2 provides necessary preliminary notations, Sect. 3 introduces approximate constrained lumping, while Sect. 4 discusses how to compute it and how to choose the lumping tolerance value. Section 5 evaluates our proposal on models from the literature, while Sect. 6 concludes the paper.

Notation. The derivative with respect to time of a function $x : [0;T] \to \mathbb{R}^m$ is denoted by $\dot{x}$. We denote a dynamical system by $\dot{x} = f(x)$, and denote its initial condition by $x(0) = x^0$. For $f : \mathbb{R}^m \to \mathbb{R}^n$, we denote the Jacobian of f at x by $J(x)$. For any vector $x \in \mathbb{R}^m$, we denote by $\mathbb{R}[x]$ the rings of polynomial functions of x with real coefficients respectively. Given a matrix $L \in \mathbb{R}^{m\times n}$, the rowspace of L or the vector space generated by the rows of L is denoted by $\text{rowsp}(L)$. We denote by $\bar{L} \in \mathbb{R}^{n\times m}$ a right pseudoinverse of L (i.e. $L\bar{L} = I_n$ where $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix). The term *pseudoinverse* refers to a right pseudoinverse.

2 Preliminaries

In this paper we study systems of ODEs with polynomial derivatives of the form:

$$\dot{x} = f(x), \tag{1}$$

where $f : \mathbb{R}^m \to \mathbb{R}^m, x \mapsto (f_1(x), \ldots, f_m(x))^T$ and $f_i \in \mathbb{R}[x]$, for $i = 1, \ldots, m$.

Definition 1. Given a system of polynomial ODEs, and a full rank matrix $L \in \mathbb{R}^{l\times m}$ with $l < m$, we say that L is an *exact lumping of dimension* l (or that the system is *exactly lumpable* by L) if there exists a function $g : \mathbb{R}^l \to \mathbb{R}^l$ with polynomial entries such that $L \circ f = g \circ L$.

Example 1. Consider the following system

$$\dot{x_1} = x_2^2 + 4x_2x_3 + 4x_3^2, \qquad \dot{x_2} = 2x_1 - 4x_3, \qquad \dot{x_3} = -x_1 - x_2. \tag{2}$$

Then, the matrix $L = (1\ 0\ 0, 0\ 1\ 2)^T$ is an exact lumping of dimension 2. To see this we compute

$$\begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 + 2\dot{x}_3 \end{pmatrix} = \begin{pmatrix} (x_2 + 2x_3)^2 \\ -2x_2 - 4x_3 \end{pmatrix} = \begin{pmatrix} y_2^2 \\ -2y_2 \end{pmatrix}.$$

In this case, we have $g(y) = (y_2^2, -2y_2)^T$. ■

Definition 2. Given a system of polynomial ODEs, and an exact lumping $L \in \mathbb{R}^{l\times m}$, we say that $y = Lx$ are the *reduced (or lumped) variables*, and their evolution is given by the *reduced system* $\dot{y} = g(y)$.

Given an initial condition $x^0 \in \mathbb{R}^n$ and an exact lumping L, there is a corresponding initial condition y^0 in the lumped variables given by $y^0 = Lx^0$. Similarly, since $y = Lx$ and $g \circ L = Lf$, we have that

$$L\dot{x} = Lf(x) = g(Lx) = g(y) = \dot{y}\ .$$

This means that we can study the evolution of the lumped variables $y(t)$ by solving the (smaller) reduced system rather than the (larger) original one.

Suppose that we want now to recover the evolution of some linear combination of state variables. To answer whether this is possible in the context of our example, we introduce the notion of constrained lumping. Roughly speaking, this is an exact lumping able to preserve given observations of interest.

Definition 3. Let $x_{obs} = Mx$ for some matrix $M \in \mathbb{R}^{p\times m}$, for $p < m$. We say that a lumping L is a *constrained lumping* with *observables* x_{obs} if $\mathrm{rowsp}(M) \subseteq \mathrm{rowsp}(L)$. This means that each entry of x_{obs} is a linear combination of the reduced variables y.

Example 2. Consider the system in Eq. (2) and suppose we are interested in observing the quantity $2x_1 + x_2 + 2x_3$. In this case $x_{obs} = Mx$ with $M = (2\ 1\ 2)$. We can see that L from Example 1 is a constrained lumping as we can recover the observable from the reduced system, i.e., $x_{obs} = 2y_1 + y_2$. Suppose now that we want to observe the quantity $x_1 + x_2 + x_3$. This will be described by $x_{obs} = Mx$ with $M = (1\ 1\ 1)$. In this case, the matrix L is not a constrained lumping as there is no way to obtain x_{obs} as a linear combination of y_1 and y_2. ■

To understand how constrained lumping can be computed, we first review the following known characterization of lumping.

Theorem 1 (Characterization of Exact Lumping [43]) *Given a system $\dot{x} = f(x)$ of m polynomial ODEs and a matrix $L \in \mathbb{R}^{l\times m}$ with rank l, the following are equivalent.*

1. *The system is exactly lumpable by* L.
2. *For any pseudoinverse* $\bar{L}$ *of* L, $Lf = (Lf) \circ \bar{L}L$.
3. *The row space of* L *is invariant under* $J(x)$ *for all* $x \in \mathbb{R}^m$, *where* $J(x)$ *is the total derivative of* f *at* x, *known also as the Jacobian. More formally,* $\text{rowsp}(LJ(x)) \subseteq \text{rowsp}(L)$ *for all* $x \in \mathbb{R}^m$.

The characterization of exact lumpings in Theorem 1 provides us with a way to compute constrained lumpings L. This is because thanks to Point 3, the problem of computing a lumping is equivalent to the problem of finding a $J(x)$-invariant subspace of $\mathbb{R}^m$ for all $x \in \mathbb{R}^m$. This gives rise to Algorithm 1.

Algorithm 1. Constrained lumping [33]

Input: A system $\dot{x} = f(x)$ of m polynomial ODEs; A $p \times m$ matrix M with row rank p.

1: **compute** $J(x)$, the Jacobian of $f(x)$
2: **compute** a representation $J(x) = \sum_{i=1}^{N} J_i \mu_i(x)$ (Theorem 2);
3: **compute** L (Algorithm 2)
4: **return** Constrained lumped ODE system $\dot{y} = Lf(\bar{L}y)$.

Algorithm 2. Computation of L

Input: Matrices J_i, $i = 1, \ldots, N$, such that $J(x) = \sum_{i=1}^{N} J_i \mu_i(x)$ (Theorem 2); A $p \times m$ matrix M with row rank p.

1: **set** $L := M$
2: **repeat**
3: **for all** $1 \leq i \leq \kappa$ and rows r of L **do**
4: **compute** rJ_i
5: **if** $rJ_i \notin \text{rowsp}(L)$ **then**
6: **append** row rJ_i to L
7: **end if**
8: **end for**
9: **until** no rows are appended to L
10: **return** Lumping matrix L.

We can see that Algorithm 1 uses a secondary algorithm, Algorithm 2, to actually compute constrained lumpings L. We explain next how this is done efficiently. In particular, the next theorem shows how to verify Point 3 of Theorem 1 by performing a finite number of numerical checks. The number is proportional to the number of monomials present in the polynomial vector field f.

Theorem 2 ([33, Lemma I.1]). *Consider a system of polynomial ODEs as in Eq. (1). Let* $J(x)$ *be the Jacobian matrix of* f. *We have that* $J(x)$ *can be represented as*

$$J(x) = \sum_{i=0}^{N} J_i \mu_i(x), \tag{3}$$

where $\{\mu_i(x) : i \in \{0, \ldots, N\}\}$ *is the set of monomials in* $J(x)$. *Then, for all rows* r *of* L, $rJ(x) \in \text{rowsp}(L)$ *for all* x *if and only if* $rJ_i \in \text{rowsp}(L)$ *for* $i = 1, \ldots, N$.

Example 3. The Jacobian for the system of Example 1 is given by

$$J(x) = \begin{pmatrix} 0 & 2x_2 + 4x_3 & 4x_2 + 8x_3 \\ 2 & 0 & -4 \\ -1 & -1 & 0 \end{pmatrix}.$$

Using Theorem 2, $J(x)$ can be represented as

$$\begin{aligned} J(x) &= J_1\mu_1(x) + J_2\mu_2(x) + J_3\mu_3(x) \\ &= \begin{pmatrix} 0 & 0 & 0 \\ 2 & 0 & -4 \\ -1 & -1 & 0 \end{pmatrix} 1 + \begin{pmatrix} 0\,2\,4 \\ 0\,0\,0 \\ 0\,0\,0 \end{pmatrix} x_2 + \begin{pmatrix} 0\,4\,8 \\ 0\,0\,0 \\ 0\,0\,0 \end{pmatrix} x_3. \end{aligned}$$

We note that, in this example, the μ_i functions are $\mu_1(x) = 1, \mu_2(x) = x_2$ and $\mu_3(x) = x_3$. Thanks to the fact that the vector field is polynomial, monomials μ_i can be computed in polynomial time [33]. ■

Theorem 2 gives a computational way to verify Point 3 in Theorem 1, see Algorithm 1–2 whose polynomial complexity is addressed in [33].

3 Approximate Constrained Lumping

In this section we provide an estimation for the error underlying an approximate lumping. We begin by demonstrating why the exact lumping condition can be too restrictive for finding reductions. Consider the following system of ODEs:

$$\dot{x_1} = x_2^2 + 4.05x_2x_3 + 4x_3^2, \quad \dot{x_2} = 2x_1 - 4x_3, \quad \dot{x_3} = -x_1 - x_2. \tag{4}$$

Notice that the system given by Eq. (4) is similar to that of Eq. (2), as we have just added a term $0.05x_2x_3$ to its first equation. This new system is not exactly lumpable by the matrix L of Example 1. However, we would like to know if it is still possible to use the matrix L to obtain a *useful* reduced system which approximates well the original one. To do so, we first want to evaluate how *close* the matrix L is to be (or how much it *deviates* from) an exact lumping.

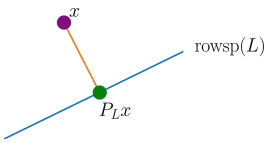


Fig. 1. Points projected to rowsp(L).

By Theorem 1, Point 2, a full rank matrix L is a lumping if and only if the equality $Lf(\bar{L}Lx) - Lf(x) = 0$ is satisfied for all $x \in \mathbb{R}^m$. In our proposal of approximate lumping, we relax this requirement by asking it to be satisfied up to a certain tolerance.

Definition 4. Consider the system of ODEs as in Eq. (1). Let $L \in \mathbb{R}^{l\times m}$ be full rank matrix with $l < m$ and denote by $\bar{L}$ the Moore-Penrose right pseudo inverse of L. We define the projection onto rowsp(L) as $P_L = \bar{L}L$ and the *deviation of* L *at* x by

$$\mathrm{dev}_L(f, x) := \|Lf(\bar{L}Lx) - Lf(x)\|_2. \tag{5}$$

To understand the intuition behind Definition 4, consider the point x in Fig. 1. Then the deviation computes the difference between the images under $L \circ f$ of x (purple) and $P_L x$ (green). The deviation is identically zero if and only if L is a lumping.

Example 4. Consider the matrix L given in Example 1. A pseudoinverse of L is given by $\bar{L} = (1\ 0\ 0, 0\ 0.2\ 0.4)^T$. Writing f^1 for the corresponding vector field, we obtain that $dev_L(f^1, (1,1,1)^T) = 0$, as L is an exact lumping. Now consider the system given by Eq. (4) and use f^2 to denote the corresponding vector field. Then, we get that $\mathrm{dev}_L(f^2, (1,1,1)^T) = 0.014$. Therefore, the matrix L is not an exact lumping of Eq. (4). ■

From a modelling perspective, a reduced model is meaningful insofar as its predictions for a set of initial conditions of interest are *close enough* to those of the original model throughout a given finite time horizon of interest. On the other hand, the theory in [33] guarantees that reduced models provide exact predictions. This might restrict the actual aggregation power of the technique. Here, we aim at relaxing the theory by considering reductions that do not need to be exact or tight. Having this in mind, we introduce the following notion.

Definition 5. Consider the system of ODEs given by Eq. (1) and a set of initial conditions S. Let $L \in \mathbb{R}^{l \times m}$ be a full rank matrix with $l < m$. Given $\eta > 0$ and a time horizon $T > 0$, we say that Eq. (1) is *approximately lumpable* by L with *deviation tolerance* η if

$$\mathrm{dev}_L(f, x(t)) \leq \eta, \tag{6}$$

for all $t \in [0, T]$ and all initial conditions $x(0) \in S$, where $x(t)$ is the solution of the system in Eq. (1). We say that L is an *approximate lumping* for the set S, time horizon T, and deviation tolerance η (or (S, T, η)-lumping). We will omit S, T or η whenever they can be inferred from the context.

Remark 1. The notion of approximate lumping generalizes that of exact lumping from Definition 1. To see this, suppose L is an exact lumping. Then by Point 2 of Theorem 1, $\|Lf(\bar{L}Lx) - Lf(x)\|_2 = \mathrm{dev}_L(f, x) = 0$, for all $x \in \mathbb{R}^m$. It follows that L is a $(\mathbb{R}^m, \infty, 0)$-lumping.

A common type of study is to see the evolution of a system $\dot{x}(t) = f(x(t))$ until it reaches a steady-state for $t \to \infty$. In practice, modelers equip biological models with finite time horizon T, thus implicitly assuming that the system will reach steady-state at T. We shall adhere to this heuristic but point out that a rigorous steady-state analysis would require to ensure asymptotic convergence using, e.g., Lyapunov functions [18].

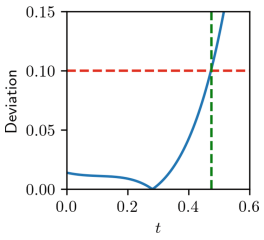
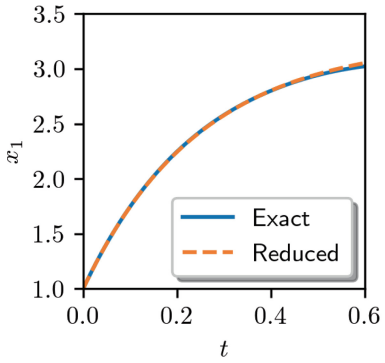
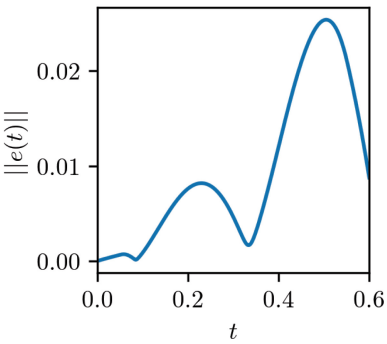


Fig. 2. Example 5: evolution of $\mathrm{dev}_L(f(x(t)))$.

Example 5. Consider the system in Eq. (4), the matrix L of Example 1 and let $x(0) = (1,1,1)^T$. Then L is an approximate



(a) Evolution of the reduced and original systems with the (singleton) set of initial conditions $S = \{(1, 1, 1)^T\}$.



(b) Error evolution $\|e(t)\|_2$

Fig. 3. Reduced system and error computation for Example 6 using the matrix L of Example 1 on Eq. (4).

lumping for $x(0)$, time $T = 0.475$, and deviation tolerance 0.1, i.e., L is a $(\{x(0)\}, 0.475, 0.1)$-lumping. To see this, note that in Fig. 2 the deviation of the dynamics is bounded by 0.1. ■

After having generalized the notion of lumping in that of approximate lumping, we next introduce approximate constrained lumping in the obvious manner.

Definition 6. Let $x_{obs} = Mx$ for some matrix $M \in \mathbb{R}^{p\times m}$ with $p < m$. We say that an approximate lumping L of Eq. (1) is an *approximate constrained lumping* with *observables* x_{obs} if $\text{rowsp}(M) \subseteq \text{rowsp}(L)$.

Similarly to Definition 2, Definition 6 means that the observables x_{obs} can be recovered as a linear combination of the reduced variables $y = Lx$.

Definition 7. The *reduced system* induced by an approximate lumping L is given by $\dot{y} = Lf(\bar{L}y)$, where $y \in \mathbb{R}^l$.

Example 6. Consider the system in Eq. (4) and the matrix L given in Example 1. We can compute the reduced system as follows

$$\begin{pmatrix}\dot{y}_1\\ \dot{y}_2\end{pmatrix} = Lf\left(\bar{L}\begin{pmatrix}y_1\\ y_2\end{pmatrix}\right) = Lf\left(\begin{pmatrix}y_1 & 0\\ 0 & 0.2y_2\\ 0 & 0.4y_2\end{pmatrix}\right) = L\begin{pmatrix}1.004y_2^2\\ 2y_1 - 1.6y_2\\ -y_1 - 0.2y_2\end{pmatrix} = \begin{pmatrix}1.004y_2^2\\ -2y_2\end{pmatrix}.$$

We can see that L induces the reduced system $\dot{y} = (1.004y_2^2, -2.0y_2)^T$. ■

Definition 8. Given the system in Eq. (1) and an approximate lumping L, we define the *error* of the reduction by $e(t) := y(t) - Lx(t)$, where y is the solution to the reduced system given by Definition 7. The dynamics of the approximation error, instead, is given by $\dot{e} = \dot{y} - L\dot{x}$, with $e(0) = 0$.

Example 7. Following Example 6, we compute the trajectories of the reduced and original system. We also compute the L_2-error of the reduction. This is summarized in Fig. 3. Note that, in this example, we obtained a reduction of a system that was not exactly lumpable. In Fig. 3a, we can appreciate that, for the given time horizon, the reduced and original values are close to each other. ■

We next show that it is possible to bound the error introduced by approximate constrained lumping. Despite that our worst case bound is often conservative in practice, the bound confirms the consistency of the approach: the error is of order $\mathcal{O}(\eta)$ — that is, the smaller the deviation η, the smaller the actual error e. As we will see in Sect. 5, despite the conservative nature of the bound, our framework can find approximate lumpings with low errors for published biological models.

Theorem 3 (Error Bounds). *Fix a bounded set of initial conditions S, a finite time horizon T and assume that the respective reachable set of the m-dimensional polynomial ODE system $\dot{x} = f(x)$ is bounded on $[0;T]$. Then, there exists a constant $C \geq 0$ such that for any $\eta > 0$ for which L is a $(S,T,\eta)-$ lumping, it holds that $\|e(t)\|_2 \leq \eta \cdot K_{C,L,T}$, where*

$$K_{C,L,T} = \frac{1}{C\|L\|_2\|\bar{L}\|_2}\left(e^{C\|L\|_2\|\bar{L}\|_2 T} - 1\right)$$

and C is the Lipschitz constant of f over the set of initial conditions S.

4 Computing Constrained Approximated Lumping

Armed with the estimation of the error underlying approximate lumpings, we next focus on their computation. In Sect. 4.1, we introduce a numerical lumping tolerance ε to compute a lumping matrix L, opposed to the *deviation tolerance* η. Moreover, we show how the lumping tolerance ε and the deviation tolerance η relate to each other. In Sect. 4.2 instead we discuss how to pick the best value for the lumping tolerance ε.

4.1 Lumping Algorithm

In this section, we relax the condition in Line 5 of Algorithm 2, thus allowing to find approximate reductions. Intuitively, we add a new row rJ_i to the matrix L only if it is far enough from $\text{rowsp}(L)$. We make this rigorous by fixing a lumping tolerance ε and adding a row rJ_i only if $\|rJ_i - \pi_i\|_2 > \varepsilon$, where $\pi_i := rJ_iP_L$ is the orthogonal projection of rJ_i onto $\text{rowsp}(L)$. Thus, we propose Algorithm 3.

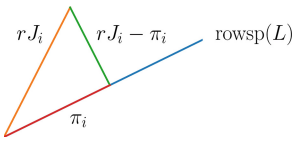


Fig. 4. Decomposition of rJ_i into $\text{rowsp}(L)$ and $\text{rowsp}(L)^{\perp}$.

Algorithm 3. Approximate Constrained Lumping Algorithm

Input: numerical threshold $\varepsilon \geq 0$;
set of matrices $J_i,\ i = 1, \ldots, N$ such that $J(x) = \sum_{i=1}^{N} J_i \mu_i(x)$ (Theorem 2)
a $p \times m$ matrix of observables M with row rank p

1: **compute** orthonormal rows spanning the row space of M and store them as M
2: **set** $L := M$
3: **repeat**
4: **for all** $1 \leq i \leq N$ and rows r of L **do**
5: **compute** $\pi_i := rJ_iL^TL$
6: **if** $\|rJ_i - \pi_i\|_2 > \varepsilon$ **then**
7: **append** row $(rJ_i - \pi_i)/\|rJ_i - \pi_i\|_2$ to L
8: **end if**
9: **end for**
10: **until** no rows have been appended to L
11: **return** lumping matrix L.

Remark 2. In Line 7 of Algorithm 3 we do not append rJ_i, given by the orange vector in Fig. 4. Rather, we append the normalized component of rJ_i in the orthogonal direction to rowsp(L), which is $rJ_i - \pi_i$, the green vector in Fig. 4. This ensures that the matrix L is orthonormal, thus we can use the fact that the pseudoinverse of an orthonormal matrix is its transpose to obtain that $\bar{L} = L^T$.

We now provide a detailed example of the use of Algorithm 3.

Example 8. We apply Algorithm 3 to Eq. (4) for $\varepsilon = 0.1$ and $M = (1, 0, 0)$. In other words, we are observing the component x_1 from the original system. The Jacobian of $f(x)$ can be represented as $J_1(1) + J_2(x_2) + J_3(x_3)$, where

$$J_1 = \begin{pmatrix} 0 & 0 & 0 \\ 2 & 0 & -4 \\ -1 & -1 & 0 \end{pmatrix},\ J_2 = \begin{pmatrix} 0 & 2 & 4.05 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix},\ J_3 = \begin{pmatrix} 0 & 4.05 & 8 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{pmatrix}.$$

We begin the computation in Line 2 by setting $L = M = (1, 0, 0)$ and begin the main loop in Line 4 with $r = (1, 0, 0)$. To carry out the computation of Line 5, we have that $rJ_1 = (0, 0, 0),\ rJ_2 = (0, 2, 4.05),\ rJ_3 = (0, 4.05, 8)$. Since $rJ_1 = \pi_1 = (0, 0, 0)$, by the check of Line 6, we do not append any new row to L. As next, we compute π_2 following Line 5. As $\pi_2 = 0$, it follows that $\|rJ_2 - \pi_2\|_2 = \|rJ_2\|_2 = 4.52 > 0.1$, which, by Line 6, means that we need to append a new row to L. By Line 7, we add normalized $rJ_2 - \pi_2$ as a row of L (Line 7), thus obtaining

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.443 & 0.897 \end{pmatrix}. \tag{7}$$

Going back to Line 5, we compute $\pi_3 = rJ_3L^TL = (0, 3.970, 8.040)$. Following Line 6, since $\|rJ_3 - \pi_3\|_2 = 0.09 < 0.1$, we do not add any additional row to L. As we have only checked one of the rows of L so far, we follow the main loop (Line 4) by setting $r = (0, 0.443, 0.897)$. We compute

$$rJ_1 = (-0.011, -0.900, -1.771),\ rJ_2 = (0, 0, 0),\ rJ_3 = (0, 0, 0).$$

We skip to the check of Line 6, where we have that $\|rJ_1 - \pi_1\|_2 = 0.02 < 0.1$. Similarly, we have that $\|rJ_2 - \pi_2\|_2 = \|rJ_3 - \pi_3\|_2 = 0$, meaning that no more rows should be added to L, terminating the algorithm. The output of Algorithm 3 is then the matrix L of Eq. (7). ■

Being a generalization of Algorithm 2, the polynomial complexity of Algorithm 3 follows from [33] and the fact that linear projections can be computed in polynomial time. Likewise, its result is unique and minimal, provided that the input matrix M has full row-rank.

The next result relies on Theorem 2 and ensures that the approximate reductions found by Algorithm 3 admit a deviation tolerance of order $\mathcal{O}(\varepsilon)$. That is, the deviation tolerance η is in the order of the lumping tolerance ε.

Theorem 4. (Correctness of Algorithm 3). *Consider a system of ODEs of the form (1) of dimension m and a set of initial conditions S, such that none of the solutions explode in a finite time $t \leq T$, where T is a given time horizon. Let L be the matrix computed via Algorithm 3 with lumping tolerance ε and let Ω be an open ball centered in the origin of diameter K such that for all $x(0) \in S$, the solution $x(t)$ of (1) remains in Ω for all $t \in [0, T]$. Then, L is a $(T, S, \sqrt{m}C'K\varepsilon)$-lumping, where $C' = \sup_{i,x\in\Omega} |\mu_i(x)|$.*

4.2 Heuristic Search of Lumping Tolerance

In order to apply Algorithm 3, it is important to choose an appropriate value for ε. While Theorem 3 together with Theorem 4 allows for an estimation of the error in terms of ε, in practice, this bound is not tight enough. For this reason, we introduce a heuristic approach to appropriately choose ε.

Intuitively, by increasing ε, one can lump more variables together at the price of incurring larger approximation errors. We would like to find the largest admissible ε such that the approximation error is small enough. Note that the minimum value that ε can have is 0, corresponding to an exact reduction. A naive idea would be to set $\varepsilon = 0$ and add small increments until the reduction is satisfying. However, this requires an appropriate choice of increment which in turn depends on the model. Instead, Lemma 1 gives us an upper bound for ε which can be computed for each model.

Lemma 1. (Upper bound on ε). *Consider a matrix of observables $M \in \mathbb{R}^{p\times m}$ of rank p with $j-$th row denoted by r_j and a decomposition of the Jacobian $J(x) = \sum_{i=1}^{n} J_i\mu_i(x)$. Then there is value $\varepsilon_{\max}$ given by*

$$\varepsilon_{\max} = \max_{j,i} \|r_j J_i\|_2, \tag{8}$$

for $i = 1, \ldots, n$ and each $j = 1, \ldots, l$, such that for any $\varepsilon \geq \varepsilon_{\max}$ the output of Algorithm 3 will be a matrix of orthonormal rows spanning the row space of M.

In other words, any $\varepsilon \geq \varepsilon_{\max}$ will collapse the dynamics of the system onto M. Note also that the error e and the deviation dev_L of the reduction underlying

an approximate lumping L increase as the lumping tolerance ε increases. To find the largest admissible ε, we need a way to approximate the error without having to simulate the original and reduced systems. To this aim, we estimate the deviation of lumping matrix L by sampling and averaging the deviation for N randomly chosen points over $[0; \|x^0\|_2]^m$, where x^0 is the center of the initial set. The computation corresponds to the Monte Carlo approximation of the L_1 norm of dev_L over $[0; \|x^0\|_2]^m$. The choice of the compactum is motivated by the fact that the diameter of the reachable set will be of the order of the initial set.

Algorithm 4. Finding the largest acceptable ε for Algorithm 3

Input: set of matrices $J_i,\ i = 1, \ldots, N$ such that $J(x) = \sum_{i=1}^{N} J_i \mu_i(x)$ (Theorem 2)
maximum deviation $\eta_{\max}$
minimal difference $d_{\min}$
set of initial conditions S and its center x^0
number of random samples $M \geq 1$

set $\varepsilon_{\min} = 0$
compute $\varepsilon_{\max} = \max_{j,i} \|r_j J_i\|_2$
repeat
  compute $\varepsilon = (\varepsilon_{\max} + \varepsilon_{\min})/2$
  compute L using Algorithm 3 with ε
  compute $\text{dev}_L = \frac{1}{M} \sum_{i=1}^{M} \|Lf(L^T L x^i) - Lf(x^i)\|_2$ for random $x^i \in [0; \|x^0\|_2]^m$
  if $\text{dev}_L > \eta_{\max}$ **then**
    set $\varepsilon_{\max} = \varepsilon$
  else
    set $\varepsilon_{\min} = \varepsilon$
  end if
  compute $d = \varepsilon_{\max} - \varepsilon_{\min}$
until $d < d_{\min}$
return $\varepsilon_{\min}$.

To find the largest admissible ε that leads to a reduction close to a given maximal deviation tolerance $\eta_{\max}$, it suffices to search over $\varepsilon \in [0, \varepsilon_{\max}]$ thanks to Lemma 1. We find such an ε by performing a binary search over $[0, \varepsilon_{\max}]$, relying on the intuition that $\text{dev}_{L(\varepsilon_1)} \leq \text{dev}_{L(\varepsilon_2)}$ where $\varepsilon_1 \leq \varepsilon_2$ and $L(\varepsilon_i)$ is the result of Algorithm 3 for input ε_i. The corresponding realization is provided in Algorithm 4, where the maximal acceptable deviation tolerance $\eta_{\max}$ is chosen as a percentage of $\|f(x^0)\|_2$, with x^0 being again the center of the initial set. This is motivated by the fact that dev_L is stated in terms of f, see Definition 4.

5 Evaluation

In this section, we evaluate our method by demonstrating its higher reduction power over exact constrained lumping. We use the following selection of models from the literature, with observables taken from the cited papers (For models with more than one observable we arbitrarily selected one.)

- **BioNetGen_CCP** [31]: the central carbon pathway of E. coli is a metabolic process that converts glucose into energy and building blocks for cell growth.
  Observable: concentration of 1,3-diphosphoglycerate (D13PG). This corresponds to the sum of 2 out of the 87 variables in the model.
- **NIHMS80246_S6** [9]: FceRI-like network of a cell-surface receptor of the receptor tyrosine kinase (RTK) family.
  Observable: total amount of phosphorylation at Y2 (S2P). This corresponds to the sum of 10 out of the 24 variables in the model.
- **pcbi_1003544_s006** [25]: trivalent-ligand bivalent-receptor (TLBR) model, which is a simplified representation of receptor aggregation following multivalent ligand binding.
  Observable: total concentration of membrane receptor (RC). This corresponds to the sum of 18 out of the 62 variables in the model.

These are chemical reaction networks written in the input language of the tool BioNetGen [8] as available in the corresponding cited papers. We use MATLAB to implement all algorithms presented in this paper, as well as to carry out the simulations of the reduced and original models. The Jacobians for all models were automatically computed using ERODE [16], using the importing capabilities of ERODE for the `.net` format generated using BioNetGen version 2.2.5-stable. The time horizon T and initial conditions for all models were taken from their original papers. For reproducibility, we provide all the generated MATLAB files, including the specifications of the models, at https://www.erode.eu/models/CMSB2023.zip.

Overall, these models showcase three exemplary situations: (1) the suitable ε computed by Algorithm 4 gives good reductions with low errors, while an excessively large ε completely destroys the observed dynamics; (2) our approach can only provide limited reduction improvements; (3) a model that cannot be reduced by exact constrained lumping, whereas approximate constrainted lumping is able to halve the number of variables while adding limited errors.

Table 1 presents the detailed results of these three experiments. Using Algorithm 4, we seek to find the largest lumping tolerance ε that does not exceed a given maximal deviation tolerance $\eta_{\max}$, where the latter is described as a percentage of the initial slope via $\eta_{\max} = \texttt{SlopePercentage}\|f(x_0)\|_2$. Similarly to the choice of the compact set for the evaluation of dev_L, the choice of $\eta_{\max}$ is justified by the fact that the slopes at points contained in the reachable set are roughly of the same order to the slope at the center of the initial set $f(x_0)$. For difference percentages, we then find the largest admissible lumping tolerance ε using Algorithm 4. Table 1 summarizes the results of the experiments. The column *Red. size* shows the size of the reduced model obtained by Algorithm 3. We display the absolute error and the relative error at the time horizon, respectively, in the columns $e(T)$ and $e(T)_{Rel}$ (where the relative error is given as the absolute error divided by the value of the observable of the exact reduction). We also display the number of iterations and the average computation time of Algorithm 4 over 5 runs in the *Time* column. All experiments were carried out on a 4.7 GHz Intel Core i7 computer with 32 GB of RAM. For each model, we

plot in Fig. 5 the corresponding simulations of the observables for the considered values of ε.

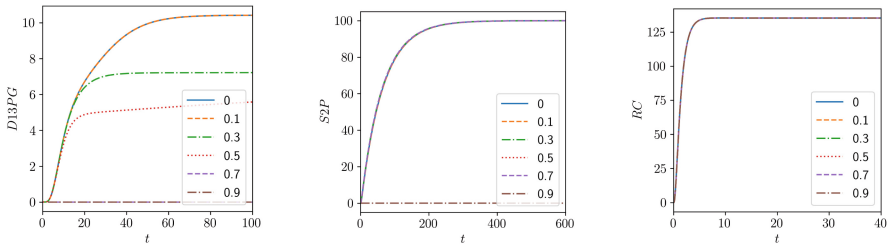
Table 1. Validation: greater reduction power on original models.

Red. size	$e(T)$	$e_{Rel}(T)[\%]$	ε	*SlopePercentage*	*Iterations*	*Time[s]*
Experiment 1) Model: BioNetGen_CCP, size: 87, exact lumping: 30						
30	6.210E–06	5.960E–05	0.13313	10%	22	2.098
23	3.199E+00	3.070E+01	0.34426	30%	22	1.459
9	4.835E+00	4.641E+01	0.71208	50%	22	0.719
5	1.042E+01	1.000E+02	0.71720	70%	22	0.593
4	1.042E+01	1.000E+02	1.32714	90%	22	0.395
Experiment 2) Model: NIHMS80246_S6, size: 24, exact lumping: 19						
19	2.706E–08	2.706E–08	0.01696	10%	19	0.607
18	4.793E–04	4.793E–04	0.03000	30%-70%	19	0.895
8	9.999E+01	1.000E+02	0.21033	90%	19	0.441
Experiment 3) Model: pcbi_1003544_s006, size: 62, exact lumping: 62						
28	3.200E–09	2.366E–09	0.00079	10%-90%	22	2.762

In Experiment 1, we can see that, while for `SlopePercentage` 10% we get no reduction improvements, for `SlopePercentage` 30% and 50% get substantial reduction improvements going down to 23 and 9 variables, respectively, while preserving the shape of the dynamics of the observable (middle lines in Fig. 5a). Furthermore, it is interesting to note that aggressive reductions come at the price of larger errors. In fact, by setting `SlopePercentage` to 70% we lump the system so much that the observable in question degenerates to a constant (bottom flat line in Fig. 5a).

In Experiment 2 we can see that by setting `SlopePercentage` to 10% we do not get reduction improvements, while by allowing it to go up to 70% we only get very limited improvements (we reduce to 18 variables rather than to 19), without adding noticeable errors (this can be seen in Fig. 5b, were the exact and approximate reductions are indistinguishable). More aggressive reductions, e.g., obtaining 8 variables for `SlopePercentage` 90%, destroy the dynamics (the bottom flat line in Fig. 5b).

Finally, Experiment 3 shows a model that is not exactly lumpable. That is, exact lumping does not allow to lump its 62 variables. Instead, our approximate variant allows to reduce the model to less than 50% of the original variables (45%). This happens for any `SlopePercentage` from 10% to 90%. Most importantly, this reduction comes at very limited cost: as we can see in Fig. 5c, the lines can be hardly distinguished.



(a) Experiment 1 (b) Experiment 2 (c) Experiment 3

Fig. 5. Simulation of exact and approximately lumped models from Table 1, for varying `SlopePercentage`. The 0-line refers to exact constrained lumping.

6 Conclusion

In this work we introduced approximate constrained lumping, a framework for the reduction of systems of polynomial differential equations. While approximate lumping has been studied before, to the best of our knowledge no polynomial time algorithm for its computation has been provided before. The proposed algorithm takes as input a numerical tolerance and computes an approximate constrained lumping whose error is guaranteed to be proportional to the numerical threshold (with the proportional constant being not dependent on the threshold). The applicability of the framework is demonstrated by finding approximate lumpings of published models with low errors. Future work will consider the extension to more general vector fields, where the challenge is to express the derivative of the vector field as a sum of computationally amenable functions. In the case of polynomial vector fields considered here, for instance, these are monomials.

Acknowledgment. The work was partially supported by the DFF project REDUCTO 9040-00224B, the Poul Due Jensen Grant 883901, the Villum Investigator Grant S4OS, the PRIN project SEDUCE 2017TWRCNB and the co-funding of European Union - Next Generation EU, in the context of The National Recovery and Resilience Plan, Investment 1.5 Ecosystems of Innovation, Project Tuscany Health Ecosystem (THE), CUP: B83C22003920001.

References

1. Abate, A., Andriushchenko, R., Ceska, M., Kwiatkowska, M.: Adaptive formal approximations of Markov chains. Perform. Evaluation **148** (2021)
2. Antoulas, A.: Approximation of Large-Scale Dynamical Systems. Advances in Design and Control. SIAM (2005)
3. Apri, M., de Gee, M., Molenaar, J.: Complexity reduction preserving dynamical behavior of biochemical networks. J. Theor. Biol. **304**, 16–26 (2012)
4. Babtie, A., Stumpf, M.: How to deal with parameters for whole-cell modelling. J. Roy. Soc. Interface **14**(133), 20170237 (2017)

5. Bacci, G., Bacci, G., Larsen, K.G., Mardare, R.: On-the-fly exact computation of bisimilarity distances. In: N. Piterman and S. A. Smolka, editors, TACAS, vol. 7795. LNCS, pp. 1–15 (2013)
6. Backenköhler, M., Bortolussi, L., Großmann, G., Wolf, V.: Abstraction-guided truncations for stationary distributions of Markov population models. In: QEST, pp. 351–371 (2021)
7. Barnat, J., Beneš, N., Brim, L., Demko, M., Hajnal, M., Pastva, S., Šafránek, D.: Detecting attractors in biological models with uncertain parameters. In: Feret, J., Koeppl, H. (eds.) CMSB 2017. LNCS, vol. 10545, pp. 40–56. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-67471-1_3
8. Blinov, M.L., Faeder, J.R., Goldstein, B., Hlavacek, W.S.: BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. Bioinformatics **20**(17), 3289–3291 (2004)
9. Borisov, N.M., Chistopolsky, A.S., Faeder, J.R., Kholodenko, B.N.: Domain-oriented reduction of rule-based network models. IET Syst. Biol. **2**(5), 342–351 (2008)
10. Cairoli, F., Carbone, G., Bortolussi, L.: Abstraction of Markov population dynamics via generative adversarial nets. In: Cinquemani, E., Paulevé, L. (eds.) CMSB 2021. LNCS, vol. 12881, pp. 19–35. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-85633-5_2
11. Cardelli, L.: From processes to odes by chemistry. In: Ausiello, G., Karhumäki, J., Mauri, G., Ong, L. (eds.) Fifth Ifip International Conference on Theoretical Computer Science - Tcs 2008 (2008)
12. Cardelli, L., Pérez-Verona, I.C., Tribastone, M., Tschaikowski, M., Vandin, A., Waizmann, T.: Exact maximal reduction of stochastic reaction networks by species lumping. Bioinform. **37**(15), 2175–2182 (2021)
13. Cardelli, L., Tribastone, M., Tschaikowski, M.: From electric circuits to chemical networks. Nat. Comput. **19**(1), 237–248 (2020)
14. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Forward and backward bisimulations for chemical reaction networks. In: CONCUR, pp. 226–239 (2015)
15. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Comparing chemical reaction networks: a categorical and algorithmic perspective. In: Grohe, M., Koskinen, E., Shankar, N. (eds.) Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2016, July 5–8, 2016, pp. 485–494. ACM, New York (2016)
16. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: ERODE: a tool for the evaluation and reduction of ordinary differential equations. In: Legay, A., Margaria, T. (eds.) TACAS 2017. LNCS, vol. 10206, pp. 310–328. Springer, Heidelberg (2017). https://doi.org/10.1007/978-3-662-54580-5_19
17. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Maximal aggregation of polynomial dynamical systems. PNAS **114**(38), 10029–10034 (2017)
18. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Guaranteed error bounds on approximate model abstractions through reachability analysis. In: McIver, A., Horvath, A. (eds.) QEST 2018. LNCS, vol. 11024, pp. 104–121. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99154-2_7
19. Cardelli, L., Tribastone, M., Tschaikowski, M., Vandin, A.: Symbolic computation of differential equivalences. Theoret. Comput. Sci. **777**, 132–154 (2019)
20. Daca, P., Henzinger, T.A., Kretínský, J., Petrov, T.: Linear distances between Markov chains. In: Desharnais, J., Jagadeesan, R. (eds.) CONCUR, vol. 59. LIPIcs, pp. 20:1–20:15 (2016)

21. Feret, J., Danos, V., Krivine, J., Harmer, R., Fontana, W.: Internal coarse-graining of molecular systems. PNAS **106**(16), 6453–6458 (2009)
22. Großmann, G., Kyriakopoulos, C., Bortolussi, L., Wolf, V.: Lumping the approximate master equation for multistate processes on complex networks. In: McIver, A., Horvath, A. (eds.) QEST 2018. LNCS, vol. 11024, pp. 157–172. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99154-2_10
23. Helfrich, M., Ceska, M., Kretínský, J., Marticek, S.: Abstraction-based segmental simulation of chemical reaction networks. In: Petre, I., Paun, A. (eds.) CMSB, vol. 13447, pp. 41–60 (2022)
24. Hillston, J., Tribastone, M., Gilmore, S.: Stochastic process algebras: from individuals to populations. Comput. J. **55**(7), 866–881 (2011)
25. Hogg, J.S., Harris, L.A., Stover, L.J., Nair, N.S., Faeder, J.R.: Exact hybrid particle/population simulation of rule-based models of biochemical systems. PLOS Comput. Biol. **10**(4), e1003544, April 2014. Publisher: Public Library of Science
26. Iacobelli, G., Tribastone, M.: Lumpability of fluid models with heterogeneous agent types. In: 2013 43rd Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), pp. 1–11, June 2013. ISSN: 2158–3927
27. Larsen, K.G., Skou, A.: Bisimulation through probabilistic testing. Inf. Comput. **94**(1), 1–28 (1991)
28. Li, G., Rabitz, H.: A general analysis of exact lumping in chemical kinetics. Chem. Eng. Sci. **44**(6), 1413–1430 (1989)
29. Li, G., Rabitz, H.: A general analysis of approximate lumping in chemical kinetics. Chem. Eng. Sci. **45**(4), 977–1002 (1990)
30. Li, G., Rabitz, H.: New approaches to determination of constrained lumping schemes for a reaction system in the whole composition space. Chem. Eng. Sci. **46**(1), 95–111 (1991)
31. Mu, F., Williams, R.F., Unkefer, C.J., Unkefer, P.J., Faeder, J.R., Hlavacek, W.S.: Carbon-fate maps for metabolic reactions. Bioinformatics (Oxford, England) **23**(23), 3193–3199 (2007)
32. Okino, M., Mavrovouniotis, M.: Simplification of mathematical models of chemical reaction systems. Chem. Rev. **2**(98), 391–408 (1998)
33. Ovchinnikov, A., Pérez Verona, I., Pogudin, G., Tribastone, M.: CLUE: exact maximal reduction of kinetic models by constrained lumping of differential equations. Bioinformatics **37**(12), 1732–1738, June 2021
34. Pérez-Verona, I.C., Tribastone, M., Vandin, A.: A large-scale assessment of exact model reduction in the BioModels repository. In: Bortolussi, L., Sanguinetti, G. (eds.) CMSB 2019. LNCS, vol. 11773, pp. 248–265. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-31304-3_13
35. Radulescu, O., Gorban, A.N., Zinovyev, A., Noel, V.: Reduction of dynamical biochemical reactions networks in computational biology. Front. Genet. **3**, 131 (2012)
36. Repin, D., Petrov, T.: Automated deep abstractions for stochastic chemical reaction networks. Inf. Comput. **281**, 104788 (2021)
37. Salazar, C., Höfer, T.: Multisite protein phosphorylation – from molecular mechanisms to kinetic models. FEBS J. **276**(12), 3177–3198 (2009)
38. Schmidt, H., Madsen, M., Danø, S., Cedersund, G.: Complexity reduction of biochemical rate expressions. Bioinformatics **24**(6), 848–854 (2008)
39. Segel, L., Slemrod, M.: The quasi-steady-state assumption: a case study in perturbation. SIAM Rev. **31**(3), 446–477 (1989)

40. Snowden, T., van der Graaf, P., Tindall, M.: Methods of model reduction for large-scale biological systems: a survey of current methods and trends. Bull. Math. Biol. **79**(7), 1449–1486 (2017)
41. Sunnaker, M., Cedersund, G., Jirstrand, M.: A method for zooming of nonlinear models of biochemical systems. BMC Syst. Biol. **5**(1), 140 (2011)
42. Tognazzi, S., Tribastone, M., Tschaikowski, M., Vandin, A.: Egac: a genetic algorithm to compare chemical reaction networks. In: GECCO, GECCO 2017, p. 833–840 (2017)
43. Tomlin, A.S., Li, G., Rabitz, H., Tóth, J.: The effect of lumping and expanding on kinetic differential equations. SIAM J. Appl. Math. **57**(6), 1531–1556 (1997). Publisher: Society for Industrial and Applied Mathematics
44. Tribastone, M.: Behavioral relations in a process algebra for variants. In: Gnesi, S., Fantechi, A., Heymans, P., Rubin, J., Czarnecki, K., Dhungana, D. (eds.) SPLC, pp. 82–91. ACM (2014)
45. Troják, M., Safránek, D., Pastva, S., Brim, L.: Rule-based modelling of biological systems using regulated rewriting. Biosyst. **225**, 104843 (2023)
46. Tschaikowski, M., Tribastone, M.: Exact fluid lumpability in Markovian process algebra. Theoret. Comput. Sci. **538**, 140–166 (2014)
47. Tschaikowski, M., Tribastone, M.: Approximate reduction of heterogeneous nonlinear models with differential hulls. IEEE TAC (2016)
48. Tschaikowski, M., Tribastone, M.: Spatial fluid limits for stochastic mobile networks. Perform. Evaluation **109**, 52–76 (2017)
49. Vallabhajosyula, R., Chickarmane, V., Sauro, H.: Conservation analysis of large biochemical networks. Bioinformatics **22**(3), 346–353 (2005)
50. Voit, E.O.: Biochemical systems theory: a review. ISRN Biomathematics **2013**, 53 (2013)
51. Whitby, M., Cardelli, L., Kwiatkowska, M., Laurenti, L., Tribastone, M., Tschaikowski, M.: PID control of biochemical reaction networks. IEEE Trans. Autom. Control **67**(2), 1023–1030 (2022)
52. Wirsing, M., et al.: Sensoria patterns: augmenting service engineering with formal analysis, transformation and dynamicity. In: Margaria, T., Steffen, B. (eds.) Leveraging Applications of Formal Methods, Verification and Validation, pp. 170–190 (2008)