Approximate constrained lumping of chemical reaction networks

PROCEEDINGS A

royalsocietypublishing.org/journal/rspa

Research

Check for updates

Article submitted to journal

Subject Areas:
Computational biology, Differential equations

Keywords:
Approximate reduction, Dynamical systems, Biological models, Constrained lumping

Author for correspondence:
Max Tschaikowski
e-mail: tschaikowski@cs.aau.dk

Approximate Constrained Lumping of Chemical Reaction Networks

Alexander Leguizamon-Robayo[1] Antonio Jiménez-Pastor[2] Micro Tribastone[3] Max Tschaikowski [1] Andrea Vandin[4,5]

[1]Aalborg University, Denmark
[2]Universidad Politécnica de Madrid, Spain
[3]IMT School for Advanced Studies Lucca, Italy
[4]Sant'Anna School of Advanced Studies, Pisa, Italy
[5]DTU Technical University of Denmark, Denmark.

Gaining insights from realistic dynamical models of biochemical systems can be challenging given their large number of state variables. Model reduction techniques can mitigate this by decreasing complexity by mapping the model onto a lower-dimensional state space. Exact constrained lumping identifies reductions as linear combinations of the original state variables in systems of nonlinear ordinary differential equations, preserving specific user-defined output variables without error. However, exact reductions can be too stringent in practice, as model parameters are often uncertain or imprecise—a particularly relevant problem for biochemical systems.

We propose approximate constrained lumping. It allows for a relaxation of exactness within a given tolerance parameter ε, while still working in polynomial time. 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 maximum allowable size of the lumped system. Our method is applied to several models from the literature, resulting in coarser aggregations than exact lumping while still capturing the dynamics of the original system accurately.

THE ROYAL SOCIETY PUBLISHING

1. Introduction

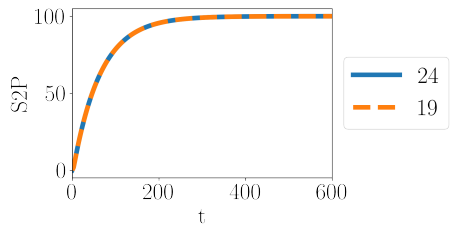
Biochemical models based on dynamical systems help discover mechanistic principles in living organisms and predict their behavior under unforeseen circumstances. Realistic and accurate models, however, often require considerable detail that inevitably leads to large state spaces. This hinders both human intelligibility and numerical/computational analysis. For example, even the relatively primary mechanism of protein phosphorylation may yield a state space that grows combinatorial with the number of phosphorylation sites [1].

As a general way to cope with large state spaces, model reduction aims to provide a lower-dimensional representation of the system under study that retains some dynamic properties of interest to the modeler. In applications to computational systems biology, it is beneficial that the reduced state space keeps physical interpretability, mainly when the model is used to validate mechanistic hypotheses [2–4]. There is a variety of approaches in this context, such as those exploiting time-scale separation properties [5,6], quasi-steady-state approximation [7,8], heuristic fitness functions [9], spatial regularity [10], sensitivity analysis [11], and conservation analysis, which detects linear combinations of variables that are constant at all times [12].

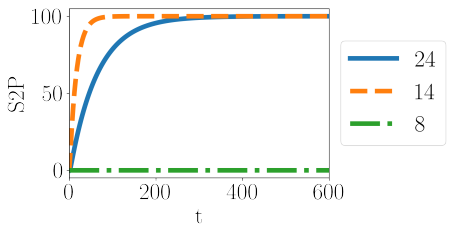
By *lumping* one generally refers to a reduction method that provides a self-consistent system of dynamical equations comprised of a set of macro-variables, each given as a combination of the original ones [5,11,13,14]. In linear lumping, this reduction is expressed as a linear transformation of the original state variables. Since this can destroy physical intelligibility in general, *constrained lumping* allows restricting to only part of the state space by defining linear combinations of state variables that ought to be preserved in the reduction [15]. Lumping techniques of Markov chains date back to the early nineties [16]. These were later extended to stochastic process algebra [17–21], and have been expanded more recently to efficient algorithmic approaches for stochastic chemical reaction networks [22] and deterministic models of biological systems based on ordinary differential equations (ODEs) with polynomial right-hand sides [23–25].

In these cases, the reduction is a specific kind of linear mapping 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. For polynomial ODE systems, CLUE has been presented as an algorithm that computes constrained linear lumping efficiently, i.e., in polynomial time [26], by avoiding the symbolic computation of the eigenvalues of a non-constant matrix that would have required in prior approaches [15,27].

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. A complete step-by-step example of CLUE reduction on a synthetic 3-variables model is provided later (Exampe 2.1). To give a concrete example of the usefulness of CLUE applied to a real model, we consider a model of FceRI-like network of a cell-surface receptor [28]. The *observable* or quantity of interest is the amount of *Phosphorylation at site Y2 (S2P)*, which is the sum of 10 out of the 24 variables in the model. Figure 1a, displays the original simulation in blue and the orange line, instead, shows the solution of such observable on a model containing 19 variables obtained by reducing the former by CLUE. This plot shows that one can study the evolution of *S2P* using either the original model or one reduced by constrained lumping with only 19 variables without introducing any error.



(a) Exactly reduced model by CLUE with 19 variables (orange).



(b) Approximately reduced model with 14 (orange) and 8 (green) variables.

Figure 1: Evolution of observable S2P in a FceRI-like network of a cell-surface receptor [28] using different models. The simulation using the original model with 24 variables is displayed in blue.

All the aforementioned lumping approaches are *exact*, i.e., 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., [30]), approximate reductions can be more robust to parametric uncertainty—which notoriously affects systems biology models (e.g., [31,32])—and offer a flexible trade-off between the aggressiveness of the reduction and its precision. This has been explored for partition-based lumping algorithms: In [33,34], the authors present an algorithm for approximate aggregation parameterized by a tolerance ε, which, informally, relaxes an underlying criterion of equality for two variables to be exactly lumped into the same partition block. These approaches present notions of approximate reduction extending the above-mentioned exact reduction techniques for chemical reaction networks and ODEs. Similarly, in this paper, we present an extension of CLUE [26] that performs approximate constrained linear lumping of ODE systems. It relaxes the conditions for (exact) CLUE by introducing a *lumping tolerance* parameter that roughly relates to how close a lumping matrix is to an exact lumping. We consider ODE systems with analytical derivatives. These include *rational* and *polynomial* derivatives thus covering kinetic models for biochemical systems with Hill [35], Michaelis-Menten [35] and mass-action kinetics (e.g., [35,36]).

To obtain an approximate reduction for a given analytical ODE system, we start by finding a finite representation of the dynamics, either symbolically or using automatic differentiation [37]. Given a user-defined lumping tolerance, we can compute a reduced system that approximates the original one up to an error proportional to the lumping tolerance. For example, the orange line in Figure 1b shows that we can study the evolution of $S2P$ using an approximate reduction with 14 variables that yield the same steady-state value as in the original model. However, a too-permissive lumping tolerance may destroy the original dynamics, as shown by the approximate reduction with 8 variables (bottom green line).

Our proposed approach works in polynomial time. To find the lumping tolerance, we propose a heuristic approach based on the expected size of the reduced model. Using a prototype implementation, we evaluate the aggregation power of our approach on five polynomial models and three rational models representative of the literature. We also evaluate the scalability of our approach on a multisite phosphorylation model [38]. Overall, the numerical results show that our method can lead to substantially smaller reduced models while introducing limited errors in the dynamics.

This paper extends the conference paper [39] by expanding the theory to include analytical drifts; including models with rational drifts; adding a scalability analysis; and proposing a new approach to find the lumping tolerance. A short tool demonstration paper discussing implementation details of our approach has been recently presented in [40].

Further related work. We are complementary to classic works [15,29] which do not address the efficient algorithmic computation of lumping. Moreover, we are more general than [33,41,42] which consider so-called "proper" lumping (i.e., [11]) where each state variable appears in exactly one aggregated variable. As less closely related abstraction techniques, we mention (bisimulation) distance approaches for the approximate reduction of Markov chains [43,44] and proper orthogonal projection [45], which, however, apply to linear systems. Abstraction of chemical reaction networks through learning [46,47] and simulation [48] are complementary to lumping. At last, we are complementary to rule-based reduction techniques [49] which are independent of kinetic parameters [50].

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

Notation. For a function f, we denote its domain by $\mathcal{D}(f)$. 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 $J(x)$. 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 $\mathrm{rowsp}(L)$. We denote by $\bar{L} \in \mathbb{R}^{n\times m}$ a pseudoinverse of L (i.e. $L\bar{L} = I_n$ where $I_n \in \mathbb{R}^{n\times n}$ is the identity matrix). We reserve the term *rational* for models with a non-trivial denominator on their right-hand side.

2. Preliminaries

In this paper, we study systems of ODEs of the form:

$$\dot{x} = f(x), \tag{2.1}$$

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

We first recall the notion of (exact) lumping [51]. Informally, it is defined as a mapping L that leads to a self-consistent system of ODEs in the reduced state space.

Definition 2.1. Given a system of ODEs of the form given by Equation (2.1), 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 $Lf = g \circ L$.

Example 2.1. Consider the following system

$$\dot{x}_1 = \frac{x_2^2 + 4x_2x_3 + 4x_3^2}{x_1^2 + 1}, \qquad \dot{x}_2 = \frac{2x_1 - 4x_3}{x_2 + 2x_3 + 1}, \qquad \dot{x}_3 = \frac{-x_1 - x_2}{x_2 + 2x_3 + 1}. \tag{2.2}$$

Then, the matrix

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 2 \end{pmatrix}$$

is an exact lumping of dimension 2, since

$$\begin{pmatrix} \dot{y}_1 \\ \dot{y}_2 \end{pmatrix} = \begin{pmatrix} \dot{x}_1 \\ \dot{x}_2 + 2\dot{x}_3 \end{pmatrix} = \begin{pmatrix} \frac{(x_2+2x_3)^2}{x_1^2+1} \\ \frac{-2x_2-4x_3}{x_2+2x_3} \end{pmatrix} = \begin{pmatrix} \frac{y_2^2}{y_1^2+1} \\ \frac{-2y_2}{y_2+1} \end{pmatrix} = \begin{pmatrix} g_1(y_1, y_2) \\ g_2(y_1, y_2) \end{pmatrix}.$$

Definition 2.2. Given a system of ODEs of the form given by Equation (2.1), 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.

To answer whether it is possible to recover the evolution of some linear combination of state variables., we introduce the notion of constrained lumping.

Definition 2.3. 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 2.2. Suppose we are interested in observing the quantity $2x_1 + x_2 + 2x_3$ where the evolution is given by the system (2.2). In this case $x_{obs} = Mx$ with $M = (2\ 1\ 2)$. We can see that L from Example 2.1 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 $x_1 + x_2 + x_3$. In this case $M = (1\ 1\ 1)$, thus matrix L, is not a constrained lumping as it is not possible to obtain x_{obs} as a linear combination of y_1 and y_2.

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

Theorem 2.1 (Characterization of Exact Lumping [51])**.** *Given a system of m ODEs of the form given by Equation (2.1) and a matrix $L \in \mathbb{R}^{l\times m}$ with rank l, the following are equivalent.*

(i) The system is exactly lumpable by L.
(ii) For any pseudoinverse $\bar{L}$ of L, $Lf = (Lf) \circ \bar{L}L$.
(iii) $\mathrm{rowsp}(LJ(x)) \subseteq \mathrm{rowsp}(L)$ for all $x \in \mathbb{R}^m$; that is, the row space of L is invariant under $J(x)$ for all $x \in \mathbb{R}^m$, where $J(x)$ is the Jacobian of f at x.

The characterization of exact lumping in Theorem 2.1 provides us with a way to compute a constrained lumping L. This is because, thanks to point (iii), the problem of computing a lumping is equivalent to the problem of finding a $J(x)$-invariant subspace of $\mathbb{R}^m$ for all $x \in \mathbb{R}^m$. However, $J(x)$ is a matrix whose entries are functions of x. In other words, there is a different real-valued matrix for each $x \in \mathbb{R}^m$. To circumvent this issue, we require the following result:

Theorem 2.2 ([37, Lemma 1])**.** *Consider a system of ODEs of the form given by Equation (2.1) and let $J(x)$ be the Jacobian matrix of f. Let $\mathcal{B}$ be any set of real-valued matrices spanning the $\mathbb{R}-$vector space*

$$\mathcal{V}_J := \langle J(x) | x \in \mathbb{R}^m \text{ and } J(x) \text{ is well-defined} \rangle \, .$$

Then L is a lumping of (2.1) if and only if $\mathrm{rowsp}(L)$ is invariant to each $J_i \in \mathcal{B}$.

A set of matrices $\{J_1, \ldots, J_N\}$ can be found analytically when $J(x)$ can be represented as

$$J(x) = \sum_{i=0}^{N} J_i \mu_i(x), \tag{2.3}$$

where $\{\mu_i(x) \mid 0 \leq i \leq N\}$ is a set of analytic functions. When working with polynomial systems, each of the entries of $J(x)$ is a polynomial and so each μ_i corresponds to the monomials of $J(x)$.

For instance, for rational systems, $J(x)$ can be obtained symbolically by differentiating $f(x)$ and then multiplying by the minimum common denominator (mcd) of all entries of $J(x)$. In practice, this approach is computationally unfeasible since this symbolic approach results in an

Algorithm 1 Computation of $\mathcal{V}_J$ from Theorem 2.2.

Input: $f:\mathbb{R}^m\to\mathbb{R}^m$, and a compact set $\Omega\subseteq\mathbb{R}^m$

set $N=0$
repeat
  set $N=N+1$
  compute x_N sampling uniformly from Ω
  compute $J_N=J(x_N)$ using automatic differentiation
until $J_N\in\langle J_i|1\leq i\leq N-1\rangle$
return $\{J_1,\ldots,J_{N-1}\}$.

explosion of the number of monomials in $J(x)$. To avoid any explicit symbolic computation, a representation $\{J_1,\ldots,J_N\}$ can be obtained by sampling $f(x)$ at different points and using automatic differentiation [37, Section 3.4]. The general idea of this sampling procedure is formalized by Algorithm 1. In [37], the authors argue that a uniform sampling of $J(x)$ values finds almost surely $\mathcal{V}_J$.

Example 2.3. Consider the system from Example 2.1. Before finding a basis for $\mathcal{V}_J$ using the sample-based approach, we exemplify the symbolic approach to display its complexity. It starts by symbolically differentiating $f(x)$ to obtain $J(x)$:

$$J(x)=\begin{pmatrix} \frac{-2x_1x_2^2-8x_1x_2x_3-8x_1x_3^2}{x_1^4+2x_1^2+1} & \frac{2x_2+4x_3}{x_1^2+1} & \frac{4x_2+8x_3}{x_1^2+1} \\ \frac{2}{x_2+2x_3+1} & \frac{-2x_1+4x_3}{x_2^2+4x_2x_3+4x_3^2+2x_2+4x_3+1} & \frac{-4x_1-4x_2-4}{x_2^2+4x_2x_3+4x_3^2+2x_2+4x_3+1} \\ \frac{-1}{x_2+2x_3+1} & \frac{x_1-2x_3-1}{x_2^2+4x_2x_3+4x_3^2+2x_2+4x_3+1} & \frac{2x_1+2x_2}{x_2^2+4x_2x_3+4x_3^2+2x_2+4x_3+1} \end{pmatrix}.$$

We then find the mcd of all entries of $J(x)$, which is $q(x)=(x_1^2+1)^2(x_2+2x_3+1)^2$. Since $J(x)$ is rational in all its entries, it follows that $J(x)=B(x)/q(x)$, where $B(x)=q(x)J(x)$ is a matrix with polynomial entries up to degree 5. Notice that to obtain a monomial representation of $B(x)$, we need to expand all its terms.

These complex symbolic computations can be avoided by using Algorithm 1. Following the efficient sampling method outlined in [37], we find that the dimension of $\mathcal{V}_J$ is 5. The set $\{J_1,\ldots,J_5\}$ can be obtained by evaluating $J(x)$ at the following points: $x_1=(1,0,0),\ x_2=(0,1,0),\ x_3=(0,0,1),\ x_4=(1,5,2),\ x_5=(3,3,2)$. These points lead to the matrices J_i:

$$J_1=\begin{pmatrix} 0 & 0 & 0 \\ 2 & -2 & -8 \\ -1 & 0 & 2 \end{pmatrix}, \qquad J_2=\begin{pmatrix} 0 & 2 & 4 \\ 1 & 0 & -2 \\ -0.5 & -0.25 & 0.5 \end{pmatrix},$$

$$J_3=\begin{pmatrix} 0 & 4 & 8 \\ 0.667 & 0.444 & -0.444 \\ -0.333 & -0.333 & 0 \end{pmatrix}, \qquad J_4=\begin{pmatrix} -40.5 & 9 & 18 \\ 0.200 & 0.060 & -0.280 \\ -0.100 & -0.040 & 0.120 \end{pmatrix},$$

$$J_5=\begin{pmatrix} -2.94 & 1.4 & 2.8 \\ 0.25 & 0.31 & -0.438 \\ -0.125 & -0.031 & 0.188 \end{pmatrix}.$$

We see that Algorithm 1 outputs a set of matrices $\{J_1,\ldots,J_N\}$ spanning $\mathcal{V}_J$. Given such a set, Theorem 2.2 provides an algorithmic way to find a constrained lumping L by performing a finite number of checks over real-valued vectors. This is shown in Algorithm 2. The use of Algorithms 1 and 2 to obtain a constrained lumping L is summarized in Algorithm 3.

Algorithm 2 Computation of L

Input: a set of matrices $\{J_1, \ldots, J_N\}$ spanning $\mathcal{V}_J$ (Theorem 2.2);
a $p \times m$ matrix M with row rank p.

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

Algorithm 3 Constrained lumping [26]

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

compute a set of matrices $\{J_1, \ldots, J_N\}$ spanning $\mathcal{V}_J$ (Algorithm 1)
compute L (Algorithm 2)
return Constrained lumped ODE system $\dot{y} = Lf(\bar{L}y)$.

The number of checks in Algorithm 2 is proportional to the number of elements in $\mathcal{V}_J$. Moreover, Algorithm 2 can be implemented in polynomial time [26,37].

3. Constrained Approximated Lumping

This section introduces the formal theory underlying approximate constrained lumping, as well as its computational aspects. In Section 3(a) we present the definition of approximate constrained lumping. We introduce the *deviation tolerance* η, which relaxes the conditions for lumping granting additional aggregation power. We prove how the error bound depends on the deviation tolerance. Next, we focus on how to compute an approximate constrained lumping. In Section 3(b) we introduce a numerical lumping tolerance ε to compute a lumping matrix L and show how it relates to the deviation tolerance. In Section 3(c), instead, we discuss how to pick an appropriate value for the lumping tolerance ε in a way that leads to satisfactory results for both polynomial and rational systems.

(a) Definition and error estimation

We begin by demonstrating why the condition for exact lumping can be too restrictive for finding reductions. Consider the following system of ODEs:

$$\dot{x}_1 = \frac{x_2^2 + 4.05x_2x_3 + 4x_3^2}{x_1^2 + 1},\ \dot{x}_2 = \frac{2x_1 - 4x_3}{x_2 + 2x_3 + 1},\ \dot{x}_3 = \frac{-x_1 - x_2}{x_2 + 2x_3 + 1}. \tag{3.1}$$

Notice that the system given by Equation (3.1) is similar to that of Equation (2.2), as we have just added a term $0.05x_2x_3$ to the numerator of its first equation. While the initial system is lumpable by $L = (1\ 0\ 0, 0\ 1\ 2)^T$, this new system is not exactly lumpable by L. However, we would like to know if it is still possible to use the matrix L to obtain a *useful* reduced system that approximates well the original one. To do so, we first want to evaluate how *close* the matrix L is to (i.e., how much it *deviates* from) an exact lumping.

By Theorem 2.1, Point (ii), 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 $x \in \mathcal{D}(f) \subseteq \mathbb{R}^m$, where $\mathcal{D}(f)$ denotes the domain of f. In our proposal of approximate lumping, we relax this requirement by asking it to be satisfied up to a certain tolerance.

Definition 3.1. Consider the system of ODEs in Equation (2.1). Let $L \in \mathbb{R}^{l \times m}$ be a full rank matrix with $l < m$ and denote by $\bar{L}$ the Moore-Penrose pseudoinverse of L and by $P_L = \bar{L}L$ the

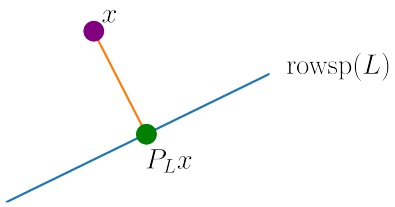


Figure 2: Points projected to $\mathrm{rowsp}(L)$ [39].

orthogonal projection operator onto $\mathrm{rowsp}(L)$. We define the *deviation of* L *at* $x \in \mathcal{D}(f) \cap \mathcal{D}(f \circ P_L)$ by

$$\mathrm{dev}_L(f, x) := \|Lf(\bar{L}Lx) - Lf(x)\|_2. \tag{3.2}$$

To understand the intuition behind Definition 3.1, consider the point x in Figure 2. Then, the deviation computes the difference between the images under Lf of x (purple) and $P_L x$ (green). The deviation is identically zero if and only if L is a lumping.

Example 3.1. Consider the matrix L given in Example 2.1. A pseudoinverse of L is given by $\bar{L} = (1\,0\,0, 0\,0.2\,0.4)^T$. Writing g for the corresponding vector field, we obtain that $dev_L(g, (1,1,1)^T) = 0$, as L is an exact lumping. Now consider the system given by Equation (3.1) and use h to denote the corresponding vector field. Then, we get that $\mathrm{dev}_L(h, (1,1,1)^T) = 0.007$. Therefore, the matrix L is not an exact lumping of Equation (3.1).

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. While the theory in [26,37] guarantees that reduced models provide exact predictions, this might restrict the actual aggregation power of the technique. We aim to relax this theory by considering reductions that need not be exact or tight. Having this in mind, we introduce the following notion.

Definition 3.2. Consider the ODEs from Equation (2.1), a set of initial conditions S, and a finite time horizon $T > 0$ such that x is well-defined on $[0, T]$ for all initial conditions $x(0) \in S$. Given a full rank matrix $L \in \mathbb{R}^{l \times m}$ and $\eta > 0$, we say that Equation (2.1) is *approximately lumpable* by L with *deviation tolerance* η if

$$\mathrm{dev}_L(f, x(t)) \leq \eta, \tag{3.3}$$

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

Remark 3.1. The notion of approximate lumping generalizes that of exact lumping from Definition 2.1. To see this, suppose L is an exact lumping. Then by Point (ii) of Theorem 2.1, $\|Lf(\bar{L}Lx) - Lf(x)\|_2 = \mathrm{dev}_L(f, x) = 0$, for all $x \in \mathbb{R}^m$. It follows that L is a $(\mathbb{R}^m, \infty, 0)$-lumping.

In practice, modellers often use biological models to study the evolution of a system $\dot{x} = f(x)$ for a fixed set S of initial conditions and a time horizon T. The assumption is that the behaviour of interest to the modeler occurs within time T.

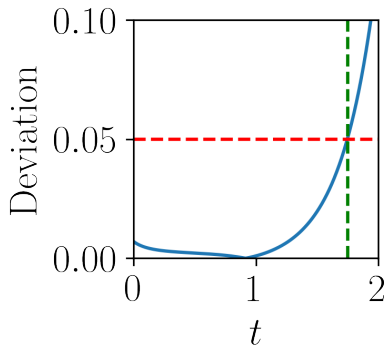


Figure 3: Example 3.2: evolution of $\mathrm{dev}_L(f(x(t)))$.

Example 3.2. Consider the system in Equation (3.1), the matrix L of Example 2.1 and let $x(0) = (1,1,1)^T$. Then L is an approximate lumping for $x(0)$, time $T = 1.75$, and deviation tolerance 0.05, i.e., L is a $(\{x(0)\}, 1.75, 0.05)$-lumping. To see this, note that in Figure 3 the deviation of the dynamics is bounded by 0.05.

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

Definition 3.3. 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 Equation (2.1) is an *approximate constrained lumping* with *observables* x_{obs} if $\mathrm{rowsp}(M) \subseteq \mathrm{rowsp}(L)$.

Similarly to Definition 2.2, Definition 3.3 means that the observables x_{obs} can be recovered as a linear combination of the reduced variables $y = Lx$.

Definition 3.4. The *reduced system* induced by an approximate lumping L is given by $\dot{y} = Lf(\bar{L}y)$, where $y \in \mathbb{R}^l$.

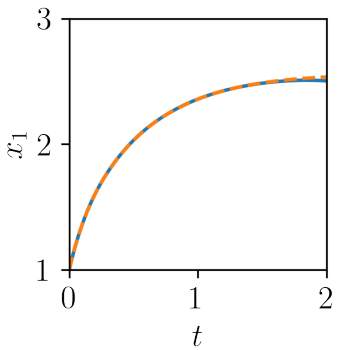
Example 3.3. Consider the system in Equation (3.1) and the matrix L given in Example 2.1. We can compute the reduced system as follows

$$\begin{pmatrix}\dot{y}_1\\ \dot{y}_2\end{pmatrix} = Lf\left(\bar{L}\begin{pmatrix}y_1\\ y_2\end{pmatrix}\right) = Lf\left(\begin{pmatrix}y_1 & 0\\ 0 & 0.2y_2\\ 0 & 0.4y_2\end{pmatrix}\right) = \begin{pmatrix}\frac{1.004y_2^2}{y_1^2+1}\\ \frac{-2y_2}{y_2+1}\end{pmatrix}.$$

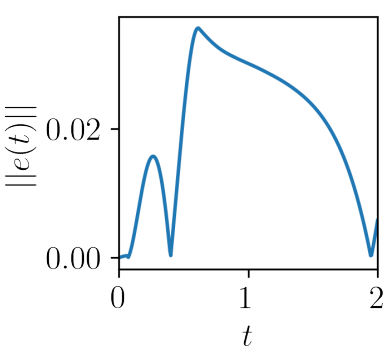
Definition 3.5. Given the system in Equation (2.1) and an approximate lumping L, we define the *error* of the reduction by $e(t) := y(t) - Lx(t)$, where y is the solution of the reduced system given by Definition 3.4. The evolution of the approximation error is given by $\dot{e} = \dot{y} - L\dot{x}$, with $e(0) = 0$.

Example 3.4. Following Example 3.3, we compute the trajectories of the reduced and original systems. We also compute the L_2-error of the reduction. This is summarized in Figure 4. Note that, in this example, we obtained a reduction of a system that was not exactly lumpable. In Figure 4a, we can appreciate that, for the given time horizon, the reduced and original values are close to each other.

Next, we show that it is possible to bound the error introduced by approximate constrained lumping.



(a) Evolution of the reduced (orange) and original (blue).



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

Figure 4: Reduced system and error computation for Example 3.3 using the matrix L of Example 2.1 on Equation (3.1).

Theorem 3.1 (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 ODE system $\dot{x} = f(x)$ is bounded on $[0, T]$. If there exists a compact set Ω such that $x(t), \bar{L}Lx(t) \in \Omega$ for all $t \in [0, T]$ and all $x(0) \in S$ and f is analytic in Ω, then 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*

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

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

The error bound exhibits an exponential dependence on the chosen time horizon T. Despite being a conservative estimation, it supports the consistency of the approach because it shows that the error is of order $\mathcal{O}(\eta)$. Moreover, as we will see in Section 4, our framework can find approximate lumpings with low empirical errors for published biological models.

(b) Lumping Algorithm

In this section we relax the condition in Line 5 of Algorithm 2, thus allowing us to find approximate reductions. Intuitively, we add a new row rJ_i to the matrix L only if it is *far enough* from $\mathrm{rowsp}(L)$. 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 $\mathrm{rowsp}(L)$. Thus, we propose Algorithm 4.

Remark 3.2. In Line 7 of Algorithm 4 we do not append rJ_i, given by the orange vector in Figure 5. Rather, we append the normalized component of rJ_i in the orthogonal direction to $\mathrm{rowsp}(L)$, which is $rJ_i - \pi_i$, the green vector in Fig. 5. This ensures that the matrix L is orthonormal. 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 Algorithm 4.

Example 3.5. Consider the system given by Equation (3.1) and let $\varepsilon = 0.2$. Set $M = (1, 0, 0)$ i.e., we are observing the component x_1 from the original system. Using Algorithm 1 with the same points as in Example 2.3, we find that the dimension of $\mathcal{V}_J$ is 6 with basis given by the matrices:

Algorithm 4 Approximate Constrained Lumping Algorithm

Input: numerical threshold $\varepsilon \geq 0$, and a set of matrices $\{J_1, \ldots, J_N\}$ spanning $\mathcal{V}_J$ (Theorem 2.2), and 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.

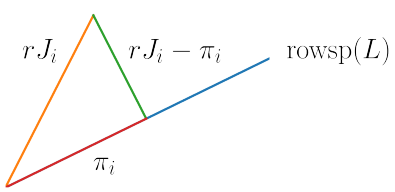


Figure 5: Decomposition of rJ_i into $\mathrm{rowsp}(L)$ and $\mathrm{rowsp}(L)^\perp$ [39].

$$J_1 = \begin{pmatrix} 0 & 0 & 0 \\ 2 & -2 & -8 \\ -1 & 0 & 2 \end{pmatrix}, \qquad J_2 = \begin{pmatrix} 0 & 2 & 4.05 \\ 1 & 0 & -2 \\ -0.5 & -0.25 & 0.5 \end{pmatrix},$$

$$J_3 = \begin{pmatrix} 0 & 4.05 & 8 \\ 0.667 & 0.444 & -0.444 \\ -0.333 & -0.333 & 0 \end{pmatrix}, \qquad J_4 = \begin{pmatrix} -40.75 & 9.05 & 18.125 \\ 0.200 & 0.060 & -0.280 \\ -0.100 & -0.040 & 0.120 \end{pmatrix},$$

$$J_5 = \begin{pmatrix} -2.958 & 1.41 & 2.815 \\ 0.25 & 0.031 & -0.438 \\ -0.125 & -0.031 & 0.188 \end{pmatrix}, \qquad J_6 = \begin{pmatrix} -0.951 & 0.621 & 1.235 \\ 0.222 & 0.025 & -0.395 \\ -0.111 & -0.025 & 0.173 \end{pmatrix}.$$

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

$$rJ_1 = (0, 0, 0), \qquad rJ_2 = (0, 2, 4.05),$$
$$rJ_3 = (0, 4.05, 8), \qquad rJ_4 = (0, 9.05, 18.125),$$
$$rJ_5 = (0, 1.41, 2.815), \qquad rJ_6 = (0, 0.621, 1.235).$$

Let $d_i := \|rJ_i - \pi_i\|_2$. By the check of Line 6, we do not append any new row to L as $d_1 = 0$ since $rJ_1 = \pi_1 = (0, 0, 0)$. Next, we compute π_2 following Line 5. As $\pi_2 = 0$, it follows that $d_2 = \|rJ_2\|_2 = 4.52 > 0.2$, 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

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 0.443 & 0.897 \end{pmatrix}. \tag{3.4}$$

Going back to Line 5 with the updated matrix L, we compute $\pi_3 = (0, 3.974, 8.046)$. Following Line 6, we do not add additional rows to L since $d_3 = 0.089 < 0.2$. We compute $\pi_4 = (-40.75, 8.978, 18.18)$ and $d_4 = 0.09 < 0.2$, leaving L unchanged. We go on with the computations for the remaining matrices J_i to obtain $d_5 = 0.018,\ d_6 = 0.01$. It follows that we need not add any new rows to L.

So far we have only checked the first row of L. Following the main loop (Line 4), we set $r = (0, 0.443, 0.897)$. We compute

$$rJ_1 = (-0.011, -0.869, -1.760), \qquad rJ_2 = (-0.006, -0.218, -0.441),$$
$$rJ_3 = (-0.004, -0.098, -0.199), \qquad rJ_4 = (-0.001, -0.008, -0.017),$$
$$rJ_5 = (-0.001, -0.013, -0.026), \qquad rJ_6 = (-0.001, -0.010, -0.021).$$

We continue computing $d_1 = 0.02,\ d_2 = 0.007,\ d_3 = 0.004$, while $d_i = 0.001$ for $i = 4, 5, 6$. It follows that the check of Line 6 is false for all $i = 1, \ldots, 6$; meaning that no more rows should be added to L, thus terminating the algorithm. The output of Algorithm 4 is the matrix L of Equation (3.4).

The following result states that approximate constrained lumpings can be efficiently computed.

Theorem 3.2 (Time Complexity of Algorithm 4)**.** *Approximate constrained lumping can be computed in polynomial time. Specifically, the complexity of Algorithm 3 when using Algorithms 1 and 4 as subroutines can be bounded by $\mathcal{O}((A + Np(p+2))m^2)$, where A is the cost of computing an entry of $f(x)$, m is the dimension of f, N is the number of matrices spanning $\mathcal{V}_J$, and p is the dimension of the reduced system.*

When the system is not approximately lumpable, it follows that the worst-case time complexity is $O(m^4)$. The complexity bound of Theorem 3.2 can be further improved by using more efficient data structures [26].

The next result relies on Theorem 2.2 and ensures that the approximate reductions found by Algorithm 4 admit a deviation tolerance of order $\mathcal{O}(\varepsilon)$. In other words, the deviation tolerance η is in the order of the lumping tolerance ε.

Theorem 3.3 (Correctness of Algorithm 4)**.** *Using the setting of Theorem 3.1, assume $J(x)$ can be written in the form given by Equation (2.3). Let L be the matrix computed via Algorithm 4 with lumping tolerance ε over the sampled matrices $\{J(x_1), \ldots, J(x_N)\}$ for $x_1, \ldots, x_N \in \Omega$. Then, L is a $(T, S, \sqrt{m}C'CK\varepsilon)$-lumping, where $C' = \sup_{i,x\in\Omega} |\mu_i(x)|$, C is a constant depending on the representation of $J(x)$ and K is a constant depending on the trajectories $x(t)$ and $\bar{L}Lx(t)$.*

(c) Heuristic search of lumping tolerance

To apply Algorithm 4, it is important to choose an appropriate value for ε. While Theorems 3.1 and 3.3 allow for an error estimation 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, it is possible to lump more variables together at the price of larger approximation errors. We would like to find the largest admissible ε such that the approximation error is small enough. To this aim, we begin by noting 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 3.1 gives us an upper bound for ε which can be computed for each model.

Lemma 3.1 (Upper bound on ε). *Consider a matrix of observables $L \in \mathbb{R}^{p\times m}$ of rank p with $j-$th row denoted by r_j and a decomposition of the Jacobian $J(x) = \sum_{i=1}^{n} J_i\mu_i(x)$. Let $\varepsilon_{\max}$ be given by*

$$\varepsilon_{\max} = \max_{j,i} \|r_j J_i - r_j J_i P_L\|_2, \tag{3.5}$$

for $i = 1, \ldots, n$ and each $j = 1, \ldots, l$, where P_L is the projection onto $\mathrm{rowsp}(L)$. *Then ε_{max} is the smallest ε such for any $\varepsilon \geq \varepsilon_{\max}$ the output of Algorithm 4 will be a matrix of orthonormal rows spanning the row space of L.*

Lemma 3.1 states that any $\varepsilon \geq \varepsilon_{\max}$ will collapse the dynamics of the system onto observable M. In this way, for any given model, we have a range $[0; \varepsilon_{\max}]$ from which to choose the right value for ε.

Remark 3.3. The value of $\varepsilon_{\max}$ depends not only on the chosen model and observable but also on the set of matrices $\mathcal{V}_J$ used to represent $J(x)$. Compared to polynomial models, the values for the drift in rational models can be more sensitive to changes in x, especially when the denominator of $f(x)$ is close to 0. This sensitivity can be explained by vanishing denominators in the Jacobian $J(x)$ of a rational drift $f(x)$. We therefore expect rational models to exhibit larger fluctuations of $\varepsilon_{\max}$.

For example, the evolution of the reductions for different values of ε is demonstrated for a model from the literature in Figure 6. The model, named BIOMD103, has been downloaded from the online repository ODEbase [52] discussed in greater detail in Section 4 (it is model 2 from Table 1).

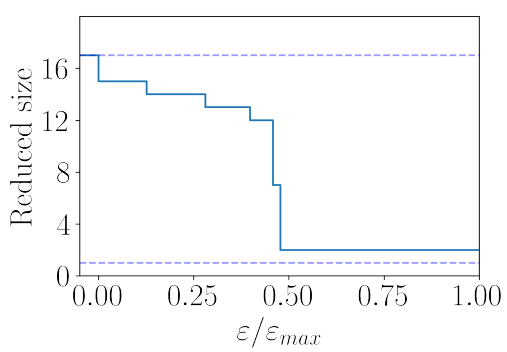


Figure 6: Reduced size vs relative ε for the model BIOMD103.

We note that previous work [39] used the deviation as a proxy to estimate the error without having to simulate the original system. This approach relied on a Monte Carlo approximation of the L_1 norm of dev_L over $[0, \|x^0\|_2]^m$, where x^0 is the center of the initial set. While being effective in finding meaningful reductions for polynomial systems, this deviation-based approach can be problematic for rational systems. This is because rational systems are more sensitive to the choice of compactum, since a small change in the denominator can have a large effect on the overall value of $f(x)$, especially for values of x where the denominator approaches 0.

We propose a novel approach which gives satisfactory results for both polynomial and rational systems (Section 4). Using the fact that the size of a reduced model decreases monotonically with ε, we set up a target (a *cutoff* size, m^*) for the reduced model given as a percentage of its original

size. We then apply a binary search algorithm to find the largest ε such that the reduced model's size m_ε satisfies $m_\varepsilon \leq m^*$. In terms of Figure 6, we want to obtain the smallest ε whose induced size is below the cutoff size. This intuition is formalized by Algorithm 5.

Algorithm 5 Finding the smallest acceptable ε for Algorithm 4

Input: set of matrices $J_i,\ i = 1, \ldots, N$ spanning $\mathcal{V}_J$, and cutoff reduced size m^*, and minimal difference $d_{\min}$

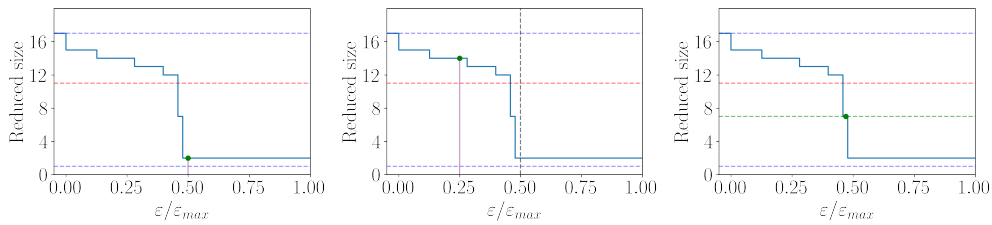
1: **set** $\varepsilon_{\min} = 0$
2: **compute** $m_{\min} = nrows(L_{\varepsilon_{\min}})$,
$\varepsilon_{\max} = \max_{j,i} \|r_j J_i\|_2$,
$m_{\max} = nrows(L_{\varepsilon_{\max}})$
3: **if** $m^* < m_{\min}$ **then**
4: **return** $\varepsilon_{\min}$
5: **else if** $m^* > m_{\max}$ **then**
6: **return** $\varepsilon_{\max}$
7: **end if**
8: **repeat**
9: **compute** $\varepsilon = (\varepsilon_{\max} + \varepsilon_{\min})/2$
10: **compute** L using Algorithm 4 with ε
11: **if** $nrows(L) < m^*$ **then**
12: **set** $\varepsilon_{\max} = \varepsilon$
13: **else**
14: **set** $\varepsilon_{\min} = \varepsilon$
15: **end if**
16: **compute** $d = \varepsilon_{\max} - \varepsilon_{\min}$
17: **until** $d < d_{\min}$
18: **return** $\varepsilon_{\max}$.

Remark 3.4. It is important to notice that the choice of minimal different d_{min} in Algorithm 5 provides a trade-off between detecting reductions and the speed of the approach. A lower value for d_{min} can lead to finding more reductions at the cost of more iterations.

Example 3.6. Let us use again model BIOMD103 from Figure 6. It has an observable of interest, $C3$ [53], consisting of one variable representing the concentration of*Activated capsase3*. We would like to find a reduction such that the reduced model size is at most 62.5% of the original size (17), i.e., 11 species. To do so, we use Algorithm 5. We first verify (Line 3) that the cutoff size is larger than the minimum size and smaller than the size of the exact reduction. As the cutoff size 11 is between 1 and 17 we continue to Line 8. On the first iteration (Figure 7a) the middle point (green) gives a reduction of size 2. As this size is smaller than the goal size 11, we set $\varepsilon_{\max} = \varepsilon$. This gives a new interval to search for a reduction (black line on Figure 7b). We continue following the algorithm until it converges to a reduction of size 7 (green line Figure 7c). This is a correct result as we can see that the closest step below the cutoff size is the one of size 7 in Figure 7a.

4. Evaluation

In this section, we evaluate our method. All experiments were performed using an extension of the tool for CLUE [26,37,40], an open-source Python library. The latest version of CLUE is available in



(a) First iteration (b) Second iteration (c) Last iteration

Figure 7: Algorithm 5 run on model 2 to reduce it to size at most 11 (red line).

https://github.com/clue-developers/CLUE/

The experiments are available in the following repository.

https://doi.org/10.5281/zenodo.15388842

In Section 4(a), we use a selection of models from the literature to show how our approach can lead to small reductions while not incurring a large error. In Section 4(b), we use a perturbed multisite protein phosphorylation model to show the scalability of our approach.

(a) Reduction power

To evaluate the reduction power of our approach, we selected several models from the literature. These models are chemical reaction networks with 5 or more species. We used two sources to select models. The first is a collection of polynomial models from the literature written in the input format of the tool BioNetGen [54]. We selected two such models. The second is the so-called ODEbase project [52] which offers importing/exporting capabilities for polynomial and rational models from the BiomodelsDB repository [55]. We selected six such models. Models with prefix `BIOMD` were taken from ODEbase, while the remaining models were taken directly from the supplementary material of their original paper. For each model, the observable was chosen according to the observables studied in the original paper describing the model. These typically consist of one variable or linear combinations of them. For models where more than one observable was studied in the original paper, we arbitrarily selected one of them. Models written in the BioNetGen format were first imported by ERODE [56], our tool that collects many techniques for the reduction of biological systems, and then exported in ERODE's `.ode` format supported by CLUE.

To import models from the ODEbase project [52], instead, we directly implemented in CLUE an importer to convert them in `.ode` format.

When available, the set of initial conditions necessary for simulations was taken from the original paper or the model files in BiomodelsDB [55]. The time horizon was either taken from the original reference or chosen experimentally by noticing when steady state was reached. All this additional information was manually added to the `.ode` files to the corresponding model.

When choosing the models, we aimed for a selection that could equally display our approach for both rational and polynomial models. The final selection of models is displayed in Table 1. The column *Type* reports whether the model is polynomial (P) or rational (R).

The source paper presenting the model is displayed in the *Ref.* column. The column *Description* contains a summary of the biological system studied in the model. In column *Observable*, we can see the species (or linear combinations of them) chosen as a constraint for the lumping. Finally, column *Size* displays the number of model variables added in the observable over the total number of variables in the system (e.g., in Model 5, S2P sums 10 out of the 24 variables of the model).

Nr.	Name	Type	Ref.	Description	Observable	Size
1	BIOMD102	P	[53]	Signaling pathways involved in the initiation of apoptosis (wild-type model)	Activated capsase3 (C3)	1/12
2	BIOMD103	P	[53]	Signaling pathways involved in the initiation of apoptosis (competitive model)	Activated capsase3 (C3)	1/17
3	BIOMD447	P	[57]	Thrombospondin-Dependent Activation of TGF-β1	Transforming growth factor (TGF)-β1 (TGFb1)	1/13
4	BioNetGen_CPP	P	[58]	Central carbon pathway of E. coli	1,3-diphosphoglycerate (D13PG)	2/87
5	NIHMS80246_S6	P	[28]	FceRI-like network of a cell-surface receptor	Phosphorylation at Y2 (S2P)	10/24
6	BIOMD437	R	[59]	Circadian clock of the fungus Neurospora crassa	Frequency of $mRNA$ (frq_RNA)	1/39
7	BIOMD448	R	[60]	Insulin Signaling in Type 2 Diabetes	Phosphorylation of site S6 (S6P)	1/23
8	BIOMD488	R	[61]	Effect of Aβ-immunization in Alzheimer's disease	Microtubular binding protein tau (Tau)	1/68

Table 1: List of selected models used for evaluation.

For each model, we used Algorithm 5 with $d_{min} = 10^{-6}$ to find the largest lumping tolerance ε such that the size of the reduced model is larger than a given bound. The bound was given as a ratio of the size of the original system. We used ratios from 12.5% to 87.5% using a step size of 12.5%. This means that, in the case of Model 2, the initial size was 17 and so the cutoff sizes were $[2.125, 4.25, 6.375, 8.5, 10.625, 12.75, 14.875]$. Notably, different ratios can lead to the same reduction. Consider Model 5. Here, the exact reduction is 34.5% of the original size, so the output of Algorithm 5 is 0 for all percentages above 34.5% as they recover the exact reduction. In such cases, only one result is shown.

Table 2 presents the results for all models from Table 1. Each model has a row with information constant across experiments, including the value for $\varepsilon_{\max}$. The first column $|Red.|$ shows the size of the reduced model obtained by Algorithm 4, while column *Red. ratio* shows the ratio between the size of the reduced model and that of the original model as a percentage. We display the absolute and the relative empirical errors at the time horizon, in the columns $e(T)$ and $e(T)_{Rel}$, respectively. Here, the relative error is given as the absolute error divided by the value of the observable of the exact reduction. Column e_{max} shows the maximum value for the empirical error at a time t, $e(t)$, within the time horizon. The obtained value of ε and its ratio w.r.t. $\varepsilon_{\max}$ are shown in the columns ε and $\varepsilon/\varepsilon_{\max}$, respectively. The column $Iter.$ shows the number of iterations and the average computation time of Algorithm 5 over 5 runs. We used a 4.7 GHz Intel Core i7 computer with 32 GB of RAM to carry out all computations. For each model, we plot in Figure 8 the corresponding simulations of the observables for the considered values of ε. We used

`Scipy` with `RK45` and `LSODA` solvers to simulate polynomial and rational models respectively. Table 2 and Figure 8 show the successful use of Algorithm 5 on both polynomial and rational models.

$\|Red.\|$	*Red. ratio(%)*	$e_{Rel}(T)$	$e(T)$	e_{max}	ε	$\varepsilon/\varepsilon_{\max}$ (%)	*Iter.*	*Time (s)*
1) Model: BIOMD102, size: 13, type: P, exact lumping: 13, $\varepsilon_{\max}$: 3.52E-03								
11	84.6	1.16E+01	1.44E+02	1.44E+02	4.24E-04	1.21E+01	13	0.284
8	61.5	4.86E+00	6.04E+01	6.78E+01	9.95E-04	2.83E+01	13	0.099
6	46.2	1.26E+00	1.56E+01	7.39E+01	1.73E-03	4.91E+01	13	0.138
2	15.4	1.50E+01	1.86E+02	1.86E+02	1.97E-03	5.60E+01	13	0.046
1	7.7	1.50E+01	1.86E+02	1.86E+02	3.52E-03	1.00E+02	13	0.052
2) Model: BIOMD103, size: 17, type: P, exact lumping: 17, $\varepsilon_{\max}$: 3.52E-03								
14	82.4	1.02E-02	1.72E+00	1.72E+00	9.89E-04	2.81E+01	13	0.598
12	70.6	9.25E-01	1.55E+02	1.57E+02	1.60E-03	4.56E+01	13	0.288
7	41.2	8.52E-01	1.43E+02	1.44E+02	1.64E-03	4.67E+01	13	0.218
2	11.8	1.82E-01	3.06E+01	3.06E+01	2.00E-03	5.67E+01	13	0.070
3) Model: BIOMD447, size: 13, type: P, exact lumping: 13, $\varepsilon_{\max}$: 2.45E+01								
11	84.6	1.33E-03	2.02E-05	2.02E-05	4.95E-03	2.02E-02	26	0.173
9	69.2	1.35E-02	2.04E-04	2.04E-04	3.41E-01	1.39E+00	26	0.122
7	53.8	8.46E-03	1.28E-04	1.29E-04	3.75E-01	1.53E+00	26	0.123
6	46.2	6.99E-03	1.06E-04	7.02E-04	1.40E+00	5.71E+00	26	0.105
3	23.1	1.22E-01	1.86E-03	7.31E-03	1.98E+01	8.10E+01	26	0.056
4) Model: BioNetGen_CCP, size: 87, type: P, exact lumping: 30, $\varepsilon_{\max}$: 2.83E+00								
30	34.5	5.86E-06	6.11E-05	5.22E-04	6.74E-07	2.38E-05	23	1.807
10	11.5	3.14E-01	3.27E+00	3.27E+00	2.67E-01	9.44E+00	23	2.959
5) Model: NIHMS80246_S6, size: 24, type: P, exact lumping: 19, $\varepsilon_{\max}$: 3.13E-01								
19	79.2	3.16E-11	3.16E-09	2.14E-04	5.97E-07	1.91E-04	20	1.370
17	70.8	8.09E-05	8.09E-03	4.86E+01	2.23E-02	7.13E+00	20	1.948
14	58.3	8.09E-05	8.09E-03	4.89E+01	2.34E-02	7.49E+00	20	1.478
9	37.5	1.00E+00	1.00E+02	1.00E+02	3.00E-02	9.58E+00	20	0.809
8	33.3	1.00E+00	1.00E+02	1.00E+02	1.29E-01	4.10E+01	20	0.586
1	4.2	1.00E+00	1.00E+02	1.00E+02	3.13E-01	1.00E+02	20	0.454
6) Model: BIOMD437, size: 39, type: R, exact lumping: 21, $\varepsilon_{\max}$: 7.11E+05								
21	53.8	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1	0.044
12	30.8	1.58E-04	2.00E-05	6.66E-05	6.47E-07	9.09E-11	41	3.021
5	12.8	1.58E-04	2.00E-05	6.66E-05	1.07E+02	1.50E-02	41	3.645
4	10.3	3.31E-01	4.19E-02	4.29E-02	3.33E+02	4.69E-02	41	3.237
7) Model: BIOMD448, size: 27, type: R, exact lumping: 23, $\varepsilon_{\max}$: 1.00E+00								
23	85.2	0.00E+00	0.00E+00	0.00E+00	0.00E+00	0.00E+00	1	0.002
19	70.4	7.45E-08	6.79E-06	2.90E-02	1.60E-02	1.60E+00	21	0.912
1	3.7	9.21E-01	8.39E+01	8.39E+01	1.00E+00	1.00E+02	21	0.291
8) Model: BIOMD488, size: 68, type: R, exact lumping: 64, $\varepsilon_{\max}$: 7.50E+10								
59	86.8	2.88E-02	1.05E-02	2.85E-02	1.50E+02	2.00E-07	58	47.800
50	73.5	1.97E-01	7.20E-02	7.20E-02	7.50E+04	1.00E-04	58	46.542
9	13.2	9.06E+00	3.30E+00	3.61E+00	1.50E+06	2.00E-03	58	7.276
8	11.8	9.06E+00	3.30E+00	3.61E+00	6.00E+07	8.00E-02	58	9.505

Table 2: Approximate constrained lumping results for the models from Table 1.

As expected, Algorithm 5 can recover exact reductions, as seen in Models 6 and 7. Similarly, having a small cutoff size (12.5% of the original model size or lower) results in aggressive reductions that collapse the dynamics as seen in the reduction to 8 species of Model 5 (Figure 8e). Taken together, these experiments display the following behaviours:

(i) *Models that do not admit exact reduction may admit approximate reduction.* Models 1, 2, and 3 were not exactly lumpable but were approximately lumped. For models 1 and 2, the reductions that incurred acceptable errors were modest, at around 80% of the original model size. In the case of Model 3, all reductions incurred in low error, with the smallest having only 3 species.

(ii) *Models that admit limited exact reduction, may admit smaller approximate reductions.* Models 5, 7, and 8 had modest exact reductions, while they could be further reduced using approximate lumping. In this case, most of the approximate reductions have a small error. The improvement compared to the exact case goes from modest, at around 90% of the size of the exact reduction, to as low as 47% of the exactly reduced size (Model 5).

(iii) *Models that admit notable exact reduction, may admit even smaller approximate reductions.* Models 4 and 6 had already significant exact reductions; nevertheless, they could be further reduced via approximate lumping. In this case, Model 4 shows a significant steady-state error for the approximate reduction with 10 species. For Model 6, it was possible to find a reduction with less than 15% of the original model size while still obtaining simulation results close to the original simulation.

While $e(t)$ increases monotonically in most cases, this does not hold for all reductions. e.g., in Figures 8a and 8e, the reductions with 7 and 14 species get closer to the original simulations as time increases.

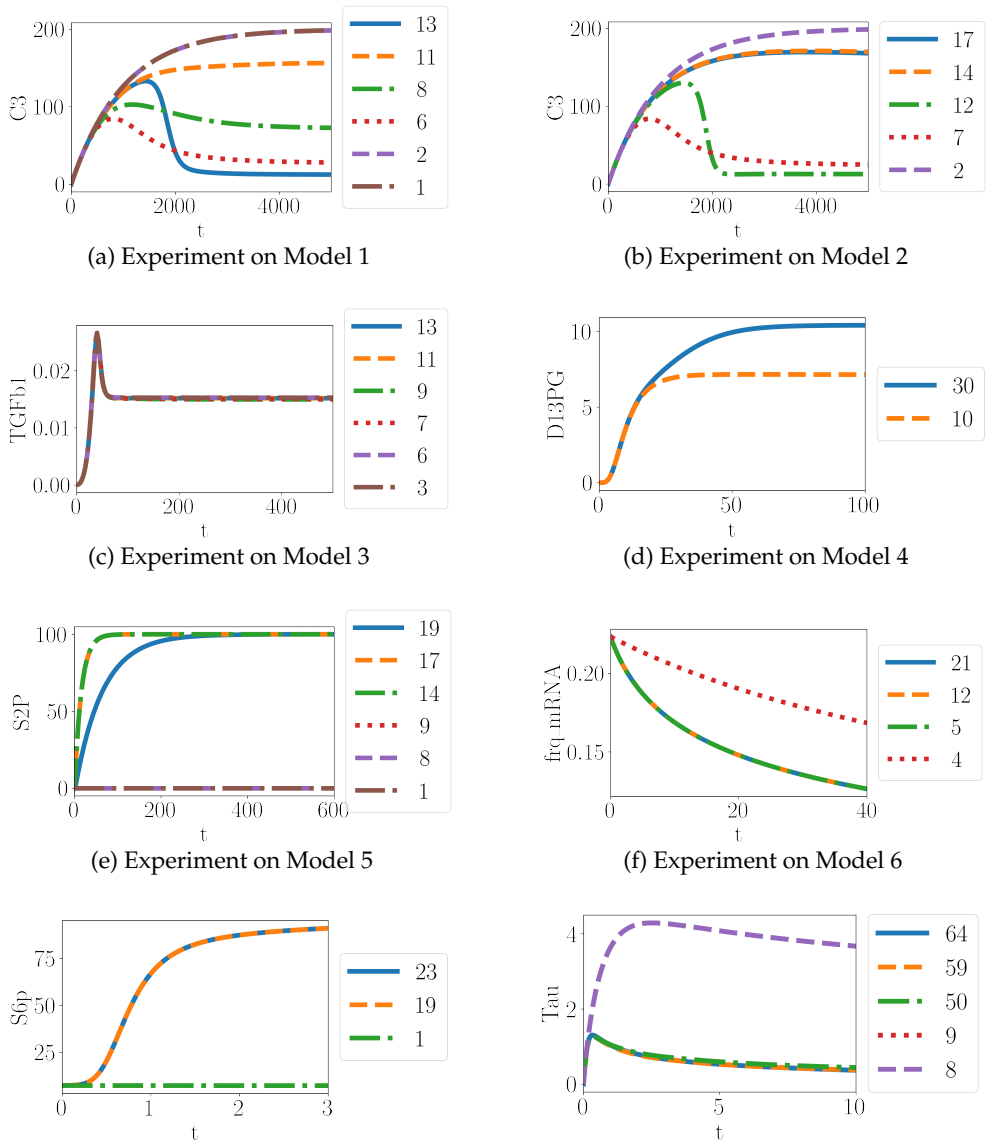
Furthermore, the steady-state error increases with ε for most models. Nonetheless, Figures 8b and 8c show that it is possible for aggressive reductions to incur lower errors. This can be explained by the influence of sampling on the representation of $\mathcal{V}_J$, as the check in Line 6 depends on the norms of each rJ_i. Overall, for given models, significant reductions without large errors were obtained by Algorithm 5 with a cutoff size of 80% of the original model size. In principle, smaller cutoff sizes may be used to find meaningful reductions; however, this requires a case-by-case analysis.

We see that rational models tend to have larger values of $\varepsilon_{\max}$ compared to polynomial ones (see Remark 3.3). This leads to more iterations of Algorithm 5. The values of $\varepsilon/\varepsilon_{\max}$ that lead to low errors vary widely depending on the model. For polynomial models, most reductions happen under the 10% threshold. Instead, for rationals the values change significantly, having reductions for ε lower than 0.01% of $\varepsilon_{\max}$. Following Remark 3.4, choosing a larger value for d in Model 8 would have led to fewer iterations and faster times at the cost of potentially missing some low-error reductions as these happened for low values of ε. Finally, we note that assumptions of Theorem 3.1 were satisfied by the experiments since the obtained approximate lumpings L led to positive $L^T Lx$ for all positive vectors x. At the same time, the denominators of rational $f(x)$ vanished only for negative values of x.

(b) Scalability analysis

We evaluate the scalability of our approach using a parametric model of multisite protein phosphorylation which can be made combinatorially larger by increasing its number of *binding sites* [1]. This mechanism is key in the study of cellular processes [62]. The model describes the phosphorylation and dephosphorylation of a substrate with n different binding sites, each capable of being in 4 different states [38]. The resulting model requires $4n + 2$ variables to track the evolution of all protein configurations and the concentrations of kinase and phosphatase.

The exact constrained lumping of this model was studied in [26]. We choose the concentration of free kinase as the observable of interest. For our setup, we added random noise to each reaction, taken from -5% to $+5\%$ of the parameter value appearing in each reaction (the same parameter could have different values for different reactions). This ensured that the perturbed model was not exactly reducible. Next, we ran Algorithm 5 with a maximum reduction size $m^* = 10$ and $d_{\min} = 10^{-4}$ for models with 2 to 5 binding sites. To compute the empirical error, the time horizon



(a) Experiment on Model 1 (b) Experiment on Model 2

(c) Experiment on Model 3 (d) Experiment on Model 4

(e) Experiment on Model 5 (f) Experiment on Model 6

(g) Experiment on Model 7 (h) Experiment on Model 8

Figure 8: Simulation of exact and approximately lumped models from Table 2, for varying cutoff sizes. The continuous blue line refers to exact constrained lumping.

was set to 2, and the initial conditions were taken from the original paper. All simulations were computed using the `LSODA` stiff solver.

Table 3 presents the results for multisite phosphorylation models. Column *Sites* displays the number of binding sites in the model. The size of the original model and that of the reduced model are shown in the columns *Size* and *Red. Size*, respectively. The number of iterations and the total computation time of Algorithm 5 are displayed in the columns *Iter.* and *Time*, respectively. The columns *Original* and *Red.* contain the time used to compute the simulation using the non-reduced and reduced models, respectively. The absolute and relative error at steady-state are shown in the columns $e(T)$ and $e_{Rel}(T)$. While the maximum absolute error is shown in $e_{\max}$.

Sites	*Size*	*Red. Size*	*Iter.*	*Time (min)*	*Original (s)*	*Red. (s)*	$e(T)$	$e_{Rel}(T)$ (%)	$e_{\max}$
2	24	5	8	0.003	0.116	0.022	0.018	2.60	0.274
3	72	4	8	0.214	0.186	0.020	0.004	1.00	0.236
4	264	4	8	8.024	1.429	0.032	0.001	0.30	0.208
5	1032	4	8	371.683	13.303	0.025	0.001	0.60	0.199

Table 3: Approximate lumpings on multisite phosphorylation.

As seen in Table 3, our approach can find a reduction with only 4 species for a model without exact reductions. These reductions have low errors (less than 3% for all models). This is a promising result for the study of more robust reductions. The runtime depends on the size of the model. Larger models take considerably longer as the number of monomials needed to represent the Jacobian increases combinatorially. This, in turn, gets amplified by the number of iterations necessary to find a lumping with the given size restriction, which depends on the tolerance d for Algorithm 5.

5. Conclusion

We have proposed approximate constrained lumping, a relaxation of the exact reduction technique constrained lumping. This allows for more aggressive reductions at the cost of introducing, in a controlled way, errors in the dynamics of the reduced model. The technique can be applied to dynamical systems with polynomial or rational derivatives, common in computational biology. For a given dynamical system, our algorithm utilizes a numerical tolerance to efficiently compute a reduced system that preserves the evolution of a linear combination of specific variables of interest up to a bounded error. We proved that the error bound is proportional to the numerical threshold. Additionally, we introduced a heuristic to obtain appropriate numerical tolerances based on the model size. We demonstrated effectiveness and scalability by obtaining low-error reductions for a collection of published biological models. Future work will explore robust reductions by considering models with uncertain kinetic parameters.

Acknowledgements. This work was partially supported by the Poul Due Jensen Grant 883901, the Villum Investigator Grant S4OS, and by SMaRT COnSTRUCT (CUP J53C24001460006), FAIR (PE0000013, CUP B53C22003630006) under Italian PNRR funded by NextGenerationEU.

References

1. Salazar C, Höfer T. 2009 Multisite protein phosphorylation — from molecular mechanisms to kinetic models. *FEBS Journal* **276**, 3177–3198.
2. Schmidt H, Madsen M, Danø S, Cedersund G. 2008 Complexity reduction of biochemical rate expressions. *Bioinformatics* **24**, 848–854.
3. Sunnaker M, Cedersund G, Jirstrand M. 2011 A method for zooming of nonlinear models of biochemical systems. *BMC Syst. Biol.* **5**, 140.
4. Apri M, de Gee M, Molenaar J. 2012 Complexity reduction preserving dynamical behavior of biochemical networks. *Journal of Theoretical Biology* **304**, 16–26.
5. Okino M, Mavrovouniotis M. 1998 Simplification of Mathematical Models of Chemical Reaction Systems. *Chemical Reviews* **2**, 391–408.
6. Whitby M, Cardelli L, Kwiatkowska M, Laurenti L, Tribastone M, Tschaikowski M. 2022 PID Control of Biochemical Reaction Networks. *IEEE Trans. Autom. Control.* **67**, 1023–1030.
7. Segel L, Slemrod M. 1989 The quasi-steady-state assumption: A case study in perturbation. *SIAM Review* **31**, 446–477.
8. Radulescu O, Gorban AN, Zinovyev A, Noel V. 2012 Reduction of dynamical biochemical reactions networks in computational biology. *Frontiers in genetics* **3**, 131.
9. Tognazzi S, Tribastone M, Tschaikowski M, Vandin A. 2017 EGAC: A Genetic Algorithm to Compare Chemical Reaction Networks. In *GECCO'17* pp. 833–840.

10. Tschaikowski M, Tribastone M. 2017 Spatial fluid limits for stochastic mobile networks. *Perform. Evaluation* **109**, 52–76.
11. Snowden T, van der Graaf P, Tindall M. 2017 Methods of Model Reduction for Large-Scale Biological Systems: A Survey of Current Methods and Trends. *Bulletin of Mathematical Biology* **79**, 1449–1486.
12. Vallabhajosyula R, Chickarmane V, Sauro H. 2005 Conservation analysis of large biochemical networks. *Bioinformatics* **22**, 346–353.
13. Aoki M. 1967 *Optimization of Stochastic Systems: Topics in Discrete-time Systems*. Acad. Press.
14. Kemeny JG, Snell JL. 1960 *Finite Markov Chains*. Van Nostrand.
15. Li G, Rabitz H. 1991 New approaches to determination of constrained lumping schemes for a reaction system in the whole composition space. *Chemical Engineering Science* **46**, 95–111.
16. Larsen KG, Skou A. 1991 Bisimulation through probabilistic testing. *Inf. Comput.* **94**, 1–28.
17. Cardelli L. 2008 From Processes to ODEs by Chemistry. In Ausiello G, Karhumäki J, Mauri G, Ong L, editors, *Fifth Ifip International Conference On Theor. Comput. Sci. – Tcs 2008*.
18. Wirsing M, Hölzl M, Acciai L, Banti F, Clark A, Fantechi A, Gilmore S, Gnesi S, Gönczy L, Koch N, Lapadula A, Mayer P, Mazzanti F, Pugliese R, Schroeder A, Tiezzi F, Tribastone M, Varró D. 2008 Sensoria Patterns: Augmenting Service Engineering with Formal Analysis, Transformation and Dynamicity. In *Leveraging Applications of Formal Methods, Verification and Validation* pp. 170–190.
19. Hillston J, Tribastone M, Gilmore S. 2011 Stochastic Process Algebras: From Individuals to Populations. *The Computer Journal* **55**, 866–881.
20. Tribastone M. 2014 Behavioral relations in a process algebra for variants. In Gnesi S, Fantechi A, Heymans P, Rubin J, Czarnecki K, Dhungana D, editors, *SPLC* pp. 82–91. ACM.
21. Tschaikowski M, Tribastone M. 2014 Exact fluid lumpability in Markovian process algebra. *Theor. Comput. Sci.* **538**, 140–166.
22. Cardelli L, Pérez-Verona IC, Tribastone M, Tschaikowski M, Vandin A, Waizmann T. 2021 Exact maximal reduction of stochastic reaction networks by species lumping. *Bioinform.* **37**, 2175–2182. (10.1093/bioinformatics/btab081)
23. Cardelli L, Tribastone M, Tschaikowski M, Vandin A. 2019 Symbolic computation of differential equivalences. *Theor. Comput. Sci.* **777**, 132–154.
24. Cardelli L, Tribastone M, Tschaikowski M, Vandin A. 2017 Maximal aggregation of polynomial dynamical systems. *PNAS* **114**, 10029–10034.
25. Cardelli L, Tribastone M, Tschaikowski M, Vandin A. 2016 Comparing Chemical Reaction Networks: A Categorical and Algorithmic Perspective. In *Proceedings of LICS'16* pp. 485–494. ACM. (10.1145/2933575.2935318)
26. Ovchinnikov A, Pérez Verona I, Pogudin G, Tribastone M. 2021 CLUE: exact maximal reduction of kinetic models by constrained lumping of differential equations. *Bioinformatics* **37**, 1732–1738. (10.1093/bioinformatics/btab010)
27. Li G, Rabitz H. 1989 A general analysis of exact lumping in chemical kinetics. *Chemical Engineering Science* **44**, 1413–1430.
28. Borisov NM, Chistopolsky AS, Faeder JR, Kholodenko BN. 2008 Domain-Oriented Reduction of Rule-Based Network Models. *IET systems biology* **2**, 342–351. (10.1049/iet-syb:20070081)
29. Li G, Rabitz H. 1990 A general analysis of approximate lumping in chemical kinetics. *Chemical Engineering Science* **45**, 977–1002. (https://doi.org/10.1016/0009-2509(90)85020-E)
30. Pérez-Verona IC, Tribastone M, Vandin A. 2019 A Large-Scale Assessment of Exact Model Reduction in the BioModels Repository. In *CMSB* , pp. 248–265. Springer.
31. Babtie A, Stumpf M. 2017 How to deal with parameters for whole-cell modelling. *J. of The Royal Society Interface* **14**, 20170237.
32. Barnat J, Benes N, Brim L, Demko M, Hajnal M, Pastva S, Safránek D. 2017 Detecting Attractors in Biological Models with Uncertain Parameters. In *CMSB* vol. 10545*LNCS* pp. 40–56. Springer.
33. Cardelli L, Tribastone M, Tschaikowski M, Vandin A. 2018 Guaranteed Error Bounds on Approximate Model Abstractions Through Reachability Analysis. In *QEST'18* vol. 11024*LNCS* pp. 104–121. Springer. (10.1007/978-3-319-99154-2_7)
34. Cardelli L, Squillace G, Tribastone M, Tschaikowski M, Vandin A. 2023 Formal Lumping of Polynomial Differential Equations through Approximate Equivalences. *JLAMP* **134**, 100876. (10.1016/j.jlamp.2023.100876)
35. Voit EO. 2013 Biochemical Systems Theory: A Review. *ISRN Biomathematics* **2013**, 53.

36. Troják M, Safránek D, Pastva S, Brim L. 2023 Rule-based modelling of biological systems using regulated rewriting. *Biosyst.* **225**, 104843.
37. Jiménez-Pastor A, Jacob JP, Pogudin G. 2022 Exact Linear Reduction for Rational Dynamical Systems. In *CMSB* , pp. 198–216. Springer. (10.1007/978-3-031-15034-0_10)
38. Sneddon M, Faeder J, Emonet T. 2011 Efficient modeling, simulation and coarse-graining of biological complexity with NFsim. *Nature methods* **8**, 177.
39. Leguizamon-Robayo A, Jiménez-Pastor A, Tribastone M, Tschaikowski M, Vandin A. 2023 Approximate Constrained Lumping of Polynomial Differential Equations. In *CMSB* LNCS pp. 106–123 Cham. Springer. (10.1007/978-3-031-42697-1_8)
40. Jiménez-Pastor A, Leguizamon-Robayo A, Tschaikowski M, Vandin A. 2024 Approximate Reductions of Rational Dynamical Systems in CLUE. In Gori R, Milazzo P, Tribastone M, editors, *Computational Methods in Systems Biology* pp. 108–116 Cham. Springer Nature Switzerland.
41. Iacobelli G, Tribastone M. 2013 Lumpability of fluid models with heterogeneous agent types. In *DSN'13* pp. 1–11. ISSN: 2158-3927 (10.1109/DSN.2013.6575346)
42. Tschaikowski M, Tribastone M. 2016 Approximate reduction of heterogeneous nonlinear models with differential hulls. *IEEE TAC*.
43. Bacci G, Bacci G, Larsen KG, Mardare R. 2013 On-the-Fly Exact Computation of Bisimilarity Distances. In Piterman N, Smolka SA, editors, *TACAS* vol. 7795*LNCS* pp. 1–15.
44. Daca P, Henzinger TA, Kretínský J, Petrov T. 2016 Linear Distances between Markov Chains. In Desharnais J, Jagadeesan R, editors, *CONCUR* vol. 59*LIPIcs* pp. 20:1–20:15.
45. Antoulas A. 2005 *Approximation of Large-Scale Dynamical Systems*. Advances in Design and Control. SIAM.
46. Repin D, Petrov T. 2021 Automated deep abstractions for stochastic chemical reaction networks. *Inf. Comput.* **281**, 104788.
47. Cairoli F, Carbone G, Bortolussi L. 2021 Abstraction of Markov Population Dynamics via Generative Adversarial Nets. In Cinquemani E, Paulevé L, editors, *CMSB* vol. 12881 pp. 19–35.
48. Helfrich M, Ceska M, Kretínský J, Marticek S. 2022 Abstraction-Based Segmental Simulation of Chemical Reaction Networks. In Petre I, Paun A, editors, *CMSB* vol. 13447 pp. 41–60.
49. Feret J, Danos V, Krivine J, Harmer R, Fontana W. 2009 Internal coarse-graining of molecular systems. *PNAS* **106**, 6453–6458.
50. Cardelli L, Tribastone M, Tschaikowski M, Vandin A. 2015 Forward and Backward Bisimulations for Chemical Reaction Networks. In *CONCUR* pp. 226–239.
51. Tomlin AS, Li G, Rabitz H, Tóth J. 1997 The Effect of Lumping and Expanding on Kinetic Differential Equations. *SIAM Journal on Applied Mathematics* **57**, 1531–1556. Publisher: Society for Industrial and Applied Mathematics (10.1137/S0036139995293294)
52. Lüders C, Sturm T, Radulescu O. 2022 ODEbase: A Repository of ODE Systems for Systems Biology. *Bioinformatics Advances* **2**. (10.1093/bioadv/vbac027)
53. Legewie S, Blüthgen N, Herzel H. 2006 Mathematical modeling identifies inhibitors of apoptosis as mediators of positive feedback and bistability. *PLoS computational biology* **2**, e120. (10.1371/journal.pcbi.0020120)
54. Blinov ML, Faeder JR, Goldstein B, Hlavacek WS. 2004 BioNetGen: software for rule-based modeling of signal transduction based on the interactions of molecular domains. *Bioinformatics* **20**, 3289–3291.
55. Li C, Donizelli M, Rodriguez N, Dharuri H, Endler L, Chelliah V, Li L, He E, Henry A, Stefan M, Snoep J, Hucka M, Le Novère N, Laibe C. 2010 BioModels Database: An enhanced, curated and annotated resource for published quantitative kinetic models.. *BMC Syst. Biol.* **4**, 92.
56. Cardelli L, Tribastone M, Tschaikowski M, Vandin A. 2017 ERODE: A Tool for the Evaluation and Reduction of Ordinary Differential Equations. In *TACAS* LNCS pp. 310–328. Springer. (10.1007/978-3-662-54580-5_19)
57. Venkatraman L, Chia SM, Narmada BC, White JK, Bhowmick SS, Forbes Dewey C, So PT, Tucker-Kellogg L, Yu H. 2012 Plasmin triggers a switch-like decrease in thrombospondin-dependent activation of TGF-β1. *Biophysical Journal* **103**, 1060–1068. (10.1016/j.bpj.2012.06.050)
58. Mu F, Williams RF, Unkefer CJ, Unkefer PJ, Faeder JR, Hlavacek WS. 2007 Carbon-fate maps for metabolic reactions. *Bioinformatics (Oxford, England)* **23**, 3193–3199. (10.1093/bioinformatics/btm498)
59. Tseng YY, Hunt SM, Heintzen C, Crosthwaite SK, Schwartz JM. 2012 Comprehensive modelling of the Neurospora circadian clock and its temperature compensation. *PLoS computational biology* **8**, e1002437. (10.1371/journal.pcbi.1002437)

60. Brännmark C, Nyman E, Fagerholm S, Bergenholm L, Ekstrand EM, Cedersund G, Strålfors P. 2013 Insulin signaling in type 2 diabetes: experimental and modeling analyses reveal mechanisms of insulin resistance in human adipocytes. *The Journal of Biological Chemistry* **288**, 9867–9880. (10.1074/jbc.M112.432062)
61. Proctor CJ, Boche D, Gray DA, Nicoll JAR. 2013 Investigating interventions in Alzheimer's disease with computer simulation models. *PloS One* **8**, e73631. (10.1371/journal.pone.0073631)
62. Gunawardena J. 2005 Multisite protein phosphorylation makes a good threshold but can be a poor switch. *PNAS* **102**, 14617–14622.
63. Rudin W. 1976 *Principles of Mathematical Analysis*. McGraw-Hill.

Appendix: Proofs

Proposition .1. *Let $\dot{x} = f(x)$ a system of ODEs such that f is analytic on a compact set $\Omega \subseteq \mathbb{R}^m$. Given a $(S, T, \eta)-$lumping L, such that $x(t), \bar{L}Lx(t) \in \Omega$ for all $t \in [0, T]$, the norm of the error $\|e(t)\|_2$ is bounded as follows*

$$\|\dot{e}(t)\|_2 \leq \beta \|e(t)\|_2 + \eta, \ \forall t \in [0, T], \ \forall x(0) \in S.$$

Proof. Note that for any initial condition $x(0) \in S$, we can compute

$$\begin{aligned}
\|\dot{e}(t)\|_2 &= \|\dot{y}(t) - L\dot{x}(t)\|_2 \\
&= \|Lf(\bar{L}y) - Lf(x)\|_2 \\
&= \|Lf(\bar{L}y) - Lf(\bar{L}Lx) + Lf(\bar{L}Lx) - Lf(x)\|_2 \\
&\leq \|Lf(\bar{L}y) - Lf(\bar{L}Lx)\|_2 + \|Lf(\bar{L}Lx) - Lf(x)\|_2
\end{aligned}$$

To bound the first term, recall that f is analytic on Ω. Therefore f is locally Lipschitz in Ω. Then there is a constant C such that

$$\|Lf(\bar{L}y) - Lf(\bar{L}Lx)\|_2 \leq C\|L\|_2\|\bar{L}\|_2\|e(t)\|_2,$$

for all $t \in [0, T]$ and all $x(0) \in S$.

Using the fact that L is an $(S, T, \eta)-$lumping, we get that

$$\|\dot{e}(t)\|_2 \leq C\|L\|_2\|\bar{L}\|_2\|e(t)\|_2 + \eta,$$

for all $t \in [0, T]$ and all $x(0) \in S$. The result follows by setting $\beta = \|L\|_2\|\bar{L}\|_2$. □

Lemma .1. *[41, Lemma 2]. Let f be a continuous scalar function on $[0, T]$ which has a right derivative $D_R f(t)$ such that*

$$D_R f(t) \leq \beta f(t) + \gamma,$$

for all $t \in [0, T)$, where β and γ re constants and $f(0) = 0$. Then

$$f(t) \leq \frac{\gamma}{\beta}\left(e^{\beta t} - 1\right),$$

for all $t \in [0, T]$.

Theorem 3.1. By hypothesis, there exists a compact set $\Omega \subseteq \mathbb{R}^m$ such that the $x(t), \bar{L}Lx \in \Omega$ for all $t \in [0, T]$ and for all $x(0) \in S$. Given that $f(x)$ is analytic in Ω, it is possible to apply Lemma .1 and Proposition .1 to obtain the desired result. □

Theorem 3.2. We first prove the polynomial complexity of Algorithm 4. To this aim, we assume a naive implementation of the matrix operations. We begin by analyzing the complexity of the operations performed in the main loop (Line 4) of Algorithm 4. To find the complexity of computing $\pi_i = rJ_iL^TL$, recall that L is an $p \times m$ matrix. This means that computing L^TL can be done in $\mathcal{O}(pm^2)$, while rJ_i can be computed in $\mathcal{O}(m^2)$. Hence, π_i can be computed in $\mathcal{O}((p+2)m^2)$. Since rJ_i and π_i are of dimensions $1 \times m$, computing $d_i = rJ_i - \pi_i$ has complexity $\mathcal{O}(m)$. Instead, computing $d_i/\|d_i\|_2$ has complexity $\mathcal{O}(4m)$. Overall, we have that the complexity

of the operations inside the main loop (Line 4) of Algorithm 4 is $\mathcal{O}((p+2)m^2)$. As this loop is computed Np times, we have that the total complexity of Algorithm 4 is $\mathcal{O}(Np(p+2)m^2)$. The result follows from the fact that Algorithm 1 can be implemented with a complexity of $\mathcal{O}(m^2A)$, where A is the arithmetic cost of computing an entry of $f(x)$. □

Theorem .1. *[63, Theorem 9.19]. Let $\Omega \in \mathbb{R}^m$ open and convex and let $f \in C^1(\Omega, \mathbb{R}^n)$. If there is a constant $C < \infty$ such that*

$$\|Df(x)\|_2 \leq C \ \forall x \in \Omega$$

then

$$\|f(x_1) - f(x_0)\|_2 \leq C\|x_0 - x_1\|_2$$

Proposition .2. *Let L be the matrix computed via Algorithm 4 with tolerance ε and let $\Omega \subset \mathbb{R}^m$ be an open, bounded and convex set. If $x \in \Omega$ and $\bar{L}Lx \in \Omega$, f is analytic in $\bar{\Omega}$, and $J(x)$ can be written in the form given by Equation 2.3, then it follows that*

$$\|Lf(\bar{L}Lx) - Lf(x)\|_2 \leq C\varepsilon\|\bar{L}Lx - x\|_2, \tag{A 1}$$

where C is a constant.

Proof. Fix $x \in \Omega$, by hypothesis $x^R := \bar{L}Lx \in \Omega$. Let $g : \ker L \to \mathbb{R}^m,\ v \mapsto f(v + x^R)$.

We claim that

$$\|DLg(v)\|_2 \leq C\varepsilon,\ \forall v \in \pi(\Omega), \tag{A 2}$$

where C is a constant, and π is the projection onto $\ker L$.

Note that $DLg(v) = LJ(v + x^R))\iota$, where J is the Jacobian of f and $\iota : \mathbb{R}^l \to \mathbb{R}^m$ is the inclusion given by $v \mapsto (v, 0, \ldots, 0)$.

We compute

$$\|DLg(v)\|_2 = \|LJ(v + x^R)\iota\|_2 = \|LJ(v + x^R)\|_2^{\ker L}, \tag{A 3}$$

where $\|\cdot\|_2^{\ker L}$ is the norm restricted to $\ker L$.

Following Equation (A 3), we have to estimate

$$\|LJ(v + x^R)u\|_2, \forall v \in \pi(\Omega), u \in \ker L \text{ s.t. } \|u\|_2 = 1. \tag{A 4}$$

To this aim, we will first estimate each entry of the vector $LJ(v + x^R)u$. Denote by $e_i \in \mathbb{R}^m$, the vector of zeroes with one in the $i-$th entry, and denote by r_i the $i-$th row of L . Let $p = v + x^R$. Given that $u \in \ker L$, we have that

$$\|e_j LJ(p)u\|_2 = \|r_j J(p)u\|_2 = \|r_j \left(J(p) - J(p)\bar{L}L\right) u\|_2.$$

Using the fact that $J(x) = \sum_{i=1}^k J_i\mu_i(x)$, we have

$$\begin{aligned}\|e_j LJ(p)u\|_2 &= \|r_j J(v + \bar{L}Lx)u\|_2 \\ &= \|r_j \left(\sum_{i=1}^k \left(J_i\mu_i(p) - J_i\mu_i(p)\bar{L}L\right)\right) u\|_2 = \|\sum_{i=1}^k \mu_i(p) \left(r_j J_i - r_j J_i \bar{L}L\right) u\|_2.\end{aligned}$$

Using the triangle inequality, and the Cauchy-Schwartz inequality we have that

$$\begin{aligned}\|e_j LJ(p)u\|_2 &\leq \sum_{i=1}^k \|\mu_i(p) \left(r_j J_i - r_j J_i \bar{L}L\right) u\|_2 \leq \sum_{i=1}^k |\mu_i(p)| \, \|\left(r_j J_i - r_j J_i \bar{L}L\right)\|_2 \|u\|_2 \\ &\leq \sum_{i=1}^k |\mu_i(p)| \, \|\left(r_j J_i - r_j J_i \bar{L}L\right)\|_2 \leq \sum_{i=1}^k |\mu_i(p)| \, \varepsilon,\end{aligned} \tag{A 5}$$

where ε is the tolerance used in the computation of Algorithm 4.

Since f is analytic in $\bar{\Omega}$, we have that its derivative is bounded in Ω. Therefore each μ_i is bounded in Ω and so, by the extreme value theorem, there exists a constant C such that

$$\sup_{i,v\in\pi(\Omega)} \left|\mu_i(v + x^R)\right| = C. \tag{A 6}$$

Combining Equation (A 6) with the previous reasoning we have that

$$\|e_j LJ(p)u\|_2 \leq C\varepsilon. \tag{A 7}$$

Using Equations (A 3) and (A 7), we get that

$$\|DLg(v)\|_2 \leq \sqrt{m}C\varepsilon, \forall v \in \pi(\Omega). \tag{A 8}$$

As π is a linear map, the set $\pi(\Omega)$ is open and convex. Therefore, we can use Equation (A 8) and Theorem .1 to show that

$$\|g(v_0) - g(v_1)\|_2 \leq \sqrt{m}C\varepsilon\|v_0 - v_1\|_2, \forall v_0, v_1 \in \pi(\Omega). \tag{A 9}$$

Notice that $x^R = \bar{L}Lx$ is the projection of x onto the rowspace of L. It follows that $\pi(\bar{L}Lx) = 0$, and so $0 \in \pi(\Omega)$. Moreover since $x \in \Omega$, it follows that $\pi(x) = x - \bar{L}Lx \in \pi(\Omega)$. So we can set $v_0 = 0$ and $v_1 = \pi(x)$ in Equation (A 9) to get that

$$\|g(0) - g(\pi(x))\|_2 \leq \sqrt{m}C\varepsilon\|0 - \pi(x)\|_2$$
$$\|f(\bar{L}Lx) - f(x)\|_2 \leq \sqrt{m}C\varepsilon\|\bar{L}Lx - x\|_2,$$

where there last result follows from the definition of x^R and the fact that $x = x^R + \pi(x)$, as we wanted to prove. □

Proposition .3. *Let $f : \mathbb{R}^m \to \mathbb{R}^m$ be an analytic function on a compact set $\Omega \subset \mathbb{R}^m$. Assume we can write $J(x)$ in the form given by Equation 2.3, where $J_i \in \mathbb{R}^{m\times m}$ and $\{\mu_i(x) : i = 1, \ldots, N\}$ are $\mathbb{R}$-linearly independent and analytic on Ω. Consider $x_1, \ldots, x_N \in \Omega$ such that $\{J(x_1), \ldots, J(x_N)\}$ is a basis of the vector space $\mathcal{V}_J$. Given a matrix $L \in \mathbb{R}^{p\times m}$ if there is an $\varepsilon > 0$ such that for all $i = 1, \ldots, N$ and all rows of r of $LJ(x_i)$*

$$||r - \bar{L}Lr||_2 < \varepsilon,$$

then there exists a constant $C > 0$ such that for all $i = 0, \ldots, N$ and all rows s of LJ_i we have

$$||s - \bar{L}Ls||_2 < C\varepsilon.$$

Proof. It is clear that the matrices J_i generate the space $\mathcal{V}_J$ and they belong to it. At the same time, we have by assumption that $J(x_i)$ is a basis of $\mathcal{V}$. Hence:

$$J_i = c_{i,1}J(x_1) + \ldots + c_{i,N}J(x_N).$$

Multiplying by L to the left, we obtain: $LJ_i = c_{i,1}LJ(x_1) + \ldots + c_{i,N}LJ(x_N)$.

Therefore, all rows of LJ_i are linear combinations of rows of the matrices $LJ(x_1), \ldots, LJ(x_N)$. Let us denote by $s_{i,j}$ and $r_{i,j}$ the j-th row of LJ_i and of $LJ(x_i)$, resp. Then we have: $s_{i,j} = c_{i,1}r_{1,j} + \ldots + c_{i,N}r_{N,j}$, which means that:

$$\begin{aligned} ||s_{i,j} - s_{i,j}\bar{L}L||_2 &= \left|\left|c_{i,1}\left(r_{1,j} - r_{1,j}\bar{L}L\right) + \ldots + c_{i,N}\left(r_{N,j} - r_{N,j}\bar{L}L\right)\right|\right|_2 \\ &\leq \left|\left|c_{i,1}\left(r_{1,j} - r_{1,j}\bar{L}L\right)\right|\right|_2 + \ldots + \left|\left|c_{i,N}\left(r_{N,j} - r_{N,j}\bar{L}L\right)\right|\right|_2 \\ &\leq N\max\{|c_{i,1}|, \ldots, |c_{i,N}|\}\varepsilon \end{aligned}$$

Thus, the statement is true by taking $C = N\max\{|c_{i,j}| : i, j = 1, \ldots, N\}$. □

Lemma .2. *In the context of Proposition .2, if there is a set of $\mathbb{R}^{m\times m}$ matrices $\{J_1, \ldots, J_N\}$ s.t. they span $\mathcal{V}_J$, then $\|f(\bar{L}Lx) - f(x)\|_2 \leq \sqrt{m}C'C\varepsilon\|\bar{L}Lx - x\|_2$, where C is the constant given by Proposition .3 and $C' = \sup_{i,x\in\Omega}|\mu_i(x)|$*

Proof. Recall, the extended Algorithm 4 checks that $\|rJ(x_i) - \bar{L}LrJ(x_i)\|_2 \leq \varepsilon$, for all rows r of L and all $i = 1, \ldots, N$, where $x_1, \ldots, x_N \in \Omega$. By using Proposition .3, we get that there is a constant C such that $\|rJi - \bar{L}LrJ_i\|_2 \leq C\varepsilon$ for all $i = 1, \ldots, N$, where, by hypothesis, we can write $J(x) = \sum_{i=0}^{N} J_i\mu_i(x)$ with $J_i \in \mathbb{R}^{n\times n}$ and $\{\mu_i(x) \,:\, i = 1, \ldots, N\}$ being $\mathbb{R}$-linearly independent and analytic on Ω. We follow the reasoning of the proof of Proposition .2, up to Equation A 5. There by the fact that each $\mu_i(x)$ is bounded, we have that

$$\|e_j LJ(p)u\|_2 \leq \sum_{i=1}^{k} |\mu_i(p)| \, \|\left(r_j J_i - r_j J_i \bar{L}L\right)\|_2 \leq C'C\epsilon,$$

where $C' = \sup_{i,v\in\pi(\Omega)} \left|\mu_i(v + x^R)\right|$. The result follows by continuing the aforementioned reasoning. □

Theorem 3.3. As f is analytic in $\bar{\Omega}$, we can use Lemma .2 for each $t \in [0, T]$ and each solution $x(t)$ with initial conditions in S to get

$$\|Lf(\bar{L}Lx(t)) - Lf(x)\|_2 \leq \sqrt{m}C'C\varepsilon\|\bar{L}Lx(t) - x(t)\|_2, \tag{A 10}$$

for all $t \in [0, T]$.

Since Ω is compact by hypothesis, it is closed and bounded. Therefore, there exists an open ball $B(K/2)$ with center in Ω such that $\Omega \in B(K/2)$. Consequently, we obtain the following bound

$$\|\bar{L}Lx(t) - x(t)\|_2 < K. \tag{A 11}$$

The result follows by combining Equations (A 10) and (A 11)

□

Lemma 3.1. Consider a matrix of observables $M \in \mathbb{R}^{l\times m}$ of rank l with $j-$th row denoted by r_j and a decomposition of the Jacobian $J(x) = \sum_{i=1}^{n} J_i\mu_i(x)$. Let ε_{max} be given by Equation 3.5. It follows that $\|r_j J_i - r_i J_i P_L\|_2 \leq \varepsilon_{max}$, for all $j = 1, \ldots, l$ and $i = 1, \ldots, n$. Using ε_{max} (or any $\varepsilon > \varepsilon_{max}$ as input for Algorithm 4 implies that the condition in Line 6 will be false. Thus, exiting the loop and returning a matrix whose rows are orthonormal rows spanning $\mathrm{rowsp}(M)$.

To show that ε_{max} is the lowest upper bound on ε, assume $\varepsilon^* \leq \varepsilon_{max}$ is such that Algorithm 4 outputs orthonormal rows spanning $\mathrm{rowsp}(M)$. This implies that $\max_{i,j}\|r_j J_i - r_i J_i P_L\|_2 \leq \varepsilon^*$. By the definition of ε_{max}, it follows that $\varepsilon^* = \varepsilon_{max}$, as we wanted to prove. □