



**HAL**  
open science

# Exact hierarchical reductions of dynamical models via linear transformations

Alexander Demin, Elizaveta Demitraki, Gleb Pogudin

► **To cite this version:**

Alexander Demin, Elizaveta Demitraki, Gleb Pogudin. Exact hierarchical reductions of dynamical models via linear transformations. *Communications in Nonlinear Science and Numerical Simulation*, 2024, 131, pp.107816. 10.1016/j.cnsns.2024.107816 . hal-03964231v2

**HAL Id: hal-03964231**

**<https://hal.science/hal-03964231v2>**

Submitted on 7 Jan 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

# Exact hierarchical reductions of dynamical models via linear transformations

Alexander Demin<sup>a,\*</sup>, Elizaveta Demitraki<sup>a</sup>, Gleb Pogudin<sup>b</sup>

<sup>a</sup>*National Research University, Higher School of Economics, Moscow, Russia*

<sup>b</sup>*LIX, CNRS, École Polytechnique, Institute Polytechnique de Paris, Palaiseau, France*

---

## Abstract

Dynamical models described by ordinary differential equations (ODEs) are a fundamental tool in the sciences and engineering. Exact reduction aims at producing a lower-dimensional model in which each macro-variable can be directly related to the original variables, and it is thus a natural step towards the model's formal analysis and mechanistic understanding. We present an algorithm which, given a polynomial ODE model, computes a longest possible chain of exact linear reductions of the model such that each reduction refines the previous one, thus giving a user control of the level of detail preserved by the reduction. This significantly generalizes over the existing approaches which compute only the reduction of the lowest dimension subject to an approach-specific constraint. The algorithm reduces finding exact linear reductions to a question about representations of finite-dimensional algebras. We provide an implementation of the algorithm, demonstrate its performance on a set of benchmarks, and illustrate the applicability via case studies. Our implementation is freely available at <https://github.com/x3042/ExactODEReduction.jl>.

*Keywords:* ordinary differential equations, exact reduction, lumping, dimensionality reduction, matrix algebras

*2000 MSC:* 34C20, 34-04, 16G10

---

## 1. Introduction

Ordinary Differential Equations (ODEs) provide a powerful and expressive language for describing systems evolving in real-time and, thus, are widely used both in the sciences and engineering. This motivates development of formal methods to analyse the structure of models defined using ODEs. One important problem which has been studied actively in the past decade from this angle is *model reduction* [1, 2, 3, 4, 5].

---

\*Corresponding author

*Email addresses:* alexander.demin.eternal@gmail.com (Alexander Demin), egdemitraki@edu.hse.ru (Elizaveta Demitraki), gleb.pogudin@polytechnique.edu (Gleb Pogudin)

In general, model reduction refers to a variety of techniques aiming at replacing the model of interest with a lower-dimensional one which allows to analyze the dynamics of the original model. Traditionally, approximate methods such as, e.g., balanced truncation [6] have been employed. *Exact model reduction* is a complementary approach in which one lowers the dimension of the model without introducing approximation errors. Because of their exactness such reductions are typically connected to the system structure and are, thus, of particular interest in the context of performing formal analysis or deriving mechanistic insights. Classical examples of exact model reductions include reductions using conservation laws or reductions based on symmetries [7, 8, 9]. In principle, exact transformations can be useful to reduce different measures of complexity of a model besides model dimension. For example, exact transformations may be used to convert a model to a linear one [10]. Such transformations are beyond the scope of the present paper.

Algorithms for finding exact dimension reductions may be applicable directly to the ODE systems as, for example, the one proposed in the present paper, or to a domain-specific description of a model (which can be then translated to an ODE system). Rule-based modeling [11, 12] is a prominent example of such concise domain-specific descriptions for which powerful reduction algorithms have been devised [1, 8, 13] and implemented [14]. Compared to the ODE-based algorithms considered in this paper, algorithms that rely on a rule-based representation can deal with very large models but are applicable only to a class of ODE systems and search only for a class of reductions (via so-called fragments [1], see also Remark 1).

In this paper, we will focus on an important class of exact reductions of ODE models, exact linear lumpings, which correspond to finding a self-contained system of differential equations for a set of *macro-variables* such that each macro-variable is a linear combination of the original variables. The case when the macro-variables are sought as sums of the original variables has received significant attention, see e.g. [1, 15, 2, 4]. In particular, **ERODE** software has been developed [3] which efficiently finds the optimal subdivision of the variables into macro-variables. **CLUE** package [5, 16] was a step towards relaxing these restrictions on the macro-variables. Compared to the earlier approaches, the macro-variables found by **CLUE** may involve arbitrary coefficients (not just zeroes and ones as before) and also allow the same variable to appear in several macro-variables at once. Consequently, the dimension of the reduced model in **CLUE** could be significantly lower [5, Table 1]. However, the input of the algorithm consisted not only of a model but also of linear functions in the state variables to be preserved (the observables). Such a set of observables may or may not be available, and guessing it correctly is crucial for finding low-dimensional reductions. In this paper, we aim at taking the best from both worlds: requiring only a model as input (as **ERODE**) and allowing the macro-variables to be any linear combinations of the original variables (as **CLUE**).

The main result of the paper is an algorithm for finding arbitrary exact linear reductions when given only a polynomial ODE model with rational coefficients. Note that the question of finding such an arbitrary linear lumping of the lowest

possible dimension may not be the most meaningful one since any linear first integral yields a reduction of dimension one with constant dynamics. Instead, our algorithm builds a hierarchy of reductions, more precisely it *finds a longest possible chain of lumpings* in which each reduction refines the next one (for details, see Section 2) so that a user can choose the desired level of details to be preserved by reduction by moving along the hierarchy and may find reductions which would likely be missed by ERODE and CLUE (e.g., see Example 3 and section 5.2). Such generality comes with a price: our software is typically slower than CLUE and ERODE.

Our algorithm is based on combining the connection of the linear lumping problem to the problem of finding a common invariant subspace of a set of matrices [17, 5] with the structure theory of finite-dimensional algebras. We use the general framework of existing algorithms over finite [18] and algebraically closed [19, 20] fields and achieve desired efficiency by

- sparsity-aware algorithm for finding a basis of an algebra (Section 3.2);
- exploiting the structure of the input to compute mostly with rational numbers and postponing passing to algebraic number fields as much as possible;
- using sparse linear algebra and modular computation to deal with large matrices and expression swell, respectively.

We implemented our algorithm, and the implementation is publicly available at <https://github.com/x3042/ExactODEReduction.jl>. We evaluate its performance on a set of benchmarks from the BioModels database [21], a large collection of models from life sciences, and demonstrate the produced reduction for two case studies.

The rest of the paper is organized as follows. In Section 2, we give a precise definition of exact linear reduction, and formulate explicitly the algorithmic problem we solve in the paper. Section 3 contains a detailed description of the algorithm and its justification. We describe our implementation and report its performance in Section 4. Section 5 contains the case studies describing the reductions produced by our software.

## 2. Problem statement

In the paper, the transpose of a matrix  $M$  is denoted by  $M^T$ . For a vector  $\mathbf{x} = (x_1, \dots, x_n)$  of indeterminates, by  $\mathbb{C}[\mathbf{x}]$  (resp.,  $\mathbb{R}[\mathbf{x}]$ ,  $\mathbb{Q}[\mathbf{x}]$ ) we will denote the set of polynomials in  $\mathbf{x}$  with complex (resp., real, rational) coefficients.

**Definition 1 (Lumping).** Consider an ODE system

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

in the variables  $\mathbf{x} = (x_1, \dots, x_n)$  with polynomial right-hand side, that is,  $\mathbf{f} = (f_1, \dots, f_n)$  and  $f_1, \dots, f_n \in \mathbb{C}[\mathbf{x}]$ . We say that a linear transformation  $\mathbf{y} = \mathbf{x}L$

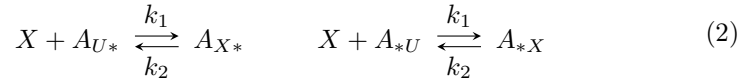
with  $\mathbf{y} = (y_1, \dots, y_m)$ ,  $L \in \mathbb{C}^{n \times m}$ , and  $\text{rank } L = m$  is a *lumping* of (1) if there exists  $\mathbf{g} = (g_1, \dots, g_m)$  with  $g_1, \dots, g_m \in \mathbb{C}[\mathbf{y}]$  such that

$$\mathbf{y}' = \mathbf{g}(\mathbf{y})$$

for every solution  $\mathbf{x}$  of (1). The number  $m$  will be called *the dimension* of the lumping, and the entries of  $\mathbf{y}$  will be referred to as *macro-variables*.

Note that a lumping is uniquely defined by matrix  $L$  only, and the corresponding  $\mathbf{g}$  can be computed, see Remark 3 for details.

Throughout this section, we will work with the following running example [22, Example 1]. The ODE model will be derived using the laws of mass-action kinetics [23, Ch. 7] from a chemical reaction network with a chemical species  $X$  and four more species  $A_{UU}$ ,  $A_{UX}$ ,  $A_{XU}$ , and  $A_{XX}$  which represent a molecule  $A$  having two identical binding sites such that each site may be either unbound (denoted by  $U$ ) or bound to  $X$  (denoted by  $X$ ). These species satisfy the following reactions



where  $* \in \{X, U\}$ , and the reaction rate constants of binding and dissociation are  $k_1$  and  $k_2$ , respectively. If we denote the concentration of any species  $S$  by  $[S]$ , the corresponding ODE system under the law of mass-action kinetics [23, Ch. 7] will be:

$$\begin{cases} [X]' = k_2([A_{XU}] + [A_{UX}] + 2[A_{XX}]) - k_1[X]([A_{XU}] + [A_{UX}] + 2[A_{UU}]), \\ [A_{UU}]' = k_2([A_{XU}] + [A_{UX}]) - 2k_1[X][A_{UU}], \\ [A_{UX}]' = k_2([A_{XX}] - [A_{UX}]) + k_1[X]([A_{UU}] - [A_{UX}]), \\ [A_{XU}]' = k_2([A_{XX}] - [A_{XU}]) + k_1[X]([A_{UU}] - [A_{XU}]), \\ [A_{XX}]' = k_1[X]([A_{XU}] + [A_{UX}]) - 2k_2[A_{XX}]. \end{cases} \quad (3)$$

**Example 1 (Conservation laws as lumpings).** We show that the matrix  $L = (0 \ 1 \ 1 \ 1 \ 1)^T$  yields a lumping of (3). We have

$$y = ([X] \ [A_{UU}] \ [A_{UX}] \ [A_{XU}] \ [A_{XX}]) \cdot L = [A_{UU}] + [A_{UX}] + [A_{XU}] + [A_{XX}].$$

Using (3) one can check that  $y' = 0$ , so we can take  $g(y) = 0$ . Indeed,  $y$  is the total concentration of  $A$  and must be constant. Furthermore, *any linear conservation law* yields a lumping of dimension one.

**Example 2 (More informative lumping).** Another lumping for the same system (3) is given by the matrix

$$L = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \end{pmatrix} \implies \begin{cases} y_1 = [X], \\ y_2 = 2[A_{UU}] + [A_{UX}] + [A_{XU}], \\ y_3 = [A_{UX}] + [A_{XU}] + 2[A_{XX}]. \end{cases}$$

The macro-variables will satisfy a self-contained system

$$y_1' = k_2 y_3 - k_1 y_1 y_2, \quad y_2' = k_2 y_3 - k_1 y_1 y_2, \quad y_3' = -k_2 y_3 + k_1 y_1 y_2.$$

The rationale behind this reduction is that  $y_2$  and  $y_3$  are concentrations of unbound and bound sites, respectively.

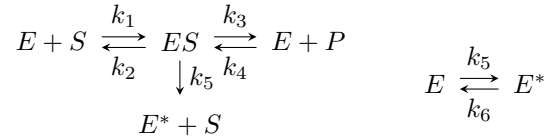
The above examples demonstrate that one system can have several lumpings (in fact, (3) has more), so a natural question is *how to find useful lumpings*. The state-of-the-art software tools CLUE [5] and ERODE [3] approach this question by finding the lumping of the smallest dimension satisfying certain constraints:

- preserving some quantities of interest unlumped (for CLUE [5]);
- or coming from a partition of the state variables (for ERODE [3]).

Both constraints may be too restrictive: not all interesting lumpings come from a partition of the state variables (for instance, Examples 1 and 2; see also [5, Table 1]), and it may be complicated to guess in advance meaningful quantities to preserve.

**Example 3 (Example difficult for CLUE and ERODE, see also Section 5.2).**

Consider another chemical reaction network [24, Eq. (19.20)] (originally due to Daniel Knight):



where we took the rate constants of  $ES \rightarrow E^* + S$  and  $E \rightarrow E^*$  to be equal.

As in the case of (2), we transform the reactions into an ODE system using the law of mass-action kinetics:

$$\begin{cases}
 [E]' = (k_2 + k_3)[ES] + k_6[E^*] - k_1[E][S] - k_4[E][P] - [E], \\
 [S]' = (k_2 + k_5)[ES] - k_1[E][S], \\
 [P]' = k_3[ES] - k_4[E][P], \\
 [ES]' = k_1[E][S] + k_4[E][P] - (k_2 + k_3 + k_5)[ES], \\
 [E^*]' = k_5[E] + k_5[ES] - k_6[E^*]
 \end{cases}$$

One meaningful linear reduction is  $y = [E] + [ES] - \frac{k_6}{k_5}[E^*]$  with the equation  $y' = -(k_5 + k_6)y$ . The macro-variable  $y$  can be understood as a potential between the amount of  $E$  (typically enzyme), both in the free form  $E$  and as a part of the complex  $ES$ , and  $E^*$  (typically inactivated enzyme). There is a bidirectional flow between these two amounts, so any  $k_5$  units of  $[E] + [ES]$  will be in an equilibrium with  $k_6$  units of  $[E^*]$ , and, thus, the dynamics of the difference  $y$  depends only on itself.

This reduction does not come from a subdivision of the species, so it cannot be found by ERODE. Furthermore, finding it using CLUE would require knowing this macro-variable in advance. Since the current state of the software does not allow symbolic parameters to appear in the coefficients of the macro-variables (see Remark 6), we have found this reduction by taking numerical values of  $k_5$  and  $k_6$ .

An alternative approach would be to find *all* the lumpings and let the user choose which ones to use. The problem is that there may be easily an infinite number of lumpings, for example, similarly to Example 1, one can show that the matrix

$$L = (\alpha \quad 1 \quad 1 + \alpha \quad 1 + \alpha \quad 1 + 2\alpha)^T$$

yields a lumping of (3) for every number  $\alpha$ . Furthermore, as we will explain later, the lumpings are in a bijection with the invariant subspaces of certain matrices coming from the Jacobian of  $\mathbf{f}(\mathbf{x})$  and (at least for arbitrary matrices) the invariant subspaces can form an arbitrary algebraic variety [25].

The approach we take in this paper is to *find a sequence of reductions* refining each other with the guarantee that this *sequence is of maximal possible length*.

**Definition 2 (Chain of lumpings).** For an ODE system of the form  $\mathbf{x}' = \mathbf{f}(\mathbf{x})$ , a sequence of linear transformations

$$\mathbf{y}_1 = \mathbf{x}L_1, \mathbf{y}_2 = \mathbf{x}L_2, \dots, \mathbf{y}_\ell = \mathbf{x}L_\ell,$$

where  $L_1 \in \mathbb{C}^{n \times m_1}, \dots, L_\ell \in \mathbb{C}^{n \times m_\ell}$ , is called *a chain of lumpings* if

1.  $0 < m_1 < \dots < m_\ell < n$ ;
2.  $\mathbf{y}_i = \mathbf{x}L_i$  is a lumping of (1) for every  $1 \leq i \leq \ell$ ;
3. for every  $1 < i \leq \ell$ , there exists a matrix  $A_i$  such that  $L_{i-1} = L_i A_i$ .

The latter means that the reductions given by  $L_1, \dots, L_\ell$  refine each other. Such a chain  $(L_1, \dots, L_\ell)$  will be called *maximal* if it is not contained as a subsequence in any longer chain.

We will show (see Corollary 2) that all maximal chains are of the same length, so they are also the longest possible chains. Given a maximal chain of lumpings, a user can “zoom in/out” by going left/right along the chain depending on the desired tradeoff between the size of the reduced model and the amount of information retained. Thus, we can now formally state the main problem studied in this paper.

**Main problem 1.**

**Given** a system  $\mathbf{x}' = \mathbf{f}(\mathbf{x})$  with  $\mathbf{f}$  being a vector of polynomials over  $\mathbb{Q}$ ;

**Compute** a maximal chain of lumpings for the system.

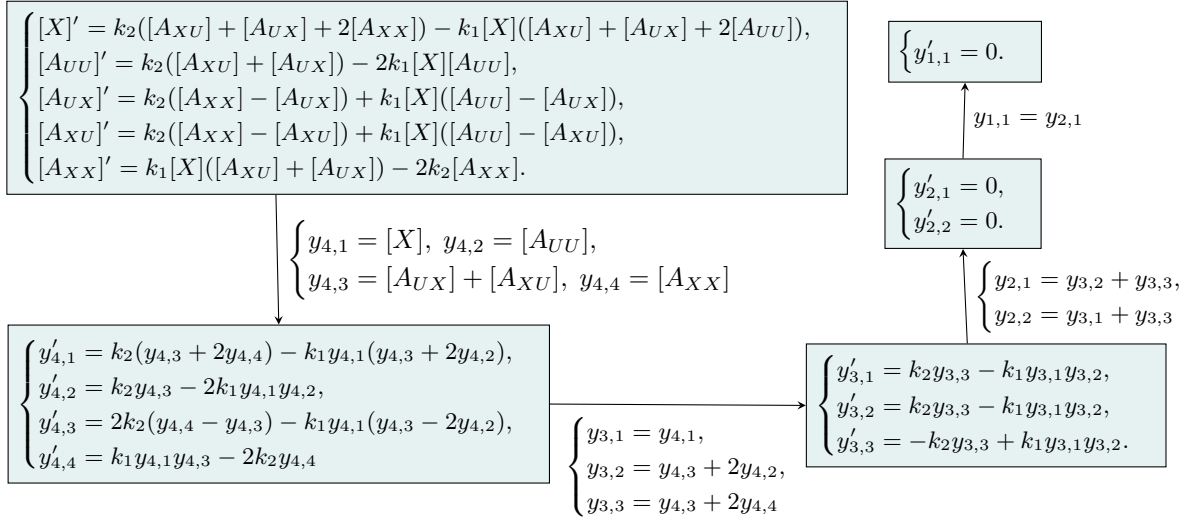


Figure 1: Maximal chain of lumpings for (3) and the corresponding reductions

**Example 4 (Maximal chain of lumpings for (3)).** Figure 1 shows a chain of lumpings and the corresponding reductions for our example system (3). The blocks contain the reduced systems and the arrows are labeled with the transformations between the consecutive reductions (matrices  $A_i$  in the terms of Definition 2). This chain of reductions includes our preceding Examples 1 and 2 as  $y_1$  and  $y_3$ , respectively.

In this example the original dimension  $n = 5$  and the dimensions of the reductions are  $m_1 = 1$ ,  $m_2 = 2$ ,  $m_3 = 3$ ,  $m_4 = 4$ , so this chain is clearly maximal.

**Remark 1 (Connection to symmetries of a rule-based representation).**

The model in (2) fits naturally into the framework of rule-based modeling [11, 12]. The model is given by the rules that stipulate the species dynamics, and the rules themselves are essentially chemical reactions parametrized by different values of  $* \in \{X, U\}$ . Note that these rules are symmetric with respect to the two binding sites of the molecule  $A$ . Thanks to this symmetry, one can find a reduction already at the level of the rules by grouping the species according to the total number of bound sites as follows:

$$y_1 = [X], \quad y_2 = [A_{UU}], \quad y_3 = [A_{UX}] + [A_{XU}], \quad y_4 = [A_{XX}].$$

Note that this is the first reduction in the chain presented on Figure 1. There exist algorithms that exploit such rule-based symmetries [13], and this particular reduction can be found using KaDE software tool [14]<sup>1</sup>. The next reduction

<sup>1</sup>We thank an anonymous referee for raising our awareness of this fact.



in the chain, however, is not discovered by KaDE, although it still admits an interpretation in terms of the structure of the underlying reaction network [22, Example 1 and Section 3.1]. We see an opportunity for synergy here: our structure-agnostic tool may find new types of reductions, and some of these types can be then incorporated in a scalable rule-based approach.

**Remark 2 (Connection to quiver-equivariant ODEs).** One can view a chain of lumpings in the framework of quiver-equivariant dynamical systems [26] with the corresponding quiver being a chain with the maps defined by  $A_i$ 's on the arrows. For this point of view, a natural generalization of the main problem stated above would be to find a maximal (in some sense) quiver  $\mathcal{Q}$  such that the original ODE can be represented as  $\mathcal{Q}$ -equivariant.

### 3. Algorithm

For finding a maximal chain of lumpings, we first use theory developed in [5] to reduce the problem to a problem about common invariant subspaces of a set of matrices (Section 3.1) and then solve the new problem using the structure theory of finite-dimensional algebras (Sections 3.3 to 3.5). The overall algorithm is summarized in Section 3.6.

#### 3.1. Reduction to the search for common invariant subspaces

Let  $\mathbf{x}' = \mathbf{f}(\mathbf{x})$  be an ODE system in variables  $\mathbf{x} = (x_1, \dots, x_n)$  and  $\mathbf{f}$  being a row vector of polynomials  $f_1, \dots, f_n \in \mathbb{C}[\mathbf{x}]$ . Let  $J(\mathbf{x})$  be the Jacobian matrix of  $\mathbf{f}$  with respect to  $\mathbf{x}$ . We denote the monomials in  $\mathbf{x}$  appearing in  $J(\mathbf{x})$  by  $m_1(\mathbf{x}), \dots, m_N(\mathbf{x})$ . Then  $J(\mathbf{x})$  can be written uniquely as

$$J(\mathbf{x}) = (\nabla f_1 \quad \dots \quad \nabla f_n) = \sum_{i=1}^N J_i \cdot m_i(\mathbf{x}), \quad \text{where } \nabla g := \left( \frac{\partial g}{\partial x_1} \quad \dots \quad \frac{\partial g}{\partial x_n} \right)^T \quad (4)$$

and  $J_1, \dots, J_N$  are constant matrices.

**Example 5.** Consider the system

$$x_1' = x_1 - 2x_2^2, \quad x_2' = -x_2 + x_2^2.$$

In this case the decomposition (4) will be

$$J(x_1, x_2) = \begin{pmatrix} 1 & 0 \\ -4x_2 & -1 + 2x_2 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ -4 & 2 \end{pmatrix} x_2.$$

**Lemma 1.** *Using the notation above, the linear transformation  $\mathbf{y} = \mathbf{x}L$ , where  $L \in \mathbb{C}^{n \times m}$ , is a lumping of  $\mathbf{x}' = \mathbf{f}(\mathbf{x})$  if and only if the column space of  $L$  is invariant with respect to  $J_1, \dots, J_N$ .*

*Proof.* For the case  $L \in \mathbb{R}^{n \times m}$ , the statement follows from [5, Lemmas S.I.1 and S.II.1]. The proof of [5, Lemmas S.I.1] remains correct after replacing  $\mathbb{R}$  with  $\mathbb{C}$ , and the proof of [5, Lemmas S.II.1] will be correct for the case of  $\mathbb{C}$  if the real inner products are replaced with the complex ones.  $\square$

**Remark 3.** A natural question is, given  $L$  satisfying the conditions of Lemma 1, how can we find  $\mathbf{g}$  from Definition 1? One approach is the following: we set  $\mathbf{y} := \mathbf{x}L$  and choose a subset  $\tilde{\mathbf{x}}$  of  $\mathbf{x}$  to complete  $\mathbf{y}$  to a basis of the linear span of  $\mathbf{x}$ . Then the derivatives  $\mathbf{y}'$ , which are equal to  $\mathbf{f}L$ , can be written in terms of  $\mathbf{y}$  and  $\tilde{\mathbf{x}}$  via a linear change of coordinates. Lemma 1 guarantees that these polynomials will not, in fact, depend on  $\tilde{\mathbf{x}}$  and, thus, will be exactly  $\mathbf{g}$ . An optimized version of this construction was used already in CLUE [5], and is used in our implementation as well.

**Corollary 1.** A sequence of linear transformations  $\mathbf{y}_1 = \mathbf{x}L_1, \dots, \mathbf{y}_\ell = \mathbf{x}L_\ell$  is a chain of lumpings if and only if the column spaces  $V_1, V_2, \dots, V_\ell$  of  $L_1, \dots, L_\ell$  satisfy

- $V_i$  is invariant with respect to  $J_1, \dots, J_N$  for every  $1 \leq i \leq \ell$ ;
- $\{0\} \subsetneq V_1 \subsetneq \dots \subsetneq V_\ell \subsetneq \mathbb{C}^n$ .

Furthermore, the chain of lumpings is maximal if and only if the chain  $V_1, \dots, V_\ell$  is not a subsequence of a chain of subspaces satisfying the two properties above.

In order to search for such chains of invariant subspaces, we will use theory of finite dimensional matrix algebras.

**Definition 3 (Matrix algebra).** Let  $k$  be a field (e.g.,  $k = \mathbb{Q}, \mathbb{R}, \mathbb{C}$ ). A subspace  $\mathcal{A} \subseteq k^{n \times n}$  of matrices is called *an algebra* if it is closed under multiplication and contains the identity matrix.

For a finite set  $A_1, \dots, A_m \in k^{n \times n}$ , we denote the smallest algebra containing  $A_1, \dots, A_m$  by  $\langle A_1, \dots, A_m \rangle$ . This algebra is equal to the span of all possible products of these matrices.

For an ODE system  $\mathbf{x}' = \mathbf{f}(\mathbf{x})$  with  $\mathbf{x} = (x_1, \dots, x_n)$  and  $f_1, f_2, \dots, f_n \in \mathbb{C}[\mathbf{x}]$ , we consider the coefficients  $J_1, \dots, J_N$  of the Jacobian matrix of  $\mathbf{f}(\mathbf{x})$  written as a polynomial in  $\mathbf{x}$  as in (4). We will call the algebra  $\langle I_n, J_1, \dots, J_N \rangle$  (where  $I_n$  is the identity  $n \times n$ -matrix) *the Jacobian algebra* of the system  $\mathbf{x}' = \mathbf{f}(\mathbf{x})$ .

**Example 6.**

- Let  $T_n$  be the set of all upper-triangular matrices in  $k^{n \times n}$ . Since the product of two upper-triangular matrices is upper-triangular again,  $T_n$  is an algebra.
- Consider the system from Example 5. Its Jacobian algebra is

$$\left\langle \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \begin{pmatrix} 0 & 0 \\ -4 & 2 \end{pmatrix} \right\rangle = \{M^T \mid M \in T_2\}.$$

Since a subspace  $V \subset \mathbb{C}^n$  is invariant with respect to  $J_1, \dots, J_N$  if and only if it is invariant with respect to  $\langle I_n, J_1, \dots, J_N \rangle$ , that is, invariant w.r.t. any element of the algebra, we will further focus on finding invariant subspaces of this Jacobian algebra. An immediate benefit is that we can use the Jordan-Hölder theorem [27, Theorem 1.5.1] to clarify our notion of the maximal chain of lumpings (Definition 2): the definition only requires that a maximal chain cannot be further refined, and this, in general, does not preclude the existence of longer chains. The following direct consequence of [27, Theorem 1.5.1] guarantees that a maximal chain indeed has the maximal possible length.

**Corollary 2.** For a given ODE system  $\mathbf{x}' = \mathbf{f}(\mathbf{x})$ , all maximal chains of lumpings have the same length.

### 3.2. Generating the algebra

For performing explicit computations with the Jacobian algebra  $\langle I_n, J_1, \dots, J_N \rangle$  (Definition 3), we will compute its basis. Algorithm 1 gives a simplified version of our approach, which is essentially [5, Algorithm 2] applied to matrices instead of vectors. Similarly to [5], we employ modular computation (cf. [5, Algorithm 3]) to avoid the intermediate expression swell and use sparse linear algebra.

Building upon this straightforward adaptation of the approach from [5], we significantly improve the performance by taking further advantage of the sparsity of the input and output. In the models from literature, the nonlinear part of the model is typically sparse and, as a result, most of  $J_1, \dots, J_N$  are extremely sparse; furthermore, the basis of the Jacobian algebra also can be often chosen to be very sparse. Hence, many of the matrices  $C$  from (Step 2)(b)i will be sparse as well. However, some of the matrices computed at the intermediate steps may be still quite dense slowing down the whole algorithm. We deal with the issue by temporarily deferring (Step 2)(b)ii for relatively dense matrices  $C$  and then, once the outer loop exits signaling that  $P$  is empty, we add each of the deferred matrices to  $P$  and restart the iteration. This way we ensure that we have generated enough sparse matrices in the algebra so that the reductions of the dense matrices will be more sparse now. Thanks to this optimization, Algorithm 1 is not a bottleneck in our computation which it was when we used the approach from [5] directly.

---

**Algorithm 1** Finding a basis of matrix algebra (basic version)

---

**Input** a set of square matrices  $A_1, \dots, A_\ell \in k^{n \times n}$ ;

**Output** a basis  $S$  of the smallest linear subspace  $\mathcal{A} \subseteq k^{n \times n}$  containing all possible products of  $A_1, \dots, A_\ell$ ;

**(Step 1)** Set  $S$  to be any basis of the linear span of  $A_1, \dots, A_\ell$  and let  $P := S$ .

**(Step 2)** While  $P \neq \emptyset$  do

- (a) Take  $B$  to be an element of  $P$  and remove it from  $P$ .
- (b) For every  $A$  in  $\{A_1, \dots, A_\ell\}$  do
  - i. Compute  $C := AB$  and reduce  $C$  w.r.t.  $S$  via Gaussian reduction.
  - ii. If  $C \neq 0$ , set  $S := S \cup \{C\}$  and  $P := P \cup \{C\}$ .

**(Step 3)** Return  $S$ .

---

### 3.3. Search for invariant subspaces: algebraic preliminaries

For this section, we fix a ground field  $k$  of characteristic zero. The cases we are mostly interested in are rational numbers  $\mathbb{Q}$ , algebraic numbers  $\overline{\mathbb{Q}}$ , and complex numbers  $\mathbb{C}$ .

**Definition 4 (Radical of an algebra).** Let  $\mathcal{A} \subseteq k^{n \times n}$  be an algebra.

- A subspace  $\mathcal{I} \subseteq \mathcal{A}$  is called *an ideal* (resp., *left ideal*) if  $AB, BA \in \mathcal{I}$  (resp.,  $AB \in \mathcal{I}$ ) for every  $A \in \mathcal{A}$  and  $B \in \mathcal{I}$ .
- An ideal (resp., left ideal)  $\mathcal{I} \subseteq \mathcal{A}$  is *nilpotent* if there exists  $N$  such that the product of any  $N$  elements of  $\mathcal{I}$  is zero.
- The set of all elements  $A \in \mathcal{A}$  such that the left ideal  $\mathcal{A} \cdot A$  is nilpotent is called *the radical* of  $\mathcal{A}$ . It is a nilpotent ideal of  $\mathcal{A}$  by [27, Theorems 3.1.6, 3.1.10].

**Example 7.** Let  $T_n$  be the set of all upper-triangular matrices in  $k^{n \times n}$ . Consider a subset  $U_n \subset T_n$  of strictly upper-triangular matrices. One can easily verify that  $U_n$  is an ideal and the product of any  $n$  elements of  $U_n$  is zero. Since, for every  $A \in U_n$ , we have  $T_n \cdot A \subseteq U_n$ , we deduce that  $U_n$  is the radical of  $T_n$ .

Dixon's theorem [28, Theorem 11] implies that the radical of an algebra  $\mathcal{A} \subseteq k^{n \times n}$  can be computed by finding the kernel of a square matrix of order  $\dim \mathcal{A} \leq n^2$ . The relevance of the notion of radical to our problem is demonstrated by the following lemma.

**Lemma 2.** Let  $\mathcal{A} \subseteq k^{n \times n}$  be an algebra, and let  $\mathcal{R} \subset \mathcal{A}$  be its radical. If  $\mathcal{R} \neq \{0\}$ , then the intersection  $\bigcap_{R \in \mathcal{R}} \text{Ker } R$  is nonzero and is invariant w.r.t.  $\mathcal{A}$ .

*Proof.* Since  $\mathcal{R}$  is a nilpotent ideal, there exists the smallest integer  $N$  such that the product of any  $N$  elements of  $\mathcal{R}$  is zero. Then, there exists  $0 \neq M \in k^{n \times n}$  which is a product of  $N - 1$  elements of  $\mathcal{R}$ . Hence, we have  $RM = 0$  for every  $R \in \mathcal{R}$ , so  $V := \bigcap_{R \in \mathcal{R}} \text{Ker } R \supseteq \text{Im } M$  is nontrivial.

Consider  $v \in V$ ,  $A \in \mathcal{A}$ , and  $R \in \mathcal{R}$ . Since  $RA \in \mathcal{R}$ , we have  $RAv = 0$ , so  $Av \in \text{Ker } R$ . Thus,  $V$  is invariant w.r.t.  $\mathcal{A}$ .  $\square$

**Example 8.** Consider the system from Example 5. In Example 6, it was shown that the Jacobian algebra of this system is the set of lower triangular matrices. Similarly to Example 7 we find that the radical of this algebra is  $\begin{pmatrix} 0 & 0 \\ \lambda & 0 \end{pmatrix}$ . The common kernel of the radical is spanned by the second basis vector yielding the reduction  $y' = -y + y^2$  (with  $y = x_2$ ).

**Definition 5 (Semisimple algebra).** An algebra  $\mathcal{A} \subseteq k^{n \times n}$  is called *semisimple* if its radical is zero.

We will use the following characterization of semisimple algebras.

**Theorem 3** (Wedderburn-Artin, [27, Theorems 2.4.3 and 2.6.2]). Let  $\mathcal{A} \subseteq k^{n \times n}$  be a semisimple algebra. Then there exist

1. algebras  $\mathcal{A}_1 \subseteq k^{n_1 \times n_1}, \dots, \mathcal{A}_\ell \subseteq k^{n_\ell \times n_\ell}$  such that  $\mathcal{A}_i$  does not have a nontrivial proper invariant subspace in  $k^{n_i}$  for every  $1 \leq i \leq \ell$ ,
2. integers  $m_1, \dots, m_\ell$  such that  $n_1 m_1 + \dots + n_\ell m_\ell = n$ ,
3. a basis in  $k^n$

such that, in this basis, we have

$$\mathcal{A} = \left\{ \text{Diag}(\underbrace{A_1, \dots, A_1}_{m_1 \text{ times}}, \dots, \underbrace{A_\ell, \dots, A_\ell}_{m_\ell \text{ times}}) \mid A_1 \in \mathcal{A}_1, \dots, A_\ell \in \mathcal{A}_\ell \right\}, \quad (5)$$

where  $\text{Diag}(B_1, \dots, B_N)$  denotes the block-diagonal matrix with blocks  $B_1, \dots, B_N$ .

**Example 9.** Consider the set of all matrices of the form

$$\begin{pmatrix} a & b & 0 & 0 \\ -b & a & 0 & 0 \\ 0 & 0 & c & 0 \\ 0 & 0 & 0 & c \end{pmatrix}, \quad \text{where } a, b, c \in \mathbb{Q}.$$

This is a semisimple algebra in the form (5) with  $\ell = 2$ ,  $m_1 = 1$ , and  $m_2 = 2$ .

In the case  $\ell = 1$  and  $m_1 = 1$  in the decomposition (5) from Theorem 3, there are no invariant subspaces in  $k^n$  but there still may be invariant subspaces in  $\overline{k}^n$  if  $k \neq \overline{k}$ , where  $\overline{k}$  is the algebraic closure of field  $k$ . These subspaces can be found using the center of the algebra.

**Definition 6 (Center/Centralizer).** Let  $\mathcal{A} \subseteq k^{n \times n}$  be an algebra.

- The *center* of  $\mathcal{A}$  is the set of all  $M \in \mathcal{A}$  such that  $MA = AM$  for every  $A \in \mathcal{A}$ .
- The *centralizer* of  $\mathcal{A}$  is the set of all  $M \in k^{n \times n}$  such that  $MA = AM$  for every  $A \in \mathcal{A}$ .

Since, for every fixed  $A$ ,  $AM = MA$  is a system of linear equations in the entries, the center and centralizer can be computed by solving a system of linear equations.

**Lemma 4.** Let  $\mathcal{A} \subseteq k^{n \times n}$  be a subalgebra. Let  $\mathcal{C}$  be the centralizer of  $\mathcal{A}$ . For every  $C \in \mathcal{C}$ , every eigenspace of  $C$  is an invariant subspace of  $\mathcal{A}$  in  $\overline{k}^n$ .

*Proof.* Let  $V$  be an eigenspace of  $C$  corresponding to the eigenvalue  $\lambda$ . Let  $A \in \mathcal{A}$ . Then, for  $v \in V$ , we have  $C(Av) = (CA)v = (AC)v = \lambda Av$ , so  $Av$  belongs to  $V$  as well.  $\square$

**Lemma 5.** Let  $\mathcal{A} \subseteq \mathbb{Q}^{n \times n}$  be a semisimple algebra. Let  $M \in \mathcal{A}$  be a matrix such that the characteristic polynomial of  $M$  is of the form  $p(t)^d$ , where  $p(t)$  is  $\mathbb{Q}$ -irreducible. Let  $\mathcal{Z}$  and  $\mathcal{C}$  be the center and centralizer of  $\mathcal{A}$ , respectively. Then the equality  $\dim \mathcal{C} = d^2 \dim \mathcal{Z}$  is equivalent to the fact that, in the Wedderburn-Artin decomposition (5) of  $\mathcal{A}$ , we have  $\ell = 1$  and  $m_1 = d$ .

*Proof.* We consider the Wedderburn-Artin decomposition (5) of  $\mathcal{A}$ . For every  $1 \leq i \leq \ell$ , we denote the center of  $\mathcal{A}_i$  by  $\mathcal{Z}_i$ . Then  $\dim \mathcal{Z} = \dim \mathcal{Z}_1 + \dots + \dim \mathcal{Z}_\ell$ . The number of irreducible factors of a characteristic polynomial of any element of  $\mathcal{A}$  will be at least  $m_1 + \dots + m_\ell$ , so  $d \geq m_1 + \dots + m_\ell$ . A direct computation using the Schur's lemma [27, Theorem 2.1.1] implies that the centralizer  $\mathcal{C}$  of  $\mathcal{A}$  is isomorphic to  $\text{Mat}_{m_1}(\mathcal{Z}_1) \times \dots \times \text{Mat}_{m_\ell}(\mathcal{Z}_\ell)$ , where  $\text{Mat}_{m_i}(\mathcal{Z}_i)$  denotes the space of  $m_i \times m_i$ -block matrices with each block being an element of  $\mathcal{Z}_i$  (cf. [27, Theorem 2.6.4]). Therefore

$$\dim \mathcal{C} = m_1^2 \dim \mathcal{Z}_1 + m_2^2 \dim \mathcal{Z}_2 + \dots + m_\ell^2 \dim \mathcal{Z}_\ell.$$

Bounding the right-hand side, we can write

$$\dim \mathcal{C} \leq (m_1 + \dots + m_\ell)^2 (\dim \mathcal{Z}_1 + \dots + \dim \mathcal{Z}_\ell) \leq d^2 \dim \mathcal{Z}$$

Both inequalities will be equalities if and only if  $\ell = 1$  and  $d = m_1$ , and this proves the lemma.  $\square$

---

**Algorithm 2** Finding a nontrivial invariant subspace of an algebra

---

**Input** a basis  $B_1, \dots, B_N \in \mathbb{Q}^{n \times n}$  of an algebra  $\mathcal{A} \subseteq \mathbb{Q}^{n \times n}$ ;

**Output** One of the following:

- NO if there is no subspace in  $\overline{\mathbb{Q}}^n$  invariant w.r.t.  $\mathcal{A}$ ;
- nontrivial proper subspace in  $\overline{\mathbb{Q}}^n$  invariant w.r.t.  $\mathcal{A}$ ;
- a maximal chain of subspaces in  $\overline{\mathbb{Q}}^n$  invariant w.r.t.  $\mathcal{A}$ .

*Considering corner cases:*

**(Step 1)** If  $N = n^2$ , return NO.

**(Step 2)** For an arbitrary nonzero vector  $v$ , consider the space  $V$  spanned by  $B_1v, \dots, B_Nv$ . If  $\dim V < n$ , **return**  $V$ .

*Examining the radical:*

**(Step 3)** Find a basis of the radical  $\mathcal{R}$  of  $\mathcal{A}$  (Definition 4) using Dixon's theorem [28, Theorem 11].

**(Step 4)** If  $\dim \mathcal{R} > 0$  compute the common kernel of the basis elements of  $\mathcal{R}$  and **return** it (see Lemma 2).

*Semisimple case:*

**(Step 5)** Set  $D := 1$ .

**(Step 6)** Compute  $M := \sum_{i=1}^N a_i B_i$ , where  $a_1, \dots, a_N$  are sampled independently and uniformly at random from  $\{1, 2, \dots, D\}$ .

**(Step 7)** If the characteristic polynomial of  $M$  has at least two distinct  $\mathbb{Q}$ -irreducible factors (say,  $p_1(t)$  and  $p_2(t)$ ):

- (a) Check the invariance of  $\text{Ker } p_1(M)$  w.r.t.  $B_1, \dots, B_N$ .
- (b) If it is invariant, **return**  $\text{Ker } p_1(M)$ . Otherwise, set  $D := 2D$  and **go to** (Step 6).

**(Step 8)** Write the characteristic polynomial of  $M$  as  $p(t)^d$ , where  $p(t)$  is  $\mathbb{Q}$ -irreducible.

**(Step 9)** Compute the center  $\mathcal{Z}$  and centralizer  $\mathcal{C}$  of  $\mathcal{A}$  (Definition 6).

**(Step 10)** If  $\dim \mathcal{C} < d^2 \dim \mathcal{Z}$ , set  $D := 2D$  and **go to** (Step 6).

**(Step 11)** Let  $C_1, \dots, C_s$  be a basis of  $\mathcal{C}$ . Set  $C := \sum_{i=1}^s b_i C_i$ , where  $b_1, \dots, b_s$  are sampled independently and uniformly at random from  $\{1, 2, \dots, D\}$ .

**(Step 12)** Compute  $q(t)$ , the minimal polynomial of  $C$ . If  $q$  is  $\mathbb{Q}$ -reducible or  $\deg q < d \dim \mathcal{Z}$ , set  $D := 2D$  and **go to** (Step 6).

**(Step 13)** Let  $V_1, \dots, V_\ell$  14  
(where  $\ell = d \dim \mathcal{Z}$ ) be the eigenspaces of  $C$ .

**(Step 14)** **Return**  $V_1 \subset V_1 \oplus V_2 \subset \dots \subset V_1 \oplus V_2 \oplus \dots \oplus V_{\ell-1}$ .

---

### 3.4. Search for invariant subspaces: how to find one

In this subsection, we present Algorithm 2 for finding an invariant subspace if there is any. The rest of the subsection is devoted to justifying its correctness and termination, see Proposition 2.

**Proposition 1.** *Let  $\mathcal{A} \subseteq \mathbb{Q}^{n \times n}$  be a semisimple algebra such that there are no nontrivial proper  $\mathcal{A}$ -invariant subspaces in  $\mathbb{Q}^n$ . Let  $M_1, \dots, M_N$  be a basis of  $\mathcal{A}$  as a vector space. Then the polynomial*

$$\det(x_1 M_1 + \dots + x_N M_N) \in \mathbb{Q}[x_1, \dots, x_N]$$

*is of the form  $P^d$ , where  $P$  is irreducible over  $\mathbb{Q}$ .*

**Remark 4 (On the importance of being a basis).** While the statement of Proposition 1 may sound quite natural, the situation is in fact quite subtle: if one replaces linear basis with a set of generators of  $\mathcal{A}$  in the statement of the proposition, it will not longer be true [29, Theorem 1.2 and Subsection 2.1].

*Proof of Proposition 1.* By performing a change of coordinates over  $\mathbb{Q}$ , we will assume that  $M_1$  is the identity matrix.

Let  $\overline{\mathcal{A}}$  be the complexification of  $\mathcal{A}$ . By the Wedderburn-Artin theorem [27, Corollary 2.4.4], there exist  $n_1, \dots, n_\ell$  such that  $N = n_1^2 + \dots + n_\ell^2$  and

$$\overline{\mathcal{A}} \cong \text{Mat}_{n_1}(\mathbb{C}) \times \dots \times \text{Mat}_{n_\ell}(\mathbb{C}). \quad (6)$$

Then the complexification  $\mathbb{C}^n$  of the original representation  $\mathbb{Q}^n$  of  $\mathcal{A}$  can be decomposed [27, Theorem 2.6.2] as

$$\mathbb{C}^n = k_1 V_1 \oplus k_2 V_2 \oplus \dots \oplus k_\ell V_\ell, \quad (7)$$

where  $V_i \cong \mathbb{C}^{n_i}$  is the unique irreducible representation of  $\text{Mat}_{n_i}(\mathbb{C})$ . We denote the base change corresponding to (7) by  $C \in \mathbb{C}^{n \times n}$ . Then  $CMC^{-1}$ , where  $M := x_1 M_1 + \dots + x_N M_N$ , is block diagonal with the dimensions of blocks as in (7). Since  $M_1, \dots, M_N$  span the whole  $\overline{\mathcal{A}}$ , the distinct nonzero entries of  $CMC^{-1}$  are  $\mathbb{C}$ -linearly independent linear forms in  $x_1, \dots, x_N$ . By denoting these forms by  $y_1, \dots, y_N$  we obtain an invertible matrix  $B \in \mathbb{C}^{N \times N}$  such that  $\mathbf{y} := B\mathbf{x}$ , and, reordering  $y_1, \dots, y_N$  if necessary, one has

$$CMC^{-1} = \text{diag}(\underbrace{Y_1, \dots, Y_1}_{k_1 \text{ times}}, \dots, \underbrace{Y_\ell, \dots, Y_\ell}_{k_\ell \text{ times}}),$$

where  $Y_i$  is a matrix with entries  $y_{n_1 + \dots + n_{i-1}^2 + 1}, \dots, y_{n_1^2 + \dots + n_i^2}$  for  $1 \leq i \leq \ell$ .

Then we have

$$\det(M) = \det(CMC^{-1}) = \det(Y_1)^{k_1} \dots \det(Y_\ell)^{k_\ell}.$$

Furthermore, since  $M_1$  is the identity,  $\det(M)|_{x_1=x_1+t}$  as a polynomial in  $t$  is the characteristic polynomial of  $-M$ . Let  $Q(\mathbf{x}) := \det Y_1 \dots \det Y_\ell \in \mathbb{Q}[\mathbf{x}]$ . Then  $Q|_{x_1=x_1+t}$  as a polynomial in  $t$  is the minimal polynomial of  $-M$ .



Since  $\det Y_i$  is a determinant of a matrix with independent entries, it is irreducible over  $\mathbb{C}$ . Let  $p(\mathbf{x})$  be a  $\mathbb{Q}$ -irreducible divisor of  $\det M$ . Then  $p$  divides  $Q$ , so, by reordering  $Y_i$ 's if necessary, we can assume that  $p(\mathbf{x}) = \det Y_1 \dots \det Y_r$  for  $r \leq \ell$ . Assume that  $r < \ell$ . Set  $p_0(t) := p(x_1 - t, x_2, \dots, x_N)$  and consider  $p_0(M)$ . We will have

$$Cp_0(M)C^{-1} = \text{diag}(\underbrace{0, \dots, 0}_{k_1 + \dots + k_r \text{ times}}, \underbrace{p_0(Y_{r+1}), \dots, p_0(Y_{r+1})}_{k_{r+1} \text{ times}}, \dots, \underbrace{p_0(Y_{r+1}), \dots, p_0(Y_{r+1})}_{k_\ell \text{ times}}).$$

Since  $p_0$  is coprime with the characteristic polynomials of  $Y_{r+1}, \dots, Y_\ell$ , each of the matrices  $p_0(Y_{r+1}), \dots, p_0(Y_\ell)$  is nonsingular. Therefore, the kernel of  $Cp_0(M)C^{-1}$  is exactly the span of the first  $k_1 + \dots + k_r$  basis vectors. Therefore, the kernel of  $p_0(M)$  is the span of this many first columns of  $C^{-1}$ . Therefore, the kernel of  $p_0(M)$  is  $\overline{\mathcal{A}}$ -invariant and is defined over  $\mathbb{C}$ . On the other hand, the entries of  $p_0(M)$  belong to  $\mathbb{Q}(\mathbf{x})$ , so the kernel of  $p_0(M)$  in fact is defined over  $\mathbb{C} \cap \mathbb{Q}(\mathbf{x}) = \mathbb{Q}$ . Therefore, the kernel of  $p_0(M)$  yields a nontrivial  $\mathcal{A}$ -invariant subspace of  $\mathbb{Q}^n$  contradicting with the irreducibility of this representation. Hence  $p$  must be equal to  $Q$ , so  $\det M$  must be a power of  $p$ .  $\square$

The proof of the proposition provides a way to find the degree of  $\deg P$ .

**Corollary 3.** In the notation of the proof (see (6)) of Proposition 1,  $\deg P = n_1 + n_2 + \dots + n_\ell$ .

**Proposition 2.** *Algorithm 2 is correct and terminates with probability one.*

**Remark 5 (On the probability of termination).** By “terminates with probability one” we mean that the algorithm makes a random choice in an infinite probability space, and will terminate for all the choices except for a set of probability zero. Simply put, the algorithm repeatedly tosses a coin and will terminate as long as there will be at least one heads which will eventually happen with probability one.

*Proof of Proposition 2.* We will first prove the *correctness*. If the algorithm returned on (Step 1), then  $\mathcal{A}$  is the full matrix algebra, and does not have any nontrivial proper invariant subspace. If the algorithm returned on (Step 2), then the returned subspace is invariant by construction. If the algorithm returned on (Step 4), the returned subspace is nonzero, proper (since if  $\mathcal{A}$  was a zero algebra, the algorithm would have returned at (Step 2)), and invariant due to Lemma 2.

It remains to consider the case when the algorithm returns after (Step 5). If the algorithm returned on (Step 7)(b), then the returned subspace is invariant by construction and is nonzero because  $p_1(t)$  divides the charpoly of  $M$ , so  $p_1(M)$  is a singular matrix. Finally, consider the case when the algorithm returned on (Step 14). Consider the decomposition (5) from Theorem 3 for  $\mathcal{A}$ . If the algorithm reached (Step 11), it contains a matrix with the charpoly being

$p(t)^d$  with  $\mathbb{Q}$ -irreducible  $p(t)$  such that  $\dim \mathcal{C} = d^2 \dim \mathcal{Z}$ . Lemma 5 implies that, in the decomposition (5), we have  $\ell = 1$  and  $m_1 = d$ . Thus, the whole space  $\mathbb{Q}^n$  can be written as  $U_1 \oplus U_2 \oplus \dots \oplus U_d$  such that each of  $U_i$ 's is  $\mathcal{A}$ -invariant without proper nontrivial  $\mathcal{A}$ -invariant subspaces. [27, Corollary 2.2.4] implies that, over  $\overline{\mathbb{Q}}$ , each of  $U_i$ 's can be decomposed as a direct sum of at most  $\dim \mathcal{Z}$   $\mathcal{A}$ -invariant subspaces. Therefore, the whole  $\overline{\mathbb{Q}}^n$  can be decomposed into at most  $d \dim \mathcal{Z}$   $\mathcal{A}$ -invariant subspaces by [27, Theorem 2.6.2].

Lemma 4 implies that each of  $V_i$ 's from (Step 13) is an invariant subspace w.r.t.  $\mathcal{A}$ . Since there are  $d \dim \mathcal{Z}$  of them, each of  $V_i$ 's does not contain nontrivial proper  $\mathcal{A}$ -invariant subspaces. Therefore, the chain  $V_1 \subset V_1 \oplus V_2 \subset \dots \subset V_1 \oplus V_2 \oplus \dots \oplus V_{\ell-1}$  returned at (Step 14) is maximal. This finished the proof of the *correctness* of the algorithm.

We will now prove that the algorithm *terminates* with probability one. Consider the decomposition (5) of  $\mathcal{A}$  from Theorem 3. Consider variables  $z_1, \dots, z_N$  and a matrix  $M_0 := \sum_{i=1}^N z_i B_i$ . Then  $M$  at (Step 6) is a specialization of  $M_0$  at  $z_i = a_i$ . Let  $P(z_1, \dots, z_N, t)$  be the charpoly of  $M_0$ . Consider the decomposition (5) for  $\mathcal{A}$ . For every  $1 \leq i \leq \ell$ , we apply Proposition 1 to the block corresponding to  $\mathcal{A}_i$  and obtain a  $\mathbb{Q}$ -irreducible  $P_i$  and its power  $d_i$ . Thus, we obtain the following factorization for  $M_0$

$$P = P_1^{d_1 m_1} P_2^{d_2 m_2} \dots P_\ell^{d_\ell m_\ell}.$$

The characteristic polynomial of  $M$  computed at (Step 6) equals  $P(a_1, \dots, a_N, t)$ . Assume that  $P_i(a_1, \dots, a_N, t)$  is  $\mathbb{Q}$ -reducible for every  $1 \leq i \leq s$  and these polynomials are distinct.

- Assume that  $\ell > 1$ . Then  $p_1(t)$  from (Step 7) will be equal to  $P_i(a_1, \dots, a_N, t)$  for some  $i$ . Then  $\text{Ker } p_1(M)$  will be the subspace corresponding to the  $i$ -th block in the decomposition (5). The subspace is invariant, so it will be returned on (Step 7)(b).
- Assume that  $\ell = 1$ . We will study matrix  $C$  similarly to the way we studied  $M$  above. Let  $y_1, \dots, y_s$  be independent variables, and we define  $C_0 := y_1 C_1 + \dots + y_s C_s$ . By the same argument as in the proof of Lemma 5, we have  $\mathcal{C} \cong \text{Mat}_r(\mathcal{Z})$  for some integer  $r$ . By [27, Proposition 2.3.4], algebra  $\mathcal{C}$  is simple and every  $\mathcal{C}$ -module (in particular, our ambient space  $\mathbb{Q}^n$ ) is a direct sum of isomorphic copies of the same  $\mathcal{C}$ -module. We apply Proposition 1 to this module and deduce that the characteristic polynomial of  $C_0$  is of the form  $Q(y_1, \dots, y_s, t)^h$  for some integer  $h$  and  $\mathbb{Q}$ -irreducible polynomial  $Q$ . Furthermore,  $\deg Q = d \dim \mathcal{Z}$  by Corollary 3. Assume that  $Q(b_1, \dots, b_s, t)$  is  $\mathbb{Q}$ -irreducible. Then  $Q(b_1, \dots, b_s, t)$  will be the minimal polynomial of  $C$ , so this polynomial will not satisfy the condition of (Step 12) and, thus, the algorithm will terminate without going back to (Step 6).

Combining the two underlined assumptions in the text above, we see that the algorithm will return for a fixed value of  $D$  if the following conditions hold:

1.  $P_i(a_1, \dots, a_N, t)$  is  $\mathbb{Q}$ -reducible for every  $1 \leq i \leq s$  and these polynomials are all distinct;
2.  $Q(b_1, \dots, b_s, t)$  is  $\mathbb{Q}$ -irreducible.

[30, Theorem 2.1] implies that there exists constants  $C_0, C_1$  such that the probability of any of  $P_i(a_1, \dots, a_N, t)$ 's and  $Q(b_1, \dots, b_s, t)$  being  $\mathbb{Q}$ -reducible is less than  $\frac{C_1}{\sqrt[3]{D}}$  if  $D > C_0$ . Furthermore, the Schwartz-Zippel lemma [31, Proposition 98] implies that there exists a constant  $C_2$  such that the probability of any of  $P_i(a_1, \dots, a_N, t)$ 's being equal does not exceed  $\frac{C_2}{D}$ . Therefore, for  $D > C_0$ , the probability that  $D$  will be updated is at most  $\frac{C_1}{\sqrt[3]{D}} + \frac{C_2}{D}$ . This number will eventually become less than 0.99, so the probability of non-termination will be bounded by  $0.99 \cdot 0.99 \cdot \dots = 0$ .  $\square$

### 3.5. Search for invariant subspaces: how to find a chain

In this section, we describe how to use Algorithm 2 in a recursive manner to find a maximal chain of invariant subspaces in  $\mathbb{Q}$  w.r.t. the Jacobian algebra  $\mathcal{A} \subset \mathbb{Q}^{n \times n}$  of an ODE system. We will denote a basis of  $\mathcal{A}$  by  $B_1, \dots, B_N$ .

In the cases when Algorithm 2 applied to  $B_1, \dots, B_N$  returned NO or a maximal chain of invariant subspaces, we are done. Therefore, we consider the case when Algorithm 2 returns a single invariant subspace  $V \subset \mathbb{Q}^n$ . In this case, we consider two subproblems:

1. *Restriction.* Since  $V$  is invariant w.r.t.  $B_1, \dots, B_N$ , there are well-defined restrictions  $B_1|_V, \dots, B_N|_V$ . We fix a basis in  $V$  and will denote the matrix representations for these restricted operators also by  $B_1^*, \dots, B_N^*$ .
2. *Quotients.* Consider the quotient space  $\mathbb{Q}^n/V$  and the quotient map  $\pi: \mathbb{Q}^n \rightarrow \mathbb{Q}^n/V$  (see [32, 3.83, 3.88]). Since  $V$  is invariant w.r.t.  $B_1, \dots, B_N$ , we can consider the quotient operators  $B_1/V, \dots, B_N/V$  (see [32, 5.14]), we denote their matrix representations by  $B_1^\circ, \dots, B_N^\circ$ . Note that, for every their common invariant subspace  $U \subset \mathbb{Q}^n/V$ , the subspace  $\pi^{-1}(U) \subset \mathbb{Q}^n$  is invariant w.r.t.  $B_1, \dots, B_N$ .

Note that the aforementioned matrix representations can be computed solving linear systems in  $n$  variables. Thus, we can work recursively with algebras  $\langle B_1^*, \dots, B_N^* \rangle$  on  $V$  and  $\langle B_1^\circ, \dots, B_N^\circ \rangle$  on  $\mathbb{Q}^n/V$ . If the resulting maximal chains of invariant subspaces are

$$0 \subsetneq V_1 \subsetneq \dots \subsetneq V_s \subsetneq V \quad \text{and} \quad 0 \subsetneq U_1 \subsetneq \dots \subsetneq U_r \subsetneq \mathbb{Q}^n/V,$$

then we can return the following maximal chain of invariant subspaces for  $B_1, \dots, B_N$

$$0 \subsetneq V_1 \subsetneq \dots \subsetneq V_s \subsetneq V \subsetneq \pi^{-1}(U_1) \subsetneq \dots \subsetneq \pi^{-1}(U_r) \subsetneq \mathbb{Q}^n.$$

### 3.6. Putting everything together

In this section we collect the subroutines from the preceding sections into the complete algorithm for finding a maximal chain of lumpings.

---

**Algorithm 3** Finding a maximal chain of lumpings

---

**Input** ODE system  $\mathbf{x}' = \mathbf{f}(\mathbf{x})$  with  $\mathbf{x} = (x_1, \dots, x_n)$ ,  $\mathbf{f} = (f_1, \dots, f_n) \in \mathbb{Q}[\mathbf{x}]^n$ ;

**Output** a maximal chain of lumpings (see Definition 2 and Example 4);

**(Step 1)** Compute the Jacobian  $J(\mathbf{x})$  of  $\mathbf{f}$  and the matrices  $J_1, \dots, J_\ell \in \mathbb{Q}^{n \times n}$  from its decomposition as in (4).

**(Step 2)** Use Algorithm 1 to compute the basis  $B_1, \dots, B_N$  of the Jacobian algebra  $\mathcal{A} = \langle I_n, J_1, \dots, J_\ell \rangle$  of the system.

**(Step 3)** Apply Algorithm 2 in a recursive way as described in Section 3.5 to compute a maximal chain  $V_1 \subsetneq \dots \subsetneq V_s$  of subspaces in  $\overline{\mathbb{Q}}^n$  invariant w.r.t.  $\mathcal{A}$ .

**(Step 4)** For each  $1 \leq i \leq s$ , find a matrix  $L_i$  with the columns being a basis of  $V_i$ .

**(Step 5) Return**  $L_1, \dots, L_s$ .

---

## 4. Implementation and performance

We have implemented Algorithm 3 (and all the algorithms it relies on) in Julia language [33] as a part of `ExactODEReduction.jl` package. The package together with relevant resources to replicate our results is freely available at

<https://github.com/x3042/ExactODEReduction.jl>

We use libraries `AbstractAlgebra.jl` and `Nemo.jl` [34]. Internally, this results in using `FLINT` [35] and `Calcium` [36] (for complex number arithmetic). We use a version of the code from [22] to improve interpretability of the computed lumpings. Additionally, during the development stage, various components of the package were profiled on collections of sparse matrices from the SuiteSparse dataset [37]. Our implementation accepts models typed manually or from the files in the `ERODE *.ode` format [3, Section 3.2]. We provide documentation, installation instructions, and usage examples.

**Remark 6 (Encoding scalar parameters).** Models in the literature often involve scalar parameters. Our algorithm transforms each such parameter  $k$  into a state  $k(t)$  satisfying equations  $k'(t) = 0$  (same transformation is used in `ERODE` under the name “currying” [38, p. 15]). This way, one may also find reductions in the parameters space, not only in the state space. Another approach to

handle the parameters could be to adjoin them to the field of coefficients (as allowed in CLUE) thus allowing parameters in the coefficients of a reduction. This feature is not implemented at the moment but most of the presented theory and algorithms can be reused for this case.

We will demonstrate the performance of our implementation on a set of benchmarks<sup>2</sup>. We use benchmarks from the BioModels database [21] collected in [39] of dimensions ranging from 4 to 133. We run Algorithm 3 over rationals on each of the models. Table 1 contains benchmark results aggregated by models’ dimension. For each range, we report:

- the number of models considered;
- the (average) length of a chain of reductions found;
- the (average) number of nonequivalent reductions, where equivalence is taken up to adding states with constant dynamics. We have chosen to report this because we think is it a reasonable first approximation to the number of interesting reductions;
- the (minimum, average, maximum) elapsed runtime of our implementation;

| Models info |          | Reductions |                  | Runtime (sec.) |          |          |
|-------------|----------|------------|------------------|----------------|----------|----------|
| Dimension   | # Models | # Total    | # Non-equivalent | Min.           | Average  | Max.     |
| 2 - 9       | 44       | 4.02       | 1.39             | 0.0 s          | 0.6 s    | 0.66 s   |
| 10 - 19     | 41       | 8.15       | 2.61             | 0.01 s         | 0.21 s   | 1.46 s   |
| 20 - 29     | 46       | 9.65       | 2.13             | 0.08 s         | 0.44 s   | 1.48 s   |
| 30 - 39     | 17       | 19.41      | 2.71             | 0.33 s         | 1.74 s   | 5.91 s   |
| 40 - 59     | 25       | 29.08      | 6.08             | 0.78 s         | 4.58 s   | 26.71 s  |
| 60 - 79     | 20       | 37.25      | 6.95             | 7.7 s          | 34.57 s  | 102.92 s |
| 80 - 99     | 11       | 42.91      | 7.09             | 24.46 s        | 96.38 s  | 497.26 s |
| 100 - 133   | 4        | 89.0       | 21.5             | 75.15 s        | 202.52 s | 312.02 s |

Table 1: Benchmark results aggregated by model dimension

The timings were produced on a laptop with 2 cores 1.60GHz each and 8 Gb RAM<sup>3</sup>. We would like to note that out of the 208 models considered, at least one reduction was found in 202 models, and 154 of them admit a non-constant reduction.

The timings in the table do not include the cost of the positivization step [22], which is optional. Here, our algorithm uses the Polymake [40] library. With the

<sup>2</sup>Models are available at <https://github.com/x3042/ExactODEReduction.jl/tree/main/data/ODEs>, commit hash 678d32c5bbc8beedc9e22b673238cde0ec673a46.

<sup>3</sup>For the overall table, we refer to [https://github.com/x3042/ExactODEReduction.jl/blob/main/benchmark/biomodels\\_benchmark](https://github.com/x3042/ExactODEReduction.jl/blob/main/benchmark/biomodels_benchmark), commit hash 23c9f532aa316cbef59a8e3e6be04156a3d9c3eb.

positivization step, the running time increases no more than by a factor of two in most instances, and usually the increase is indistinguishable at all<sup>4</sup>. In the earlier versions of the implementation of Algorithm 3, computing the algebra basis on (Step 2) had often been a clear bottleneck on our dataset. With the modifications to the Algorithm 1 as described in Section 3.2, currently, the most time-consuming steps are the restriction and quotienting procedures applied on (Step 3) of Algorithm 3. Solving a number of linear systems to find the matrix representations of restricted and quotient operators is a clear bottleneck here.

## 5. Case studies

### 5.1. Inactivation of factor Va

We will consider a model from [41] which appears in the BioModels database [21] as BIOMD0000000365. Factor V is a protein involved in the process of coagulation (transforming blood from liquid to gel), and thus is closely related to blood vessel repair and thrombosis. In particular, it can assist in activating anticoagulant protein C. The activated factor V, factor Va, can no longer do this. A model describing deactivation of Va by means of activated protein C (APC) was proposed and studied in [41].

Factor Va consists of the heavy chain (HC) and light chain (LC), and the binding of APC happens through the light chain. The model consists of the following species

- Factor Va and its versions Va<sub>3</sub>, Va<sub>5</sub>, Va<sub>6</sub>, Va<sub>53</sub>, Va<sub>56</sub>, Va<sub>36</sub>, and Va<sub>536</sub>;
- LC, HC, and the versions of the latter (HC<sub>3</sub>, HC<sub>5</sub>, etc) corresponding to the versions of Va;
- the A<sub>1</sub> domain of factor Va, Va<sub>LC·A1</sub> and versions of the A<sub>2</sub> domain such as Va<sub>A3</sub>, Va<sub>A53</sub>, etc.
- APC, complexes formed by it and LC/Va (such as APC·Va<sub>3</sub>).

In total, the model contains 30 variables and 9 parameters, and the parameters are encoded as constant states as in Remark 6. Our code finds a maximal chain of lumpings of length 14 in under 5 second on a laptop. The smallest reduction with nonzero dynamics has dimension three and involves two parameters (similar to the one in Example 4):

$$\begin{cases} y'_1 = -k_1 y_1 y_2 + k_2 y_3, \\ y'_2 = -k_1 y_1 y_2 + k_2 y_3, \\ y'_3 = k_1 y_1 y_2 - k_2 y_3. \end{cases} \quad (8)$$

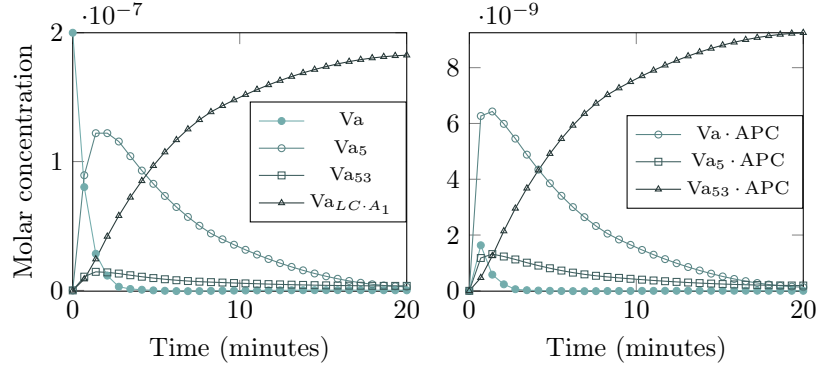
---

<sup>4</sup>One notable exception are models that admit large reductions with large coefficients. For example, model BIOMD0000000153 of dimension 76 has 22 nontrivial reductions of dimensions 55 and more, and applying the positivization routine increases the total runtime from 40 s to 1240 s.

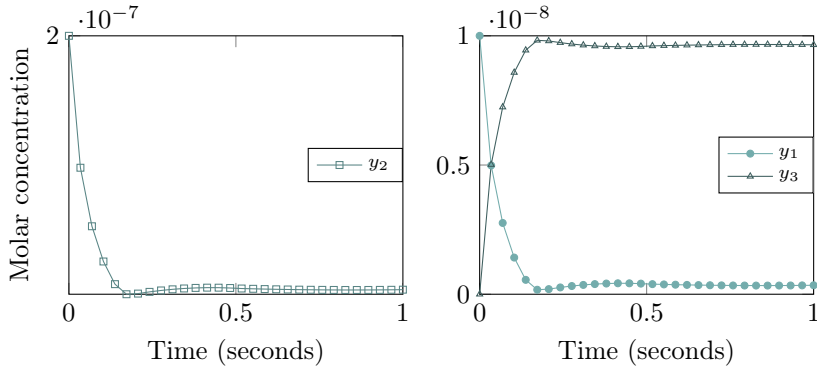
The macro-variables are

$$\begin{aligned}
 y_1 &= [\text{APC}], \\
 y_2 &= [\text{LC}] + [\text{Va}] + [\text{Va}_3] + [\text{Va}_{36}] + [\text{Va}_5] + [\text{Va}_{53}] + [\text{Va}_{536}] + [\text{Va}_{56}] + [\text{Va}_{\text{LC}\cdot A_1}], \\
 y_3 &= [\text{LC}\cdot \text{APC}] + [\text{Va}\cdot \text{APC}] + [\text{Va}_3\cdot \text{APC}] + [\text{Va}_{36}\cdot \text{APC}] + [\text{Va}_5\cdot \text{APC}] \\
 &\quad + [\text{Va}_{53}\cdot \text{APC}] + [\text{Va}_{536}\cdot \text{APC}] + [\text{Va}_{56}\cdot \text{APC}] + [\text{Va}_{\text{LC}\cdot A_1}\cdot \text{APC}].
 \end{aligned}$$

Variable  $y_2$  (resp.,  $y_3$ ) can be described as the total concentration of the light chains without (resp., with) bound APC. Therefore, the reduction (8) focuses on the process of binding/unbinding of APC to the light chains, and it turns out that the other processes such as reactions between the heavy and light chains become irrelevant and, in particular, the  $\text{HC}_n$  species do not appear in the macro-variables at all.



(a) Some of the states of the original model appearing in  $y_2$  and  $y_3$



(b) States of the reduced model

Figure 2: Numerical simulation for the model from [41] and its reduction using the initial conditions and parameter values from [41]

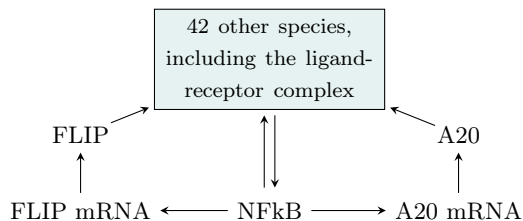


Figure 3: The relevant chemical species and dependencies between them

From a numerical perspective<sup>5</sup>, the reduction (8) can be interpreted as exact timescale separation since the dynamics of the macro-variables turns out to be transient compared to the dynamics of the original system. More precisely, the original system was studied in [41] and has nontrivial dynamics on the timespan of 1200 second. In particular, this is the case for the variables contributing to the macro-variable  $y_2$ , see Figure 2a. On the other hand, as Figure 2b shows, the macro-variables  $y_1, y_2, y_3$  have much faster dynamics and reach the steady state after less than one second.

### 5.2. Model of cell death

In this subsection, we consider a model designed in [44] in order to study the sensitivity of the apoptosis (programmed cell death) to the TNF (tumor necrosis factor) stimulation. The overall model involves 47 chemical species and numerous interactions between them schematically described in [44, Figure 1]. Our code produces a maximal chain of lumpings of length 23 (16 out of them with nonconstant dynamics).

We will consider the nonconstant reduction of the smallest dimension. It involves two proteins, A20 and FLIP, whose concentrations depend on the concentrations of the corresponding mRNAs, A20\_mRNA and FLIP\_mRNA. The concentrations of these mRNAs are governed by the concentrations of nuclear NF- $\kappa$ B (NF $\kappa$ B\_N). The latter depends (directly or indirectly) on many other species including the aforementioned protein A20.

These species and relations between them are summarized on Figure 3, and the corresponding differential equations are:

$$\begin{aligned} [A20]' &= k_1[A20\_mRNA] + k_2, & [A20\_mRNA]' &= k_5[NF\kappa B\_N], \\ [FLIP]' &= k_3[FLIP\_mRNA] + k_4, & [FLIP\_mRNA]' &= k_6[NF\kappa B\_N], \end{aligned}$$

where  $k_1, \dots, k_6$  are numeric parameters. Our code finds a three-dimensional reduction which can be straightforwardly simplified further a two-dimensional

---

<sup>5</sup>All numerical simulations in this paper have been done using `ModelingToolkit` [42] and `DifferentialEquations.jl` [43]



with the following macro-variables  $y_1, y_2$  and the reduced system:

$$\begin{cases} y_1 = \frac{k_6}{k_1}[A20] - \frac{k_5}{k_3}[FLIP], \\ y_2 = k_6[A20\_mRNA] - k_5[FLIP\_mRNA] \end{cases} \implies \begin{cases} y_1' = y_2 + \frac{k_2 k_6}{k_1} - \frac{k_4 k_5}{k_3}, \\ y_2' = 0 \end{cases}$$

So the idea is that, although both A20 and FLIP are involved in a complex reaction network, one can, by eliminating the dependence on NF $\kappa$ B, find a linear combination of them satisfying a simple system of differential equations which can be explicitly solved. Such explicit relations on the states can be, for example, combined with the differential inequalities method in order to obtain tighter reachability bounds [45].

By going further along the chain of the reductions one can include gradually more species into the reduced model, for example, a combination of the RIP protein and the transitional receptor can be included in a similar fashion.

## 6. Conclusions

We have presented a new algorithm which takes as input a system of ODEs and produces a longest possible chain of exact linear reductions of the system such that each reduction in the chain is a refinement of the previous one. This specification is more general compared to the existing tools as it does not put any restriction on the new variables other than being the linear combinations of the original ones and it does not require any initial observable/guess. We expect that our approach can be extended to the systems with the rational right-hand side using the ideas from [16], we leave this for future research.

We provided a publicly available implementation in Julia. Our code is able to analyze models of dimension over a hundred in a couple of minutes using commodity hardware. We have also demonstrated its applicability to models arising in life sciences. The performance can be further improved, for example, by first searching for linear first integrals (which appear frequently, for example, in chemical reaction networks) and using these constant reductions to skip some of the steps of our algorithm.

Since the produced reductions are exact, our tool can be used for formal verification and as a preprocessing for approximate reduction techniques. While exactness is thus an important feature, it can also be viewed as a limitation since some models have only a few exact reductions (if any). Therefore, one intriguing direction for future research is to produce a relaxed version of our algorithm to find approximate lumpings together with rigorous error bounds. For existing results on approximate lumping and related approaches based on the singular perturbation theory, see [46, 47, 48] and references therein. Interestingly, the core linear algebraic problem of our algorithm, finding common invariant subspaces, has been recently studied from the perspective of approximate but rigorous computation in [49, 50] motivated by factoring linear differential operators. We expect the ideas from these papers to be useful in our context as well.

## Acknowledgments

We would like to thank David E. Speyer for his clear and detailed note [20]. We would like to thank Mirco Tribastone for helpful discussions and Rongwei Yang for discussions about Proposition 1. We would like to thank the referees for helpful feedback. GP was supported by the Paris Ile-de-France region (project “XOR”) and partially supported by NSF grants DMS-1853482, DMS-1760448, and DMS-1853650. AD was supported by the Max Planck Institute for Informatics.

## References

- [1] J. Feret, V. Danos, J. Krivine, R. Harmer, W. Fontana, [Internal coarse-graining of molecular systems](#), Proceedings of the National Academy of Sciences 106 (16) (2009) 6453–6458.  
URL <http://dx.doi.org/10.1073/pnas.0809908106>
- [2] J. Feret, T. Henzinger, H. Koepl, T. Petrov, [Lumpability abstractions of rule-based systems](#), Theoretical Computer Science 431 (2012) 137–164.  
URL <https://doi.org/10.1016/j.tcs.2011.12.059>
- [3] L. Cardelli, M. Tribastone, M. Tschaikowski, A. Vandin, [ERODE: A tool for the evaluation and reduction of ordinary differential equations](#), in: TACAS 2017, Vol. 10206 of LNCS, 2017.  
URL [https://doi.org/10.1007/978-3-662-54580-5\\_19](https://doi.org/10.1007/978-3-662-54580-5_19)
- [4] L. Cardelli, M. Tribastone, M. Tschaikowski, A. Vandin, [Maximal aggregation of polynomial dynamical systems](#), Proceedings of the National Academy of Sciences 114 (38) (2017) 10029–10034.  
URL <https://www.pnas.org/content/114/38/10029>
- [5] A. Ovchinnikov, I. P. Verona, G. Pogudin, M. Tribastone, [CLUE: exact maximal reduction of kinetic models by constrained lumping of differential equations](#), Bioinformatics (2021).  
URL <https://doi.org/10.1093/bioinformatics/btab010>
- [6] A. Antoulas, Approximation of Large-Scale Dynamical Systems, Adv. in Design and Control, SIAM, 2005.
- [7] E. Hubert, G. Labahn, [Scaling invariants and symmetry reduction of dynamical systems](#), Foundations of Computational Mathematics 13 (4) (2013) 479–516.  
URL <https://doi.org/10.1007/s10208-013-9165-9>
- [8] F. Camporesi, J. Feret, [Formal reduction for rule-based models](#), Electronic Notes in Theoretical Computer Science 276 (2011) 29–59.  
URL <https://doi.org/10.1016/j.entcs.2011.09.014>

- [9] P. J. Olver, [Equivalence, Invariants and Symmetry](#), Cambridge University Press, 1995.  
URL <https://doi.org/10.1017/cbo9780511609565>
- [10] S. Sankaranarayanan, [Automatic abstraction of non-linear systems using change of bases transformations](#) in: Proceedings of the 14th international conference on Hybrid systems: computation and control, 2011.  
URL <https://doi.org/10.1145/1967701.1967723>
- [11] J. R. Faeder, M. L. Blinov, W. S. Hlavacek, [Graphical rule-based representation of signal-transduction networks](#), in: Proceedings of the 2005 ACM symposium on Applied computing, 2005.  
URL <https://doi.org/10.1145/1066677.1066712>
- [12] V. Danos, J. Feret, W. Fontana, R. Harmer, J. Krivine, [Rule-based modelling of cellular signalling](#), in: CONCUR 2007 – Concurrency Theory, 2007, pp. 17–41.  
URL [https://doi.org/10.1007/978-3-540-74407-8\\_3](https://doi.org/10.1007/978-3-540-74407-8_3)
- [13] F. Camporesi, J. Feret, J. Hayman, [Context-sensitive flow analyses: A hierarchy of model reductions](#), in: Computational Methods in Systems Biology, 2013, pp. 220–233.  
URL [https://doi.org/10.1007/978-3-642-40708-6\\_17](https://doi.org/10.1007/978-3-642-40708-6_17)
- [14] F. Camporesi, J. Feret, K. Q. Lý, KaDE: a tool to compile Kappa rules into (reduced) ODEs models, in: Fifteenth International Workshop on Static Analysis and Systems Biology (SASB'17), Vol. 10545 of LNCS/LNBI, springer, 2017.
- [15] L. Cardelli, M. Tribastone, M. Tschaikowski, A. Vandin, [Symbolic computation of differential equivalences](#), Theoretical Computer Science 777 (2019) 132–154.  
URL <https://doi.org/10.1016/j.tcs.2019.03.018>
- [16] A. Jiménez-Pastor, J. P. Jacob, G. Pogudin, [Exact linear reduction for rational dynamical systems](#), in: Computational Methods in Systems Biology, Springer International Publishing, 2022, pp. 198–216.  
URL [https://doi.org/10.1007/978-3-031-15034-0\\_10](https://doi.org/10.1007/978-3-031-15034-0_10)
- [17] G. Li, H. Rabitz, [A general analysis of exact lumping in chemical kinetics](#), Chemical Engineering Science 44 (6) (1989) 1413–1430.  
URL [https://doi.org/10.1016/0009-2509\(89\)85014-6](https://doi.org/10.1016/0009-2509(89)85014-6)
- [18] L. Rónyai, [Computing the structure of finite algebras](#), Journal of Symbolic Computation 9 (3) (1990) 355–373.  
URL [https://doi.org/10.1016/S0747-7171\(08\)80017-X](https://doi.org/10.1016/S0747-7171(08)80017-X)

- [19] A. Chistov, G. Ivanyos, M. Karpinski, [Polynomial time algorithms for modules over finite dimensional algebras](#), in: Proceedings of the 1997 International Symposium on Symbolic and Algebraic Computation, 1997, p. 68–74.  
URL <https://doi.org/10.1145/258726.258751>
- [20] D. Speyer, [Response to “Is there a clean way to extract the subspaces invariant under a list of matrices?”](#)  
URL <https://mathematica.stackexchange.com/questions/6519/is-there-a-clean-way-to-extract>
- [21] R. S. Malik-Sheriff, M. Glont, T. V. N. Nguyen, K. Tiwari, M. G. Roberts, A. Xavier, M. T. Vu, J. Men, M. Maire, S. Kananathan, E. L. Fairbanks, J. P. Meyer, C. Arankalle, T. M. Varusai, V. Knight-Schrijver, L. Li, C. Dueñas-Roca, G. Dass, S. M. Keating, Y. M. Park, N. Buso, N. Rodriguez, M. Hucka, H. Hermjakob, [BioModels — 15 years of sharing computational models in life science](#), Nucleic Acids Research 48 (D1) (2020) D407–D415.  
URL <https://doi.org/10.1093/nar/gkz1055>
- [22] G. Pogudin, X. Zhang, [Interpretable exact linear reductions via positivity](#), in: E. Cinquemani, L. Paulevé (Eds.), Computational Methods in Systems Biology, 2021, pp. 91–107.  
URL [https://doi.org/10.1007/978-3-030-85633-5\\_6](https://doi.org/10.1007/978-3-030-85633-5_6)
- [23] S. Dunn, A. Constantinides, P. Moghe, [Numerical Methods in Biomedical Engineering](#), Academic Press, 2006.  
URL <https://doi.org/10.1016/B978-0-12-186031-8.X5000-6>
- [24] M. Feinberg, [Foundations of Chemical Reaction Network Theory](#), Springer Cham, 2019.  
URL <https://doi.org/10.1007/978-3-030-03858-8>
- [25] M. Reineke, [Every projective variety is a quiver Grassmannian](#), Algebras and Representation Theory 16 (2013) 1313–1314.  
URL <https://doi.org/10.1007/s10468-012-9357-z>
- [26] E. Nijholt, B. W. Rink, S. Schwenker, [Quiver representations and dimension reduction in dynamical systems](#), SIAM Journal on Applied Dynamical Systems 19 (4) (2020) 2428–2468.  
URL <https://doi.org/10.1137/20m1345670>
- [27] Y. A. Drozd, V. V. Kirichenko, Finite Dimensional Algebras, Springer-Verlag, 1994.
- [28] M. R. Bremner, [How to compute the Wedderburn decomposition of a finite-dimensional associative algebra](#), Groups, Complexity, Cryptology 3 (1) (2011) 47–66.  
URL <https://doi.org/10.1515/gcc.2011.003>
- [29] I. Klep, J. Volčič, A note on group representations, determinantal hypersurfaces and their quantizations, in: M. A. Bastos, L. Castro, A. Y. Karlovich

- (Eds.), *Operator Theory, Functional Analysis and Applications*, Springer International Publishing, 2021, pp. 393–402.
- [30] S. D. Cohen, [The distribution of Galois groups and Hilbert’s irreducibility theorem](#), *Proceedings of the London Mathematical Society* s3-43 (2) (1981) 227–250.  
URL <https://doi.org/10.1112/plms/s3-43.2.227>
- [31] R. Zippel, *Effective Polynomial Computation*, Springer, 1993.  
URL <http://dx.doi.org/10.1007/978-1-4615-3188-3>
- [32] S. Axler, *Linear Algebra Done Right*, Springer Cham, 2015.  
URL <https://doi.org/10.1007/978-3-319-11080-6>
- [33] J. Bezanson, A. Edelman, S. Karpinski, V. B. Shah, [Julia: A fresh approach to numerical computing](#), *SIAM review* 59 (1) (2017) 65–98.  
URL <https://doi.org/10.1137/141000671>
- [34] C. Fieker, W. Hart, T. Hofmann, F. Johansson, [Nemo/hecke: Computer algebra and number theory packages for the julia programming language](#), in: *Proceedings of the 2017 ACM on International Symposium on Symbolic and Algebraic Computation, ISSAC ’17*, ACM, New York, NY, USA, 2017, pp. 157–164.  
URL <https://doi.acm.org/10.1145/3087604.3087611>
- [35] W. B. Hart, *Fast library for number theory: An introduction*, in: *Proceedings of the Third International Congress on Mathematical Software, ICMS’10*, Springer-Verlag, Berlin, Heidelberg, 2010, pp. 88–91,  
<https://flintlib.org>
- [36] F. Johansson, [Calcium](#), in: *Proceedings of the 2021 on International Symposium on Symbolic and Algebraic Computation*, 2021.  
URL <https://doi.org/10.1145/3452143.3465513>
- [37] T. A. Davis, [Algorithm 1000: Suitesparse:graphblas: Graph algorithms in the language of sparse linear al](#), *ACM Trans. Math. Softw.* 45 (4) (dec 2019).  
URL <https://doi.org/10.1145/3322125>
- [38] [ERODE: Evaluation and Reduction of Stochastic Reaction Networks and Differential Equations](#), user’s manual (2020).  
URL <https://www.erode.eu/docs/ERODE-manual.pdf>
- [39] I. C. Pérez-Verona, M. Tribastone, A. Vandin, *A large-scale assessment of exact model reduction in the BioModels repository*, in: L. Bortolussi, G. Sanguinetti (Eds.), *Computational Methods in Systems Biology*, Springer International Publishing, 2019, pp. 248–265.
- [40] B. Assarf, E. Gawrilow, K. Herr, M. Joswig, B. Lorenz, A. Paffenholz, T. Rehn, [Computing convex hulls and counting integer points with polymake](#), *Math. Program. Comput.* 9 (1) (2017) 1–38.  
URL <http://dx.doi.org/10.1007/s12532-016-0104-z>

- [41] M. F. Hockin, K. M. Cawthorn, M. Kalafatis, K. G. Mann, [A model describing the inactivation of factor Va by APC: Bond cleavage, fragment dissociation, and pro](#) Biochemistry 38 (21) (1999) 6918–6934.  
URL <https://doi.org/10.1021/bi981966e>
- [42] Y. Ma, S. Gowda, R. Anantharaman, C. Laughman, V. Shah, C. Rackauckas, ModelingToolkit: A composable graph transformation system for equation-based modeling (2021). [arXiv:2103.05244](#).
- [43] C. Rackauckas, Q. Nie, DifferentialEquations.jl—a performant and feature-rich ecosystem for solving differential equations in Julia, Journal of Open Research Software 5 (1) (2017).
- [44] M. Schliemann, E. Bullinger, S. Borchers, F. Allgöwer, R. Findeisen, P. Scheurich, [Heterogeneity reduces sensitivity of cell death for TNF-stimuli](#), BMC Systems Biology 5 (204) (2011).  
URL <https://doi.org/10.1186/1752-0509-5-204>
- [45] J. K. Scott, P. I. Barton, [Bounds on the reachable sets of nonlinear control systems](#), Automatica 49 (1) (2013) 93–100.  
URL <https://doi.org/10.1016/j.automatica.2012.09.020>
- [46] M. Tschaikowski, M. Tribastone, [Approximate reduction of heterogenous nonlinear models with differentia](#) IEEE Transactions on Automatic Control 61 (4) (2016) 1099–1104.  
URL <https://doi.org/10.1109/TAC.2015.2457172>
- [47] A. Girard, G. J. Pappas, [Approximate bisimulation: A bridge between computer science and control theor](#) European Journal of Control 17 (5) (2011) 568–578.  
URL <https://doi.org/10.3166/ejc.17.568-578>
- [48] S. Soliman, F. Fages, O. Radulescu, [A constraint solving approach to model reduction by tropical equilibration](#), Algorithms for Molecular Biology 9 (1) (2014).  
URL <https://doi.org/10.1186/s13015-014-0024-2>
- [49] A. Goyer, [A Sage package for the symbolic-numeric factorization of linear differential operators](#), ACM Commun. Comput. Algebra 55 (2) (2021) 44–48.  
URL <https://doi.org/10.1145/3493492.3493496>
- [50] F. Chyzak, A. Goyer, M. Mezzarobba, [Symbolic-numeric factorization of differential operators](#), in: Proceedings of the 2022 International Symposium on Symbolic and Algebraic Computation, ISSAC '22, 2022, p. 73–82.  
URL <https://doi.org/10.1145/3476446.3535503>