Computing exact nonlinear reductions of dynamical models

Antonio Jiménez-Pastor
LIX, CNRS, École Polytechnique
Institute Polytechnique de Paris
Palaiseau, France, 92160
jimenezpastor@lix.polytechnique.fr

Gleb Pogudin
LIX, CNRS, École Polytechnique
Institute Polytechnique de Paris
Palaiseau, France, 92160
gleb.pogudin@polytechnique.edu

Abstract

Dynamical systems are commonly used to represent real-world processes. Model reduction techniques are among the core tools for studying dynamical systems models, they allow to reduce the study of a model to a simpler one. In this poster, we present an algorithm for computing exact nonlinear reductions, that is, a set of new rational function macro-variables which satisfy a self-consistent ODE system with the dynamics defined by algebraic functions. We report reductions found by the algorithm in models from the literature.

1 Introduction

Dynamical systems are frequently used to model phenomena in the life sciences and engineering. It is well-known that even relatively small dynamical systems may have very complicated dynamics (e.g., the celebrated Lorenz system), and, as the dimension grows, studying a model becomes more and more complicated.

One standard way of dealing with high-dimensional models is to use model reduction: replace a model with a simpler one which preserves, at least approximately, some features of the original model. Perhaps, the most commonly used tools are the ones for approximate reduction (see, e.g. [14]). However, such reductions typically introduce approximation errors and may destroy intrinsic structural properties of the model. Therefore, it may be beneficial to complement the approximate reduction techniques with the exact ones. The existing methods for exact model reduction include:

- Finding first integrals (i.e., conserved quantities). Indeed, a first integral h of the system can be viewed as a reduction to a one-dimensional system $h' = 0$. A number of algorithms has been proposed to find the first integrals, e.g. [16, 4, 13, 15].
- Finding exact reductions among linear projections. Efficient algorithms and publicly available software exist for this problem, see [5, 11] and references therein.
- The invariance of the system with respect to a group action can be used to perform exact reduction. An efficient algorithm was proposed in [7] for the case of scaling transformations, extending the celebrated Buckingham π-theorem.

In this paper, we present an algorithm that, given an ansatz of the form of a rational function involving the state variables of the system and unknown coefficients, finds the constraints on the values of the coefficients, under which this function can be completed to a nontrivial nonlinear reduction. This allows us to find reductions of more general form than possible using the approaches mentioned above (see examples in Section 4). On the other hand, our algorithm relies on polynomial system solving, so it cannot tackle very large systems that can be analyzed using these less general approaches.

Once the coefficients are chosen and a reduction is built, our algorithm may perform additional reparametrization of the resulting reduction aiming at simplifying the new variables and reduced system.

We give a high-level description of our algorithm and show some interesting reductions found by our implementation for models from the literature.

2 Problem statement

Let $k \subset \mathbb{C}$ be a constructive field. We consider a rational dynamical system, that is, a system of differential equations

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

in variables $\mathbf{x} = (x_1, \ldots, x_n)$, where $\mathbf{f} = (f_1, \ldots, f_n)$ is a tuple of rational functions $f_1, \ldots, f_n \in k(\mathbf{x})$. The integer n will be referred to as *the dimension* of (1).

Our main goal is to find *exact reductions* of (1). Informally speaking, *reduction* is a list of rational functions $y_1(\mathbf{x}), \ldots, y_m(\mathbf{x})$ with $m < n$ such that $y_1, \ldots, y_m$ satisfy a self-contained system of the form (1) but possible with algebraic functions in the right-hand side. That is, there exist algebraic functions $g_1(\mathbf{y}), \ldots, g_m(\mathbf{y})$ such that, for every solution $\mathbf{x}^*$ of (1), the functions $y_1(\mathbf{x}^*), \ldots, y_m(\mathbf{x}^*)$ satisfy

$$y_1' = g_1(\mathbf{y}), \ldots, y_m' = g_m(\mathbf{y})$$

after a suitable choice of branches for $g_1, \ldots, g_m$. In practice, it frequently happens (see Section 4) that the right-hand side of the reduced system is rational as well. We give a formal algebraic definition of reduction below.

Definition 1 (Lie derivative) *Since* (1) *defines a vector field, for every rational function $h \in k(\mathbf{x})$ we can define the* Lie derivative *with respect to* (1)*:*

$$\mathcal{L}(h) := \sum_{i=1}^{n} f_i \frac{\partial h}{\partial x_i}.$$

Definition 2 (Reduction) *A list $y_1, \ldots, y_m \in k(\mathbf{x})$ of k-algebraically independent rational functions with $m < n$ will be called a reduction of* (1) *if, for every $1 \leqslant i \leqslant m$, $\mathcal{L}(y_i)$ is algebraic over the field $k(y_1, \ldots, y_m)$.*

Remark 1 *The definition above can be easily adapted to the case when one looks for rational (resp., polynomial) reduction: one should just replace algebraicity of $\mathcal{L}(y_i)$ over $k(y_1, \ldots, y_m)$ by the containment $\mathcal{L}(y_i) \in k(y_1, \ldots, y_m)$ (resp., $\mathcal{L}(y_i) \in k[y_1, \ldots, y_m]$). But the corresponding algorithmic problem of finding such reductions seems to be very challenging, and we are not aware of any practical solution.*

We will illustrate the definition on a couple of examples.

Example 1 ([1, Example 4.1.9]) *Consider the following dynamical system:*

$$\begin{cases} x' &= xz \\ y' &= yz \\ z' &= -x^2 - y^2 \end{cases} \tag{2}$$

Then, the functions $f = z$ and $g = x^2 + y^2$ are a reduction of (2) *since:*

$$\mathcal{L}(f) = -g, \quad \mathcal{L}(g) = 2fg.$$

Therefore, for every solution (x, y, z) of (2)*, the values of f and g satisfy the reduced system*

$$f' = -g, \quad g' = 2fg.$$

Example 2 *Consider the following dynamical system:*

$$\begin{cases} x' &= -2x^2y \\ y' &= x^2y^3 + y \end{cases} \tag{3}$$

Then, the function $f = xy$ is a reduction of (3) since:

$$\mathcal{L}(f) = f^3 - 2f^2 + f,$$

and the reduced system will be $f' = f^3 - 2f^2 + f$.

3 Sketch of the algorithm

3.1 General idea behind the algorithm

Our algorithm is based on the following observation.

Proposition 1 *Consider system (1) of dimension n. Let $y_1, \ldots, y_m \in k(\mathbf{x})$ be a reduction of (1). Then there exists $r \leqslant m$ such that*

$$y_1,\ \mathcal{L}(y_1),\ \ldots,\ \mathcal{L}^{r-1}(y_1)$$

is also a reduction of (1). Furthermore, any element $\mathcal{L}^i(y_1)$ of this reduction is algebraic over $k(y_1, \ldots, y_m)$.

This proposition implies that every reduction contains a "subreduction" of the form $y, \mathcal{L}(y), \ldots, \mathcal{L}^{r-1}(y)$ for some $y \in k(\mathbf{x})$. Moreover, using the differential primitive element theorem [12], one can show that any reduction with nonconstant dynamics is equivalent to a reduction of this form. Our approach is the following:

1. We fix a positive integer $r \geqslant 1$ and an ansatz $y \in k(\mathbf{a}, \mathbf{x})$, where $\mathbf{a}$ is a vector of unknown coefficients.
2. We express the fact that "$y, \mathcal{L}(y), \ldots, \mathcal{L}^{r-1}(y)$ contains a reduction" as a polynomial system on the ansatz coefficients $\mathbf{a}$ (Section 3.2).
3. We analyze the solution set of this polynomial system and consider y's resulting from different prime components.
4. For each of the constructed y's, we form the corresponding reduction $y, \mathcal{L}(y), \ldots, \mathcal{L}^{s-1}(y)$ and apply our simplification procedure to it in order to make the resulting reduction easier to interpret and compute with (Section 3.3).
5. For each of the computed reductions, we produce the reduced ODE system.

3.2 Formulating the existence of reduction via a polynomial system

Consider any $y \in k(\mathbf{x})$ and positive integer $r \geqslant 1$. The definition of reduction implies that the functions $y, \mathcal{L}(y), \ldots, \mathcal{L}^{r-1}(y)$ contains a reduction if and only if $\mathcal{L}^r(y)$ is algebraic over $k(y, \mathcal{L}(y), \ldots, \mathcal{L}^{r-1}(y))$. Furthermore, one can show that this is equivalent to $y, \mathcal{L}(y), \ldots, \mathcal{L}^r(y)$ being algebraically dependent. Algebraic dependence in $k(\mathbf{x})$ can be verified using the following lemma (which follows from [6, Theorem 16.14]):

Lemma 1 *Let $y_1, \ldots, y_m \in k(\mathbf{x})$ be rational functions. They are algebraically dependent over k if and only if the rank of the matrix*

$$\left(\nabla y_1 \mid \nabla y_2 \mid \ldots \mid \nabla y_m\right),$$

where $\nabla y = \left(\frac{\partial y}{\partial x_1}, \ldots, \frac{\partial y}{\partial x_n}\right)^T$, is less than m.

With this lemma at hand, we can search for such y using an *ansatz*:

$$y(\mathbf{a}, \mathbf{x}) \in k(\mathbf{a}, \mathbf{x}), \tag{4}$$

where $\mathbf{a}$ is the tuple of new indeterminates. The above discussion implies that y can be completed to a reduction of dimension at most r if and only if the following matrix has rank less than $r + 1$:

$$J = (\nabla y \mid \nabla \mathcal{L}(y) \mid \ldots \mid \nabla \mathcal{L}^r(y))^T .$$

This rank inequality is equivalent to all $(r + 1) \times (r + 1)$ minors of J to be zero. The entries of J belong to $k(\mathbf{a}, \mathbf{x})$, the minors are zero if their numerators are zero polynomials in $\mathbf{x}$. Thus, the coefficients w.r.t. $\mathbf{x}$ of these numerators provide a system of polynomial equations on $\mathbf{a}$ such that, if we take a solution of it, we get a function $y^* \in k(\mathbf{x})$ such that $\{\mathcal{L}^i(y^*) \mid i = 0, \ldots, s - 1\}$ is a reduction of (1) for some $s \leqslant r$.

Let $\mathcal{A} \subset k[\mathbf{a}]$ be the ideal generated by the constructed polynomial system. In order to get the components of the solution variety, we use triangular sets [8, 9, 17]. Once we have one of these components, we can plug a generic point from it to the ansatz and obtain a reduction by computing necessary Lie derivatives.

3.3 Computing simpler generators

In the previous subsection, we have described a method to obtain a reduction $y_1, \ldots, y_m$ for a system of the form (1). In this subsection, we will describe a method allowing to simplify the set of new variables while obtaining an equivalent reduction. More precisely, for a reduction $y_1, \ldots, y_m \in k(\mathbf{x})$ with algebraically independent $y_1, \ldots, y_m$, we try to obtain an equivalent reduction $z_1, \ldots, z_m \in k(\mathbf{x})$ such that $k(y_1, \ldots, y_m) = k(z_1, \ldots, z_m)$ and each z_i is either equal to y_i or is a polynomial of small degree.

For doing so, we will use again an ansatz approach. We choose a degree bound d, and consider z to be a general polynomial of degree at most d in $\mathbf{x}$ with undetermined coefficients. Next we want to check if one of the $y_1, \ldots, y_m$ can be replaced by z. This is equivalent to the fact that $y_1, \ldots, y_m, z$ are algebraically dependent. Using Lemma 1, we know this is equivalent to the matrix $J = (\nabla y_1 | \nabla y_2 | \ldots | \nabla y_m | \nabla z)^T$ having the rank less than $m + 1$, so all the $(m + 1) \times (m + 1)$ minors of this matrix must vanish. As we did previously, we now take the numerators and coefficients w.r.t. $\mathbf{x}$, obtaining a system of equations in the ansatz coefficients.

However, this system differs from the system obtain in Section 3.2 since the ansatz coefficients appear only in ∇z. Hence, the system we obtain is **linear** in the ansatz variables. We can, therefore, obtain a basis of solutions for this system.

For each linearly independent solution, we can then compute a concrete function z^* that we know is algebraically dependent with $y_1, \ldots, y_m$. If we now solve the linear system

$$\nabla z^* = \mathbf{v}(\mathbf{x}) \begin{pmatrix} \nabla y_1 \\ \vdots \\ \nabla y_m \end{pmatrix},$$

we know that we can exchange for z^* any of the y_i if the i-th component of $\mathbf{v}(\mathbf{x})$ is not zero.

This process only involves *linear algebra* and can be the repeated as many times as desired increasing the degree bound d if necessary.

3.4 Computing the new equations

Up to this point, we have only considered the problem of deciding when a set of algebraically independent functions $y_1, \ldots, y_m \in k(\mathbf{x})$ is a reduction for a dynamical system (1). By definition of reduction, this means that there are algebraic functions $g_1, \ldots, g_m$ over $k(y_1, \ldots, y_m)$ such that $\mathcal{L}(y_i) = g_i$ for $i = 1, \ldots, m$. The minimal polynomials of g_i is the minimal polynomial for $\mathcal{L}(y_i)$ over $k(y_1, \ldots, y_m)$ and can be computed, for example, using [10, Algorithm 3.2].

4 Examples

Reduction in Examples 1 and 2 were found using our algorithm. In this section we will show a couple of larger examples. Examples 1 and 3 were taken from a collection of ODE models used in [16].

Example 3 (Phytoplankton model) *The following dynamical system model of the usage of carbon by the phytoplankton during the photosynthesis process has been studied in [2, Section 6]:*

$$\begin{cases} x_1' &= 1 - x_1 - a x_1 x_2 \\ x_2' &= -b x_2 + 2 x_2 x_3 \\ x_3' &= a x_1 - b x_3^2 \end{cases} \tag{5}$$

If we set up a quadratic ansatz and apply the procedure described in Section 3.2 for one Lie derivative, we obtain that for any constant α, the function $y = \alpha(x_1 + x_2x_3)$ is a reduction of (5). No further simplification was possible, so our algorithm will produce the following equation for y:

$$y' = \alpha - y.$$

This reduction (for $\alpha = 1$) was proven in [2, Property 4] and now we can find it automatically.

Example 4 (Ordered Phosphorylation) *Let us consider the following dynamical system:*

$$\begin{cases} x_1' &= -2Kx_1x_2 + kx_3, \\ x_2' &= -2Kx_1x_2 - kx_2x_3 + kx_3 + 2kx_4, \\ x_3' &= 2Kx_1x_2 - Kx_2x_3 - kx_3 + 2kx_4, \\ x_4' &= Kx_2x_3 - 2kx_4, \end{cases} \tag{6}$$

where the k and K are scalar parameters. This model is a result of the reduction of the 227-dimensional model from [3] using the software CLUE [11] which searches for the new variables as linear combinations of the original ones.

If we apply the methodology of Section 3.2 to the linear ansatz $z = a_1x_1 + a_2x_2 + a_3x_3 + a_4x_4$ and $r = 3$ derivations, we obtain that $z, \mathcal{L}(z), \mathcal{L}^2(z)$ is a reduction of (6) if and only if the ansatz variables satisfy the equation

$$a_1 - 2a_3 + a_4 = 0.$$

For example, we can take the function $z^* = x_2$ and applying Section 3.4 we obtain the following reduced system for $y_1 = z^*$, $y_2 = \mathcal{L}(z^*)$ and $y_3 = \mathcal{L}^2(z^*)$:

$$\begin{cases} y_1' &= y_2 \\ y_2' &= y_3 \\ y_3' &= \frac{-2Ky_2^3 + y_3^2}{y_2} \end{cases} \tag{7}$$

We can see that the equations for y_2' and y_3' do not involve y_1, so this system can be further reduced by removing the first equation.

If we write y_2 and y_3 explicitly in terms of the state variables, we will obtain:

$$\begin{aligned} y_2 &= -2Kx_1x_2 - kx_2x_3 + kx_3 + 2kx_4 \\ y_3 &= 4K^2x_1^2x_2 + (4K^2 - 2kK)x_1x_2^2 + 4kKx_1x_2x_3 \\ &\quad +kKx_2^2x_3 + k^2x_2x_3^2 + 2kKx_1x_2 \\ &\quad -2kKx_1x_3 + (k^2 - kK)x_2x_3 - k^2x_3^2 \\ &\quad -4kKx_1x_4 - 2k^2x_2x_4 - 2k^2x_3x_4 \\ &\quad -k^2x_3 - 2k^2x_4 \end{aligned}$$

The size of the expression motivates us to use the simplification algorithm from Section 3.3.

We obtain the following simpler functions:

$$z_1 = 2x_1 + x_2 + x_3, \quad z_2 = 2x_1x_2 + x_2x_3 + \frac{k}{K}\left(2x_1 + x_2 - 2x_4\right).$$

Finally, applying Section 3.4, we obtain the following reduced system:

$$\begin{cases} z_1' &= 2kz_1 - 2Kz_2 \\ z_2' &= kz_1 - Kz_1z_2 + 3\frac{k^2}{K}z_1 - 3kz_2 \end{cases}$$

which is polynomial unlike (7).

Acknowledgements

This work was supported by the Paris Ile-de-France region (via project “XOR”). GP was partially supported by NSF grants DMS-1853482, DMS-1760448, and DMS-1853650. We are grateful to Mirco Tribastone for helpful discussions and, in particular, for suggesting Example 4.

References

[1] J. M. Alongi and G. S. Nelson. *Recurrence and topology.* American Mathematical Society, Providence, RI, 2012.

[2] O. Bernard and J.-L. Gouzé. Global qualitative description of a class of nonlinear dynamical systems. *Artificial Intelligence*, 136(1):29–59, 2002. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(01)00169-2. URL https://www.sciencedirect.com/science/article/pii/S0004370201001692.

[3] N. Borisov, B. Kholodenko, J. Faeder, and A. Chistopolsky. Domain-oriented reduction of rule-based network models. *IET Systems Biology*, 2(5):342–351, 2008. URL https://doi.org/10.1049/iet-syb:20070081.

[4] A. Bostan, G. Chéze, T. Cluzeau, and J.-A. Weil. Efficient algorithms for computing rational first integrals and Darboux polynomials of planar polynomial vector fields. *Mathematics of Computation*, 85:1393–1425, 2016. URL https://doi.org/10.1090/mcom/3007.

[5] L. Cardelli, M. Tribastone, M. Tschaikowski, and A. Vandin. ERODE: A tool for the evaluation and reduction of ordinary differential equations. In A. Legay and T. Margaria, editors, *Tools and Algorithms for the Construction and Analysis of Systems*, pages 310–328, 2017. ISBN 978-3-662-54580-5.

[6] D. Eisenbud. *Commutative algebra.* Graduate Texts in Mathematics. Springer, New York, NY, 1 edition, Feb. 1999.

[7] E. Hubert and G. Labahn. Scaling invariants and symmetry reduction of dynamical systems. *Foundations of Computational Mathematics*, 13:479–516, 2013. URL https://doi.org/10.1007/s10208-013-9165-9.

[8] M. Kalkbrener. A generalized euclidean algorithm for computing triangular representations of algebraic varieties. *Journal of Symbolic Computation*, 15(2):143–167, 1993. ISSN 0747-7171. doi: https://doi.org/10.1006/jsco.1993.1011. URL https://www.sciencedirect.com/science/article/pii/S0747717183710114.

[9] Y. Lu. Searching dependency between algebraic equations: an algorithm applied to automated reasoning. 1994.

[10] J. Müller-Quade and S. R. Basic algorithms for rational function fields. *Journal of Symbolic Computation*, 27(2):143–170, 1999. URL `https://doi.org/10.1006/jsco.1998.0246`.

[11] A. Ovchinnikov, I. Pérez Verona, G. Pogudin, and M. Tribastone. CLUE: exact maximal reduction of kinetic models by constrained lumping of differential equations. *Bioinformatics*, 37(19):3385–3385, 2021. URL `https://doi.org/10.1093/bioinformatics/btab258`.

[12] G. Pogudin. A primitive element theorem for fields with commuting derivations and automorphisms. *Selecta Mathematica*, 27, 2019. URL `https://doi.org/10.1007/s00029-019-0504-9`.

[13] M. Prelle and M. F. Singer. Elementary first integrals of differential equations. *Transactions of American Mathematical Society*, 279:215–229, 1983. URL `https://doi.org/10.1090/S0002-9947-1983-0704611-X`.

[14] W. H. A. Schilders, H. A. Vorst, and J. Rommes. *Model Order Reduction: Theory, Research Aspects and Applications*. Springer Berlin, Heidelberg, 2008. ISBN 978-3-540-78840-9. URL `https://doi.org/10.1007/978-3-540-78841-6`.

[15] D. Schlomiuk. Elementary first integrals and algebraic invariant curves of differential equations. In *Exposition. Math*, volume 11, pages 433–454, 1993.

[16] A. Sogokon, S. Mitsch, Y. K. Tan, K. Cordwell, and A. Platzer. Pegasus: Sound continuous invariant generation. *Form. Methods Syst. Des.*, 58(1):5–41, 2022. ISSN 0925-9856. doi: 10.1007/s10703-020-00355-z. Special issue for selected papers from FM'19.

[17] D. Wang. Computing triangular systems and regular systems. *Journal of Symbolic Computation*, 30(2):221–236, 2000. ISSN 0747-7171. doi: https://doi.org/10.1006/jsco.1999.0355. URL `https://www.sciencedirect.com/science/article/pii/S0747717199903553`.