SIAM J. APPLIED DYNAMICAL SYSTEMS
Vol. 23, No. 1, pp. 982–1016

Exact and Optimal Quadratization of Nonlinear Finite-Dimensional Nonautonomous Dynamical Systems*

Andrey Bychkov†, Opal Issan‡, Gleb Pogudin§, and Boris Kramer‡

Abstract. Quadratization of polynomial and nonpolynomial systems of ordinary differential equations (ODEs) is advantageous in a variety of disciplines, such as systems theory, fluid mechanics, chemical reaction modeling, and mathematical analysis. A quadratization reveals new variables and structures of a model, which may be easier to analyze, simulate, and control, and provides a convenient parametrization for learning. This paper presents novel theory, algorithms, and software capabilities for quadratization of nonautonomous ODEs. We provide existence results, depending on the regularity of the input function, for cases when a quadratic-bilinear system can be obtained through quadratization. We further develop existence results and an algorithm that generalizes the process of quadratization for systems with arbitrary dimension that retain the nonlinear structure when the dimension grows. For such systems, we provide dimension-agnostic quadratization. An example is semidiscretized PDEs, where the nonlinear terms remain symbolically identical when the discretization size increases. As an important aspect for practical adoption of this research, we extended the capabilities of the QBee software towards both nonautonomous systems of ODEs and ODEs with arbitrary dimension. We present several examples of ODEs that were previously reported in the literature, and where our new algorithms find quadratized ODE systems with lower dimension than the previously reported lifting transformations. We further highlight an important area of quadratization: reduced-order model learning. This area can benefit significantly from working in the optimal lifting variables, where quadratic models provide a direct parametrization of the model that also avoids additional hyperreduction for the nonlinear terms. A solar wind example highlights these advantages.

Key words. nonlinear dynamical systems, quadratization, polynomialization, symbolic computation, lifting transformation

MSC codes. 34A34, 34C20, 68W30, 93B11

DOI. 10.1137/23M1561129

1. Introduction.

Evolutionary processes in engineering and science are often modeled with nonlinear nonautonomous ordinary differential equations (ODEs) that describe the time evolution of the states of the system, i.e., the physically necessary and relevant variables. However,

*Received by the editors March 24, 2023; accepted for publication (in revised form) by I. Belykh December 4, 2023; published electronically March 26, 2024.
https://doi.org/10.1137/23M1561129
Funding: The fourth author was supported in part by NSF-CMMMI award 2144023. Part of fourth author's work on this paper was undertaken during the program "The Mathematical and Statistical Foundation of Future Data-Driven Engineering" (supported by EPSRC grant EP/R014604/1) and during a visit to École Polytechnique within the PANTOMIME project funded by AAP INS2I CNRS. The third author was partially supported by the PANTOMIME project, the Paris Ile-de-France region, and NSF grants DMS-1760448 and DMS-1853650.
†Thales Research & Technology, France (abychkov_edu@proton.me).
‡Department of Mechanical and Aerospace Engineering, University of California San Diego, La Jolla, CA 92093-0411 USA (oissan@ucsd.edu, bmkramer@ucsd.edu).
§LIX, CNRS, École Polytechnique, Institute Polytechnique de Paris, France (gleb.pogudin@polytechnique.edu).

these models are not unique: the same evolutionary process can be modeled with different variables, which can have a tremendous impact on computational modeling and analysis. This idea of variable transformation (referred to as *lifting* when extra variables are added) to promote model structure is found across different communities, with literature spanning half a century. We first review general variable transformations and then focus on the appealing quadratic form.

In fluid dynamics, variable transformations have long been recognized as providing useful alternative representations. While the Euler and Navier–Stokes equations are most commonly derived in conservative variables, where each state is a conserved quantity (mass, momentum, energy), symmetric variables have been exploited to guarantee stable models [34]. Additionally, recent work in [27] shows that the fluid and plasma equations can be reformulated via an antisymmetric variable transformation to retain robust conservation properties in numerical simulations. Stability-preserving inner products for projection-based reduced-order models (ROMs) hint at variable transformations as a proper choice to increase stability for nonlinear fluid models; see [38, 55]. As another classic example, the well-known Cole–Hopf transformation turns the nonlinear Burgers' partial differential equation (PDE) into a linear PDE [20, 32]. In the dynamical systems field, the Koopman operator is a linear infinite-dimensional operator that describes the dynamics of observables of nonlinear systems. Dynamic mode decomposition [57, 60] approximates the spectrum of the Koopman operator, and the choice of observables has a significant impact on the quality of the learned model, which led to the extended dynamic mode decomposition algorithm [62, 50]. For control design, the idea of transforming a general nonlinear system into a system with more structure is also common practice: the concept of feedback linearization transforms a general nonlinear system into a structured linear model [37, 40]. This is done via a nonlinear state transformation, where the transformed state might be augmented, i.e., have increased dimension relative to the original state. The lifting transformations known in feedback linearization are specific to the desired target model form. A change of variables can also ensure that physical constraints are more easily met in a simulation; see [49, 28]. Bringing nonlinear systems into canonical and abstract forms can then improve analysis, as seen in [45, 12]. The authors in [58] showed that all ODE systems with (nested) elementary functions can be recast in a special polynomial system form, which is then faster to solve numerically.

Quadratic model structure, as a special case, has seen broad interest due to the ease of working with quadratic models. In the context of optimization, McCormick [46] is credited with first introducing variable substitutions to achieve quadratic structure so that nonconvex optimization problems can be recast as convex problems in the new variables. In fluid mechanics, the quadratic model structure of the specific volume variable representation has been exploited in [6] to allow for model stabilization. To analyze equilibrium branches of geometrically nonlinear finite element models, the authors of [25] recast the model into a quadratic form, for which the Jacobian and a specific Taylor series can be easily obtained. They then use the Asymptotic Numerical Method to find the equilibrium branches. In the area of analog computing with chemical reaction networks, quadratic forms correspond to the notion of elementary chemical reactions. Transforming an arbitrary polynomial ODE system into a quadratic one can be used to establish the Turing completeness of elementary chemical reactions [11, 22, 30]. Lifted models in quadratic and quadratic-bilinear (QB) form have seen great interest in the

systems and control community in the past decade, as the QB structure is appealing for intrusive model reduction [24, 8, 9, 42, 43]. Those methods project the quadratic operators of a high-dimensional lifted model onto a reduced space to obtain a quadratic reduced model. The quadratic model structure eliminates the need for additional hyperreduction/interpolation to reduce the expense of evaluating the nonlinear term [19, 5, 7, 15, 51], where in some cases the number of interpolation points required for accuracy eliminates computational gains [10, 33]. Moreover, the quadratic model structure has fewer degrees of freedom in the ROM than a cubic or high-order polynomial model, making quadratic models the most desired lifted form. The *Lift & Learn* method [54, 53] and related work [61, 23, 36, 47] leverage lifting transformations to learn low-order polynomial ROMs of complex nonlinear systems, such as combustion dynamics, from lifted data. For quadratic and cubic model structures, one can equip these learned ROMs with stability guarantees; see [41, 59].

With such great appeal and interest in quadratic and polynomial dynamical system structure, it is natural to ask: Which systems can be brought into polynomial (via *polynomialization*) and further into quadratic (via *quadratization*) form? Which algorithms and methods exist to perform such transformations? The theoretical results that any system written using (nested) elementary functions can be polynomialized and every polynomial system can be quadratized have been discovered and established in different communities and contexts; see [4, 44, 39, 16, 24, 17, 18]. The proofs of these theorems are constructive and can be turned into algorithms. However, a straightforward approach would require introducing an excessively large number of variables and thus making the resulting dynamical system hard to use and/or analyze. The resulting quadratic systems can be pruned using constrained programming techniques, and designed algorithms and methods are presented for polynomialization in [31] and quadratization in [30]. Both algorithms are implemented in the Biocham software [2]. We take a different approach in [13] where the quadratization problem for a polynomial system is framed as a tree exploration. The quadratization is obtained by building up the tree structure rather than pruning it as in [30]. Attractive features of the resulting algorithm QBee [14] include optimality guarantees for the dimension of the lifted system and good performance in practice [13, Table 3]. Although the previous version of QBee performed well on a selection of benchmark models from synthetic biology and some academic examples (see [13]), we found its applicability to engineering problems quite limited. In particular, dynamical systems models of engineering systems share common features that the previous version of QBee cannot handle:

1. Engineering models are often driven by time-dependent inputs or controls, i.e., they are nonautonomous. Currently, there is no suitable theory for quadratization of such systems, and the previous version of QBee could not handle this.
2. Semidiscretization of PDEs produces a system of ODEs with n unknowns in the discretization. For such models, it would be computationally advantageous to find a uniform quadratization scheme valid for any n.
3. Many models are not polynomial (e.g., involve fractions or exponential functions), so a polynomialization procedure as in Biocham [31] is missing in QBee.

There are four main contributions of this paper that address these challenges. First, we provide new theorems (with constructive proofs) of existence of quadratizations for different classes of models. These theorems lead us to develop practical algorithms. Second, we extend the algorithms and implementation in the previous version of QBee with extra functionality to

(i) optimally quadratize polynomial systems with time-dependent inputs, (ii) find dimension-agnostic quadratizations for systems with variable dimension (e.g., semidiscretized PDEs), and (iii) polynomialize and then quadratize nonpolynomial terms. This functionality is available in QBee version 0.8 [14]. Third, we demonstrate how this new functionality addresses the three challenges above on models from chemical engineering, rocket combustion, and space weather. These novel quadratizations from QBee outperform reported results in the literature, as we show in the corresponding sections. Fourth, we present a numerical study that shows how these lifting transformations can create better coordinate systems to learn ROMs from data, a key use case of quadratization.

This paper is organized as follows. Section 2 provides background on quadratization of autonomous polynomial ODEs *of fixed dimension.* Section 3 presents new existence results and constructive proofs for quadratization of *nonautonomous* polynomial models. Section 4 then presents new results for uniform quadratizations of families of ODEs arising from semidiscretization of PDEs. Section 5 outlines the QBee algorithm and its new capabilities. In section 6, we showcase the results of our algorithm on models from chemical engineering and rocket combustion. Section 7 demonstrates the advantages of quadratization for learning ROMs on a problem from solar wind velocity prediction. Finally, section 8 offers conclusions and an outlook towards future work.

2. Notation and background.

We start in subsection 2.1 with defining notation and then review known results about the existence of quadratizations in subsection 2.2. These results also provide an upper bound for the order of quadratization, i.e., the maximum required number of variables needed for achieving a lifted quadratic model.

2.1. Notation and definitions.

We denote by $\mathbf{x} = [x_1, x_2, \ldots, x_N]^\top$ an N-dimensional column vector (in either $\mathbb{R}^N$ or $\mathbb{C}^N$), which generically represents the state of a dynamical system (we use different notation for the application problems where states have physical meaning). We denote with $\dot{\mathbf{x}}$ its derivative (usually with respect to time, t, where $t > 0$). Moreover, we denote by $\mathbf{u} = \mathbf{u}(t) \in \mathbb{R}^r$ a generic input vector. We often omit the explicit dependence of t for ease of notation. The symbol $\odot$ denotes the componentwise product of vectors (the Hadamard product), and $\mathbf{x}^\ell$ denotes the ℓth power of $\mathbf{x}$, also understood componentwise. Moreover, $\otimes$ denotes the Kronecker product of vectors, e.g., $[x_1\ x_2]^\top \otimes [y_1\ y_2]^\top = [x_1y_1\ x_1y_2\ x_2y_1\ x_2y_2]^\top$, and $\otimes'$ is the compact Kronecker product, which removes redundant terms (only one term here, $x_1x_2 = x_2x_1$) in the standard Kronecker product, e.g., $[x_1\ x_2]^\top \otimes' [y_1\ y_2]^\top = [x_1y_1\ x_1y_2\ x_2y_2]^\top$. For a matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$, its (i,j)th entry is A_{ij}. The set of nonnegative integers is denoted by $\mathbb{Z}_{\geqslant 0}$.

A product of positive-integer powers of variables is referred to as *monomial* (e.g., x^5y) and the total degree of a monomial is the sum of the powers of the variables appearing in it. A *polynomial* is a sum of monomials, e.g., $x + y^2$. By $\mathbb{C}[\mathbf{x}]$ and $\mathbb{C}[\mathbf{x}, \mathbf{w}]$ we denote the sets of all polynomials with (possibly complex) coefficients in $\mathbf{x}$ and $\mathbf{x}, \mathbf{w}$, respectively. The sets $\mathbb{C}[\mathbf{x}, \mathbf{w}, \mathbf{u}]$ and $\mathbb{C}[\mathbf{x}, \mathbf{w}, \mathbf{u}, \dot{\mathbf{u}}]$ are sets of polynomials defined similarly. Let $p(\mathbf{x})$ be a polynomial; then $\deg_{x_i} p(\mathbf{x})$ denotes the degree of p with respect to x_i, that is, the maximal power of x_i appearing in $p(\mathbf{x})$. Similarly, $\deg p(\mathbf{x})$ denotes the total degree of p, that is, the maximum of the total degrees of the monomials appearing in $p(\mathbf{x})$. For example, $\deg_x(x^5y) = 5$ and $\deg(x^5y) = 6$. The degree of a vector or a matrix is defined as the maximum of the degrees of its entries.

2.2. Quadratization of autonomous polynomial ODEs. We begin with a definition.

Definition 2.1 (quadratization). *Consider a polynomial system of ODEs*

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

where $\mathbf{x} = \mathbf{x}(t) = [x_1(t), \dots, x_N(t)]^\top$ *and* $\mathbf{p}(\mathbf{x}) = [p_1(\mathbf{x}), \dots, p_N(\mathbf{x})]^\top$ *with* $p_1, \dots, p_N \in \mathbb{C}[\mathbf{x}]$. *Then an* ℓ*-dimensional vector of new variables*

$$\mathbf{w} = \mathbf{w}(\mathbf{x}) \quad \in \mathbb{C}[\mathbf{x}]^\ell \tag{2.2}$$

is said to be a quadratization of (2.1) *if there exist vectors* $\mathbf{q}_1(\mathbf{x}, \mathbf{w})$ *and* $\mathbf{q}_2(\mathbf{x}, \mathbf{w})$ *of dimensions* N *and* ℓ, *respectively, with the entries being polynomials of total degree at most two such that*

$$\dot{\mathbf{x}} = \mathbf{q}_1(\mathbf{x}, \mathbf{w}) \quad \text{and} \quad \dot{\mathbf{w}} = \mathbf{q}_2(\mathbf{x}, \mathbf{w}) \tag{2.3}$$

for every $\mathbf{x}$ *solving* (2.1). *The dimension* ℓ *of the vector* $\mathbf{w}$ *is called the order of quadratization. A quadratization of the smallest possible order is called an* optimal quadratization.

Example 2.2. Consider a two-dimensional system

$$\dot{x}_1 = (x_1 + 1)^3 + x_2, \qquad \dot{x}_2 = x_1 + x_2. \tag{2.4}$$

We claim that a new variable $w(\mathbf{x}) = (x_1 + 1)^2$ is a (polynomial) quadratization. Indeed, for the original states x_1 and x_2 we have

$$\dot{x}_1 = (x_1 + 1)w + x_2, \qquad \dot{x}_2 = x_1 + x_2,$$

so we take $\mathbf{q}_1(\mathbf{x}, w) = [(x_1 + 1)w + x_2,\ x_1 + x_2]^\top$. Furthermore,

$$\dot{w} = 2(x_1 + 1)\dot{x}_1 = 2(x_1 + 1)^4 + 2(x_1 + 1)x_2 = 2w^2 + 2(x_1 + 1)x_2,$$

so $q_2(\mathbf{x}, w) = 2w^2 + 2(x_1 + 1)x_2$. Since this quadratization is of order one, it is optimal.

2.2.1. Monomial quadratization. This section considers the case where the new variables are restricted to be of monomial form. Those are useful in applications in synthetic biology [30] and are comparatively easier to find than polynomial terms because the search space is discrete and the problem of finding an optimal quadratization can be phrased as a combinatorial optimization problem.

Definition 2.3 (monomial quadratization). *If all the polynomials* $w_1(\mathbf{x}), \dots, w_\ell(\mathbf{x})$ *are monomials, the quadratization is called a monomial quadratization. If a monomial quadratization of a system has the smallest possible order among all the monomial quadratizations of the system, it is called an optimal monomial quadratization.*

Example 2.4 (continuation of Example 2.2). The quadratization $w(\mathbf{x}) = (x_1 + 1)^2$ introduced in Example 2.2 is not monomial. We next show that $w(\mathbf{x}) = x_1^2$ is a monomial quadratization for the same system (2.4). For x_1 and x_2, we have

$$\dot{x}_1 = x_1^3 + 3x_1^2 + 3x_1 + 1 + x_2 = x_1 w + 3w + 3x_1 + 1 + x_2, \quad \dot{x}_2 = x_1 + x_2,$$

so $\mathbf{q}_1(\mathbf{x}, w) = [x_1 w + 3w + 3x_1 + 1 + x_2,\ x_1 + x_2]^\top$. Furthermore,

$$\begin{aligned}\dot{w} &= 2(x_1+1)\dot{x}_1 = 2x_1^4 + 8x_1^3 + 12x_1^2 + 8x_1 + 2 + (x_1+1)x_2 \\ &= 2w^2 + 8x_1 w + 12w + 8x_1 + 2 + (x_1+1)x_2,\end{aligned}$$

so $q_2(\mathbf{x}, w) = 2w^2 + 8x_1 w + 12w + 8x_1 + 2 + (x_1+1)x_2$. Therefore, $w(\mathbf{x}) = x_1^2$ is an optimal monomial quadratization of (2.4). Here, both the monomial quadratization and the polynomial quadratization (cf. Examples 2.6 and 2.7) are optimal, yet the dynamics of the auxiliary state w are a bit more involved in the monomial case.

The next theorem ensures that systems of polynomial ODEs are always guaranteed to have an exact quadratic representation in a different, possibly augmented, set of variables.

Theorem 2.5 ([16, Theorem 1]). *For every ODE system of the form (2.1), there exists a monomial quadratization. Furthermore, if* $d_i := \deg_{x_i} \mathbf{p}(\mathbf{x})$, *then the order of an optimal monomial quadratization does not exceed* $\Pi_{i=1}^N (d_i + 1)$.

Monomial quadratizations have been actively studied in the literature. We summarize a few known results about the optimality, e.g., minimal order, of such quadratizations:

1. In [30, Theorem 2] it is shown that finding an optimal monomial quadratization is an NP-hard problem even if the polynomials are represented using dense encoding (that is, by all the coefficients up to certain degree).
2. One natural approach (used, e.g., in [30]) to finding an optimal monomial quadratization is to take an explicit but large monomial quadratization constructed in the standard proof of Theorem 2.5 and search for the smallest subset of it which is a quadratization itself. However, this approach can lead to a nonoptimal quadratization, as illustrated in [13, Example 3].
3. If one allows the new variables to be Laurent monomials (that is, have negative integer degrees), one can obtain a better upper bound for the number of new variables: it is always sufficient to take as many new variables as there are monomials on the right-hand side of the system [13, Proposition 1]. However, we are not aware of any algorithm for finding an optimal Laurent monomial quadratization.

2.2.2. Polynomial quadratization. Despite the appeal of monomial quadratizations in certain communities, it is natural to ask whether one may be able to obtain a substantially more concise quadratization if *arbitrary polynomials* are allowed for the new variables (as in Definition 2.1). Although, in the examples above, allowing arbitrary polynomials did not result in a lower-order quadratization, the following example from [3, section 4] shows that this is not always the case, and the difference between optimal monomial and arbitrary quadratizations may be significant.

Example 2.6 (lower-order quadratization using arbitrary polynomials). Consider the scalar ODE

$$\dot{x} = (x+1)^k,$$

where k is a positive integer. A simple combinatorial argument [3, section 4] shows that at least $\frac{\sqrt{8k+9}-5}{2}$ monomial (that is, powers of x) variables are required for a monomial

quadratization. Since k can be arbitrarily large, this is concerning. In contrast, it is always possible to quadratize the ODE by adding one new variable $w(x) := (x+1)^{k-1}$ so that

$$\begin{aligned}\dot{x} &= (x+1)w,\\ \dot{w} &= \dot{x}(k-1)(x+1)^{k-2} = (k-1)(x+1)^{2k-2} = (k-1)w^2.\end{aligned}$$

Although the new variable w above is per definition not a monomial, it can be thought of as a "shifted monomial." In [3, Theorem 3.1], the authors state that if it is possible to quadratize a scalar ODE by a single new variable, then the new variable can always be taken to be such a shifted monomial. However, if more new variables are added, the set of potentially useful new variables becomes richer, as the next example [3, Theorem 3.2] shows.

Example 2.7 (lower-order quadratization using arbitrary polynomials, continued). Consider the scalar ODE of degree six:

$$\dot{x} = x^6 + x^4 + x^3. \tag{2.5}$$

One can show using [3, Theorem 3.1] that (2.5) cannot be quadratized using a single new variable. In fact, if monomial new variables are used, at least three of them are needed [3, Lemma 5.7]. However, one can quadratize (2.5) using just two new polynomial variables $w_1(x) := x^3$ and $w_2(x) := x^5 + \frac{5}{8}x^2$ as follows:

$$\begin{aligned}\dot{x} &= w_1^2 + xw_1 + w_1,\\ \dot{w}_1 &= 3\left(w_1w_2 + xw_2 + \frac{3}{8}w_2 - \frac{5}{8}w_1 - \frac{15}{64}x^2\right),\\ \dot{w}_2 &= 5\left(w_2^2 + w_1w_2 - \frac{9}{64}xw_1 - \frac{3}{8}w_2 + \frac{15}{64}x^2\right).\end{aligned}$$

The previous examples show that there is merit in searching for polynomial quadratizations to limit the growth of additional variables needed to achieve quadratic form.

3. Quadratization of polynomial ODEs with inputs. Many engineering systems are forced with external inputs or disturbances. This section presents new theoretical results for quadratization of polynomial ODEs with external forcing, which then lay the foundation for efficient algorithms. We are interested in bringing polynomial ODEs into quadratic-bilinear[1] (QB) form, where

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{H}(\mathbf{x}\otimes\mathbf{x}) + \sum_{i=1}^{r}\mathbf{N}_i\mathbf{x}u_i + \mathbf{B}\mathbf{u}, \tag{3.1}$$

and $\mathbf{A} \in \mathbb{R}^{N\times N}, \mathbf{H} \in \mathbb{R}^{N\times N^2}$, $\mathbf{N}_i \in \mathbb{R}^{N\times N}$ for $1 \leqslant i \leqslant r$, and $\mathbf{B} \in \mathbb{R}^{N\times r}$. QB systems have seen great interest in the systems and control community and are therefore an appealing target structure for the proposed quadratization methods; see the introduction for further discussion. We separate two cases for quadratization: subsection 3.1 covers polynomial ODEs with differentiable inputs, and subsection 3.2 presents quadratization results for systems where the inputs $\mathbf{u}(t)$ are not differentiable.

[1] In the systems and control literature, the terms $\mathbf{N}_i\mathbf{x}u_i$ above are called *bilinear*—a subclass of *control-affine systems*—as they are linear in the state $\mathbf{x}$ and linear in the input/control $\mathbf{u}$.

3.1. Systems with differentiable inputs. We begin this section with a definition that extends Definition 2.1 to the nonautonomous polynomial case where the inputs are differentiable. Since we aim at the QB form (3.1), our definitions are different from the ones in [18], where the input variables are ignored when counting the degree.

Definition 3.1 (quadratization of nonautonomous polynomial ODEs). *Consider the system*

$$\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x}, \mathbf{u}), \tag{3.2}$$

where $\mathbf{x} = \mathbf{x}(t) = [x_1(t), \dots, x_N(t)]^\top$ *are the states,* $\mathbf{u} = \mathbf{u}(t) = [u_1(t), \dots, u_r(t)]^\top$ *denote the inputs, and* $p_1, \dots, p_N \in \mathbb{C}[\mathbf{x}, \mathbf{u}]$. *Then an* ℓ*-dimensional vector of new variables*

$$\mathbf{w} = \mathbf{w}(\mathbf{x}, \mathbf{u}) \quad \in \mathbb{C}[\mathbf{x}, \mathbf{u}]^\ell \tag{3.3}$$

is said to be a quadratization of (3.2) *if there exist vectors* $\mathbf{q}_1(\mathbf{x}, \mathbf{w}, \mathbf{u}, \dot{\mathbf{u}})$ *and* $\mathbf{q}_2(\mathbf{x}, \mathbf{w}, \mathbf{u}, \dot{\mathbf{u}})$ *of dimensions* N *and* ℓ*, respectively, with the entries being polynomials of total degree at most two such that*

$$\dot{\mathbf{x}} = \mathbf{q}_1(\mathbf{x}, \mathbf{w}, \mathbf{u}, \dot{\mathbf{u}}) \quad \text{and} \quad \dot{\mathbf{w}} = \mathbf{q}_2(\mathbf{x}, \mathbf{w}, \mathbf{u}, \dot{\mathbf{u}}) \tag{3.4}$$

for every $\mathbf{x}$ *solving* (3.2). *The number* ℓ *is called the order of quadratization. A quadratization of the smallest possible order is called an optimal quadratization. A monomial quadratization of nonautonomous systems is defined similarly as in Definition 2.3.*

Example 3.2. Consider a nonautonomous scalar ODE $\dot{x} = x + x^2 u$. We claim that $w(x, u) = xu$ is a quadratization. Indeed, the corresponding quadratic system is

$$\dot{x} = x + xw, \quad \dot{w} = xu' + (x + x^2 u)u = xu' + xu + w^2,$$

so $q_1(x, w, u, \dot{u}) = x + xw$ and $q_2(x, w, u, \dot{u}) = xu' + xu + w^2$. This quadratization is optimal and monomial.

The next theorem addresses the existence of a quadratization for nonautonomous polynomial nonlinear ODEs.

Theorem 3.3. *Every polynomial ODE system of the form* (3.2) *has a monomial quadratization assuming that the inputs are differentiable. Moreover, the order of the optimal monomial quadratization does not exceed* $\Pi_{i=1}^{N+r}(d_i + 1)$*, where*

$$d_i := \deg_{x_i} \mathbf{p}(\mathbf{x}, \mathbf{u}) \text{ for } 1 \leqslant i \leqslant N \quad \text{and} \quad d_{N+i} := \deg_{u_i} \mathbf{p}(\mathbf{x}, \mathbf{u}) \text{ for } 1 \leqslant i \leqslant r$$

are the maximal degrees of the polynomials in the state and inputs, respectively.

Proof. We consider the system (3.2) and construct a quadratization for it. We consider the set of monomials

$$\begin{aligned}\mathcal{M} := \{m(\mathbf{x}, \mathbf{u}) \,|\, & m(\mathbf{x}, \mathbf{u}) \text{ is a monomial in } \mathbf{x}, \mathbf{u} \text{ s.t.} \\ & \forall\, 1 \leqslant i \leqslant N\colon \deg_{x_i} m(\mathbf{x}, \mathbf{u}) \leqslant d_i \text{ and } \forall\, 1 \leqslant i \leqslant r\colon \deg_{u_i} m(\mathbf{x}, \mathbf{u}) \leqslant d_{N+i}\}\end{aligned}$$

and claim that $\mathcal{M}$ is a quadratization of the original system (3.2). To prove this, we consider an arbitrary $m(\mathbf{x},\mathbf{u}) \in \mathcal{M}$ and examine the monomials on the right-hand side of its derivative $\dot{m}(\mathbf{x},\mathbf{u})$, where

$$\dot{m}(\mathbf{x},\mathbf{u}) = \sum_{i=1}^{N} p_i(\mathbf{x},\mathbf{u}) \frac{\partial m(\mathbf{x},\mathbf{u})}{\partial x_i} + \sum_{j=1}^{r} \dot{u}_j \frac{\partial m(\mathbf{x},\mathbf{u})}{\partial u_i},$$

which follows by the chain rule. By construction of $\mathcal{M}$, every monomial in $p_i(\mathbf{x},\mathbf{u})$ for $1 \leqslant i \leqslant N$ belongs to $\mathcal{M}$, so every monomial in the first sum can be written as $m_0(\mathbf{x},\mathbf{u})\frac{m(\mathbf{x},\mathbf{u})}{x_i}$, where x_i appears in m and $m_0 \in \mathcal{M}$. Since $\frac{m}{x_i} \in \mathcal{M}$, this product is a quadratic expression in $\mathcal{M}$. Furthermore, every monomial in the second sum is of the form $\dot{u}_i \frac{m(\mathbf{x},\mathbf{u})}{u_i}$, where u_i appears in m. Since $\frac{m(\mathbf{x},\mathbf{u})}{u_i} \in \mathcal{M}$, this monomial can also be written as a quadratic expression in $\mathcal{M}$. The claim about the size of the optimal quadratization follows from the fact that $|\mathcal{M}| = \Pi_{i=1}^{N+r}(d_i+1)$. ■

The external inputs $\mathbf{u}(t)$ may not always be differentiable, as step inputs, ramp inputs, pulses, and many other nondifferentiable inputs are used in engineering. While those can often be smoothed, the next section considers the case where the inputs are not differentiable.

3.2. Systems with nondifferentiable inputs.

We consider systems with nondifferentiable inputs $\mathbf{u}(t)$ for which we define a new quadratization. While it is not always possible to relax the differentiability condition on the inputs in Theorem 3.3 (see Example 3.10 and subsection 6.2 for examples), we characterize the cases when it is possible. We start with a corresponding definition.

Definition 3.4 (input-free quadratization of nonautonomous polynomial ODEs). *In the notation of Definition 3.1, an ℓ-dimensional vector of new variables*

$$\mathbf{w} = \mathbf{w}(\mathbf{x}) \quad \in \mathbb{C}[\mathbf{x}]^{\ell} \tag{3.5}$$

is said to be an input-free quadratization of (3.2) if there exist vectors $\mathbf{q}_1(\mathbf{x},\mathbf{w},\mathbf{u})$ and $\mathbf{q}_2(\mathbf{x},\mathbf{w},\mathbf{u})$ of dimensions N and ℓ, respectively, with the entries being polynomials of total degree at most two such that

$$\dot{\mathbf{x}} = \mathbf{q}_1(\mathbf{x},\mathbf{w},\mathbf{u}) \quad \text{and} \quad \dot{\mathbf{w}} = \mathbf{q}_2(\mathbf{x},\mathbf{w},\mathbf{u}). \tag{3.6}$$

The number ℓ is called the order of quadratization. A quadratization of the smallest possible order is called an optimal quadratization.

Remark 3.5. The key difference between Definition 3.1 and Definition 3.4 is that $\dot{\mathbf{u}}$ does not appear in the right-hand side terms of the resulting system (that is, in the vectors $\mathbf{q}_1$ and $\mathbf{q}_2$ of quadratic polynomials). This, in turn, implies that $\mathbf{w}$ cannot depend on $\mathbf{u}$. For example, the quadratization from Example 3.2 is not input-free (and in fact the model does not admit an input-free quadratization; see also Example 3.10).

Example 3.6. Consider a two-dimensional system describing the Duffing oscillator:

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = -\alpha x_1 - \delta x_2 - \beta x_1^3 + u.$$

We claim that $w(\mathbf{x}) = x_1^2$ is an input-free (monomial and optimal) quadratization. Indeed, the corresponding quadratic system is

$$\dot{x}_1 = x_2, \quad \dot{x}_2 = -\alpha x_1 - \delta x_2 - \beta x_1 w + u, \quad \dot{w} = 2x_1 x_2,$$

so $\mathbf{q}_1(\mathbf{x}, w, u) = [x_2, -\alpha x_1 - \delta x_2 - \beta x_1 w + u]^\top$ and $q_2(\mathbf{x}, w, u) = 2x_1 x_2$. The quadratic system above does not involve any derivatives of u and, thus, does not add any additional constraints on the input function.

One nice feature of input-free quadratizations is that they produce the QB structure, as the following lemma shows.

Lemma 3.7. *Assume that the original system (3.2) is input-affine, that is, of the form*

$$\dot{\mathbf{x}} = \mathbf{p}_0(\mathbf{x}) + \sum_{i=1}^{r} \mathbf{p}_i(\mathbf{x}) u_i. \tag{3.7}$$

If there exists an input-free quadratization (3.6) with variables $\mathbf{w}$, then this quadratization is quadratic-bilinear.

Proof. Since every $\dot{x}_i$ is input-affine, the same is true for every $\dot{w}_j$. Since the inputs do not appear in the new variables and $w_1, \ldots, w_\ell$ is a quadratization of the original system, the system (3.6) has at most a quadratic right-hand side which does not involve products of inputs. This shows that it is quadratic-bilinear. ∎

Unlike general quadratizations of nonautonomous systems, an input-affine quadratization may not always exist for a given input-affine polynomial system (see Example 3.10). In the remainder of this section, we present additional theoretical results on the existence of an input-free quadratization for input-affine systems. We start with a simple class of systems admitting an input-free quadratization.

Lemma 3.8. *Assume that the system (3.7) is* polynomial-bilinear*, that is, the total degree of $\mathbf{p}_i(\mathbf{x})$ in (3.7) is at most one for every $1 \leqslant i \leqslant r$. Then there is an input-free quadratization for (3.7).*

Proof. Let $d := \deg \mathbf{p}_0(\mathbf{x})$ and assume that the total degree of $\mathbf{p}_i(\mathbf{x})$ in (3.7) is at most one for every $1 \leqslant i \leqslant r$. We consider the set of monomials

$$\mathcal{M} = \{m(\mathbf{x}) \mid m(\mathbf{x}) \text{ is a monomial in } \mathbf{x} \text{ and } \deg m(\mathbf{x}) \leqslant d\}$$

and claim that $\mathcal{M}$ is an input-free quadratization of the polynomial-bilinear system (3.7). To prove this, consider any $m(\mathbf{x}) \in \mathcal{M}$, which has derivative

$$\dot{m}(\mathbf{x}) = \sum_{i=1}^{N} p_{0,i}(\mathbf{x}) \frac{\partial m(\mathbf{x})}{\partial x_i} + \sum_{j=1}^{r} u_j \sum_{i=1}^{N} p_{j,i}(\mathbf{x}) \frac{\partial m(\mathbf{x})}{\partial x_i}.$$

We separately consider the terms in the first and second sums:

1. In the first sum, every monomial is the form $\frac{m(\mathbf{x})}{x_i} m_0(\mathbf{x})$, where $m_0(\mathbf{x})$ is a monomial of $p_{0,i}(\mathbf{x})$. It is the product of two elements of $\mathcal{M}$: $\frac{m(\mathbf{x})}{x_i}$ and $m_0(\mathbf{x})$.

2. In the second sum, every monomial is of the form $\frac{m(\mathbf{x})}{x_i} m_0(\mathbf{x}) u_j$, where $m_0(\mathbf{x})$ is a monomial from $p_{i,j}(\mathbf{x})$. Since $\deg p_{j,i}(\mathbf{x}) \leqslant 1$, we have $\deg m_0 \leqslant 1$, so $\frac{m(\mathbf{x})}{x_i} m_0(\mathbf{x}) \in \mathcal{M}$.

Together, this shows that every monomial on the right-hand side of the equation for $m(\mathbf{x})$ is at most quadratic in $\mathcal{M}$, so $\mathcal{M}$ is indeed an input-free quadratization. ∎

More generally, it turns out that the existence of an input-free quadratization can be characterized via the properties of certain linear differential operators associated with the inputs $\mathbf{u}(t)$. This characterization turns out to be a generalization of the classical concept of locally finite derivation, as explained in Remark 3.12 below. The next proposition provides this existence result.

Proposition 3.9. *Consider an input-affine system of the form (3.7). We introduce r differential operators:*

$$D_i := \mathbf{p}_i(\mathbf{x})^\top \cdot \frac{\partial}{\partial \mathbf{x}}, \quad 1 \leqslant i \leqslant r,$$

where $\frac{\partial}{\partial \mathbf{x}} = \left[\frac{\partial}{\partial x_1}, \dots, \frac{\partial}{\partial x_N}\right]^\top$. Let $\mathcal{A}$ be a subalgebra generated by $D_1, \dots, D_r$ in the algebra $\mathbb{C}\left[\mathbf{x}, \frac{\partial}{\partial \mathbf{x}}\right]$ of all polynomial differential operators in $\mathbf{x}$. Then there is an input-free quadratization for (3.7) if and only if

$$\dim\{A(x_i) \mid A \in \mathcal{A}\} < \infty \quad \text{for every } 1 \leqslant i \leqslant N. \tag{3.8}$$

Before proving the proposition, let us illustrate it with two examples.

Example 3.10 (infinite dimension). Consider the scalar input-affine ODE with a single input:

$$\dot{x} = x^2 u.$$

In the notation of Proposition 3.9, we have $D_1 = x^2 \frac{\partial}{\partial x}$. Then the algebra $\mathcal{A}$ is spanned by $D_1, D_1^2, \dots$ and its application to x yields

$$D_1(x) = x^2, \quad D_1^2(x) = 2x^3, \quad D_1^3(x) = 6x^4, \quad \dots.$$

Thus, $\dim\{A(x) \mid A \in \mathcal{A}\} = \infty$ and the proposition implies that there is no input-free quadratization for this ODE.

Example 3.11 (finite dimension). We consider a modification of the previous example:

$$\dot{x}_1 = x_1 + x_1 u, \quad \dot{x}_2 = x_1^2 u. \tag{3.9}$$

In this case $D_1 = x_1 \frac{\partial}{\partial x_1} + x_1^2 \frac{\partial}{\partial x_2}$. We have $D_1(x_1) = x_1$ and $D_1(x_2) = x_1^2$. Therefore, $\{A(x_1) \mid A \in \mathcal{A}\} = \operatorname{span}\{x_1\}$ and $\{A(x_2) \mid A \in \mathcal{A}\} = \operatorname{span}\{x_2, x_1^2\}$. Therefore, Proposition 3.9 implies that (3.9) admits an input-free quadratization.

The condition in Proposition 3.9 is related to locally finite derivations, as we explain next.

Remark 3.12 (algorithmic decidability of condition (3.8)). For the special case $r = 1$, condition (3.8) is equivalent to saying that the operator D_1 is *locally finite*. A differential

operator D on a polynomial ring in variables $x_1, \dots, x_N$ is called locally finite if the dimension of $\operatorname{span}\{x_i, D(x_i), D^2(x_i), \dots\}$ is finite for every i. Locally finite derivations appear in different contexts [21], but the problem of determining algorithmically whether a given derivation is locally finite has been solved only for the bivariate case [21, section 4] and remains an open problem in the general case.

Proof of Proposition 3.9. $\Leftarrow$: Assume that $\{A(x_i) \mid A \in \mathcal{A}\}$ is finite-dimensional for every $1 \leqslant i \leqslant N$. For each of these spaces we choose a basis, and let $\mathcal{M}$ be the union of these bases. Consider any $p(\mathbf{x}) \in \mathcal{M} \cup \{x_1, \dots, x_N\}$. Then

$$\dot{p}(\mathbf{x}) = \sum_{j=1}^{N} p_{0,j}(\mathbf{x}) \frac{\partial p}{\partial x_i} + \sum_{i=1}^{r} u_i D_i(p(\mathbf{x})). \tag{3.10}$$

Since $p(\mathbf{x})$ is of the form $A(x_j)$ for some $A \in \mathcal{A}$ and $1 \leqslant j \leqslant N$, $D_i(p(\mathbf{x}))$ is also of this form. Thus, $D_i(p(\mathbf{x}))$ can be written as a linear combination of elements of $\mathcal{M} \cup \{x_1, \dots, x_N\}$. Therefore, if we collect equations (3.10) for $p(\mathbf{x})$ ranging over $\mathcal{M} \cup \{x_1, \dots, x_N\}$, we obtain an ODE system in the variables $\{x_1, \dots, x_N\} \cup \mathcal{M}$, and this system satisfies the requirements of Lemma 3.8. Therefore, this system has an input-free quadratization, and, thus, the same is true for the original system.

$\Rightarrow$: Assume that (3.7) has an input-free quadratization with the ℓ-dimensional vector of new variables $\mathbf{w}(\mathbf{x})$. Let $\mathcal{X} := \operatorname{span}\{x_1, \dots, x_N, w_1, \dots, w_\ell\}$. We prove by induction on an integer $h \in \mathbb{Z}_{\geqslant 0}$ that, for every $1 \leqslant i_1, \dots, i_h \leqslant r$ and $1 \leqslant j \leqslant N$, the element $[D_{i_1} \dots D_{i_h}](x_j)$ belongs to $\mathcal{X}$. The base case $h = 0$ is true. Assume that the statement has been proven for h, and consider any $1 \leqslant i_1, \dots, i_h \leqslant r$ and $1 \leqslant j \leqslant N$ and let $p(\mathbf{x}) = [D_{i_1} \dots D_{i_h}](x_j)$. Since $p(\mathbf{x}) \in \mathcal{X}$ by the induction hypothesis, $\dot{p}(\mathbf{x})$ can be written as at most quadratic polynomial in $\mathcal{X}$ and $\mathbf{u}$. From the expression (3.10) one can observe that this implies that, for every $1 \leqslant s \leqslant r$, $D_s(p(\mathbf{x})) = [D_s D_{i_1} \dots D_{i_h}](x_j) \in \mathcal{X}$. Since this holds for any $1 \leqslant i_1, \dots, i_h \leqslant r$ and $1 \leqslant j \leqslant N$, this proves the induction step. Since $\mathcal{X}$ is finite-dimensional, the dimension of $\{A(x_i) \mid A \in \mathcal{A}\}$ must be finite for every $1 \leqslant i \leqslant N$. ∎

Although it may be complicated to verify the general condition (3.8) from Proposition 3.9, there is an important special case when it can be easily verified.

Lemma 3.13. *In the notation of Proposition 3.9, if, for every $1 \leqslant i \leqslant r$ and $1 \leqslant j \leqslant N$,*

$$D_i(x_j) \in \mathbb{C}[x_1, \dots, x_{j-1}]$$

holds, then the condition $\dim\{A(x_i) \mid A \in \mathcal{A}\} < \infty$ is fulfilled.

Proof. The assumption that, for every $1 \leqslant i \leqslant r$ and $1 \leqslant j \leqslant N$, $D_i(x_j)$ depends only on x_k with $k > j$ implies that, for every $1 \leqslant i_1, \dots, i_N \leqslant N$, the product $D_{i_1} \dots D_{i_N}$ sends each of $x_1, \dots, x_N$ to zero. Therefore, the dimension of each $\{A(x_i) \mid A \in \mathcal{A}\}$ does not exceed $1 + r + \dots + r^{N-1}$. ∎

4. Dimension-agnostic quadratizations of families of ODE systems. This section presents a new quadratization method that is applicable to families of ODE systems of variable dimension, for which it produces a dimension-agnostic quadratization. The most natural use case is ODEs that are derived via semidiscretization of PDEs, i.e., where the symbolic form

of the nonlinear terms stays the same but the discretization dimension n of the system can be varied. The current way to do this is to compute a quadratization for every n separately. However, this is time-consuming for large n and appears unnecessary. The methods derived in this section improve upon this by exploiting similar structures in these systems. We note that while semidiscretized PDEs are presented in the first example, the methods herein apply to a wider class of problems than that, as we see below. To motivate the setup and subsequent definitions, let us start with an example.

Example 4.1. Consider a special case of a dimensionless scalar PDE used in traffic flow modeling [26, section 12.6]:

$$\frac{\partial \rho}{\partial t}(t,\xi) = \rho(t,\xi) + \rho^2(t,\xi)\frac{\partial \rho(t,\xi)}{\partial \xi}, \qquad \rho(t,0) = 0, \quad \rho(t,1) = 1,$$

where $\rho(t,\xi)$ is the traffic density, t denotes times, and $\xi \in [0,1]$ is the spatial variable. Let the initial condition be a square-integrable function $\rho(0,\xi) = \varphi(\xi)$. We discretize the spatial domain uniformly and denote the density at the nodal values as $x_i^{[n]}(t) = \rho(t, i/(n+1))$ for $1 \le i \le n$. This defines the n-dimensional state vector $\mathbf{x}^{[n]}(t) = [x_1^{[n]}(t), \dots, x_n^{[n]}(t)]^\top$. We approximate the spatial derivative by a backward first-order differencing formula as

$$\frac{\partial \rho}{\partial \xi}(t, i/(n+1)) \approx \frac{\rho(t, i/(n+1)) - \rho(t, (i-1)/(n+1))}{\Delta \xi},$$

where $\Delta\xi = 1/(n+1)$, which then yields the n-dimensional system of ODEs

$$\dot{\mathbf{x}}^{[n]} = \mathbf{x}^{[n]} + (\mathbf{x}^{[n]})^2 \odot (\mathbf{D}\mathbf{x}^{[n]}), \tag{4.1}$$

where $\odot$ denotes the Hadamard (or elementwise) product and where

$$\mathbf{D} = \frac{1}{\Delta\xi}\begin{bmatrix} 1 & 0 & & & & \\ -1 & 1 & 0 & & & \\ 0 & -1 & 1 & & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 1 & 0 \\ & & & 0 & -1 & 1 \end{bmatrix} \in \mathbb{R}^{n\times n}. \tag{4.2}$$

Our goal is to quadratize (4.1) for arbitrary n; for brevity we drop the superscript $[n]$ in what follows. Consider the vectors $\mathbf{w}_1 = \mathbf{x}^2$ and $\mathbf{w}_2 = \mathbf{x} \odot (\mathbf{S}\mathbf{x})$, where $\mathbf{S}$ is the lower shift matrix (with ones on the subdiagonal and zeros elsewhere), i.e., $\mathbf{S}\mathbf{x} = [0,\, x_1,\, \dots,\, x_{n-1}]^\top$. We claim that the nonzero entries of $\mathbf{w}_1$ and $\mathbf{w}_2$ are a quadratization of (4.1) for any n. The original system can be now written as

$$\dot{\mathbf{x}} = \mathbf{x} + \mathbf{w}_1 \odot (\mathbf{D}\mathbf{x}). \tag{4.3}$$

Furthermore, using $\mathbf{Dx} = \frac{1}{\Delta\xi}(\mathbf{x} - \mathbf{Sx})$ we have

$$
\begin{aligned}
\dot{\mathbf{w}}_1 &= 2\mathbf{x}(\mathbf{x} + \mathbf{x}^2 \odot (\mathbf{Dx})) = 2\mathbf{w}_1 + \frac{2}{\Delta\xi}\mathbf{w}_1(\mathbf{w}_1 - \mathbf{w}_2), \\
\dot{\mathbf{w}}_2 &= \left(\mathbf{x} + \mathbf{x}^2 \odot \frac{1}{\Delta\xi}(\mathbf{x} - \mathbf{Sx})\right)(\mathbf{Sx}) + \mathbf{x}\left(\mathbf{Sx} + (\mathbf{Sx})^2 \odot \frac{1}{\Delta\xi}(\mathbf{Sx} - \mathbf{S}^2\mathbf{x})\right) \\
&= 2\mathbf{w}_2 + \frac{1}{\Delta\xi}(\mathbf{w}_1 \odot \mathbf{w}_2 - \mathbf{w}_2^2 + \mathbf{w}_2 \odot (\mathbf{Sw}_1) - \mathbf{w}_2 \odot (\mathbf{Sw}_2)).
\end{aligned} \tag{4.4}
$$

Therefore, $\mathbf{w}_1 = \mathbf{w}_1^{[n]}$ and $\mathbf{w}_2 = \mathbf{w}_2^{[n]}$ is a quadratization of (4.1) for every n, which we refer to as a *dimension-agnostic* quadratization (see Definition 4.4 for a formal definition). Note that here we can solve for $\mathbf{w}_1$ and $\mathbf{w}_2$ and then recover $\mathbf{x} = \sqrt{\mathbf{w}_1}$, so the state equations are decoupled and we do not need to solve the equation for $\dot{\mathbf{x}}$.

We highlight that there are two types of quadratizing variables in the above example: *uncoupled* variables $\mathbf{w}_1$, where each variable depends only on one of the x_i's, and *coupled* variables $\mathbf{w}_2$, where each variable involves two different x_i and x_j so that the ith and jth equations are coupled, which is the case when $|i - j| = 1$.

We next provide a formal definition for the class of systems we consider. If those are derived from PDEs, then n_d is the number of dependent variables in the polynomial PDE system. For example, $n_d = 1$ for Example 4.1 and $n_d = 2$ for the solar wind example (7.3) in section 7 which was lifted with one additional variable to polynomial form. Moreover, n is the dimension of spatial discretization.

Definition 4.2 (family of linearly coupled polynomial ODEs). *Let $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x}) \in \mathbb{C}[\mathbf{x}]^{n_d}$ be $n_d + 1$ vectors with entries polynomial in $\mathbf{x} = [x_1, \dots, x_{n_d}]^\top$. For such $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x})$ we assign a family of linearly coupled ODE systems as follows. Consider a positive integer n and matrices $\mathbf{D}_1, \dots, \mathbf{D}_{n_d} \in \mathbb{C}^{n \times n}$. We construct an ODE system for $N = n_d \cdot n$ variables $x_{i,j}^{[n]}$ with $1 \leqslant i \leqslant n$ and $1 \leqslant j \leqslant n_d$. A vector of these variables is denoted by $\mathbf{x}^{[n]}$, and we define $\mathbf{x}_i^{[n]} = [x_{i,1}^{[n]}, \dots, x_{i,n_d}^{[n]}]^\top$ and $\mathbf{x}_{*,j}^{[n]} = [x_{1,j}^{[n]}, \dots, x_{n,j}^{[n]}]^\top$. Then the ODE system defined by the polynomial vectors $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x})$ and constant matrices $\mathbf{D}_1, \dots, \mathbf{D}_{n_d}$ is*

$$
\dot{\mathbf{x}}_i^{[n]} = \mathbf{p}_0(\mathbf{x}_i^{[n]}) + \sum_{j=1}^{n_d} \mathbf{p}_j(\mathbf{x}_i^{[n]})(\mathbf{D}_j \mathbf{x}_{*,j}^{[n]})_i, \quad i = 1, 2, \dots, n. \tag{4.5}
$$

We refer to the ODE system (4.5) as $\mathcal{F}^{[n]}(\mathbf{p}_0, \dots, \mathbf{p}_n, \mathbf{D}_1, \dots, \mathbf{D}_{n_d})$.

Example 4.3 (vectors $\mathbf{p}_0, \dots, \mathbf{p}_{n_d}$ for (4.1)). Recall the system (4.1) where $n_d = 1$. We can rewrite this system into the form of (4.5) with $\mathbf{p}_0(x) = [x]$ and $\mathbf{p}_1(x) = [x^2]$ and as well as the matrix $\mathbf{D}_1 = \mathbf{D}$ from (4.2).

Since the infinite family of ODE systems (4.5) is defined by a finite number of variables, that is, by $n_d + 1$ polynomial vectors $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x})$, it is natural to define an entire family of quadratizations also by some finite amount of variables.

Definition 4.4 (dimension-agnostic quadratization). *Consider a family of linearly coupled ODEs defined by $n_d + 1$ polynomial vectors $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x})$ in $\mathbf{x} = [x_1, \dots, x_{n_d}]^\top$, and consider an ℓ-dimensional vector $\mathbf{w}_1(\mathbf{x}) \in \mathbb{C}[\mathbf{x}]^\ell$ together with an L-dimensional vector $\mathbf{w}_2(\mathbf{x}, \tilde{\mathbf{x}}) \in$*

$\mathbb{C}[\mathbf{x}, \tilde{\mathbf{x}}]^L$, *where* $\tilde{\mathbf{x}} = [\tilde{x}_1, \dots, \tilde{x}_{n_d}]^\top$ *are formal variables used as placeholders for the coupled variables (to be made precise below). For every integer* n *and matrices* $\mathbf{D}_1, \dots, \mathbf{D}_{n_d} \in \mathbb{C}^{n\times n}$, *we define a set*

$$\begin{aligned}\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}_1, \dots, \mathbf{D}_{n_d}) := &\{\mathbf{w}_1(\mathbf{x}_i^{[n]}) \mid 1 \leqslant i \leqslant n\} \\ &\cup \{\mathbf{w}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]}) \mid 1 \leqslant i_0 \neq i_1 \leqslant n, \text{ and } \exists k\colon (\mathbf{D}_k)_{i_0,i_1} \neq 0\}.\end{aligned} \tag{4.6}$$

Then we say that $\mathbf{w}_1$ *and* $\mathbf{w}_2$ *are a dimension-agnostic quadratization of the family if, for every integer* n *and matrices* $\mathbf{D}_1, \dots, \mathbf{D}_{n_d} \in \mathbb{C}^{n\times n}$, $\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}_1, \dots, \mathbf{D}_{n_d})$ *is a quadratization of* $\mathcal{F}^{[n]}(\mathbf{p}_0, \dots, \mathbf{p}_{n_d}, \mathbf{D}_1, \dots, \mathbf{D}_{n_d})$.

Example 4.5 (vectors $\mathbf{w}_1$ and $\mathbf{w}_2$ for (4.1)). Recall that in Example 4.1 we obtained a quadratization with the vectors

$$\mathbf{w}_1^{[n]} = [(x_1^{[n]})^2, \dots, (x_n^{[n]})^2]^\top \quad \text{and} \quad \mathbf{w}_2^{[n]} = [0, x_1^{[n]}x_2^{[n]}, x_2^{[n]}x_3^{[n]}, \dots, x_{n-1}^{[n]}x_n^{[n]}]^\top.$$

We claim that these new variables are $\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D})$, where $\mathbf{w}_1 = x^2$ and $\mathbf{w}_2 = x\cdot\tilde{x}$ in this case are scalars. Indeed, $\mathbf{w}_1 = x^2$ gives rise to $\mathbf{w}_1^{[n]}$ by (4.6). Since the off-diagonal nonzero entries of $\mathbf{D}$ (4.2) are at the $(i, i-1)$ locations for $2 \leqslant i \leqslant n$, so the new variables in (4.6) coming from $\mathbf{w}_2 = x \cdot \tilde{x}$ are obtained by setting $x = x_i^{[n]}$ and $\tilde{x} = x_{i-1}^{[n]}$ for every $2 \leqslant i \leqslant n$ yielding the nonzero entries of $\mathbf{w}_2^{[n]}$. Assume, for example, that we had periodic boundary conditions, and thus (using $\Delta\xi = \frac{1}{n}$)

$$\mathbf{D}_{\text{per}} = \frac{1}{\Delta\xi}\begin{bmatrix} 1 & 0 & & & & -1 \\ -1 & 1 & 0 & & & \\ 0 & -1 & 1 & & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 1 & 0 \\ & & & 0 & -1 & 1 \end{bmatrix} \in \mathbb{R}^{n\times n},$$

which has an additional nonzero off-diagonal entry (the $(1, n)$th entry). This results in

$$\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}_{\text{per}}) = \mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}) \cup \{x_1^{[n]}x_n^{[n]}\}.$$

Furthermore, it is possible to show that $\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D})$ is a quadratization of (4.1) for any complex-valued matrix $\mathbf{D}$ (see Example 5.4).

Although the requirements on the dimension-agnostic quadratization—that is, having the same "shape" of new variables for all dimensions and matrices—may appear restrictive, we show that such a quadratization always exists.

Theorem 4.6. *Every family of linearly coupled ODEs admits a monomial dimension-agnostic quadratization; this can be chosen such that* $\deg_{\tilde{\mathbf{x}}} \mathbf{w}_2(\mathbf{x}, \tilde{\mathbf{x}}) = 1$.

Furthermore, if the family is defined by $n_d + 1$ *polynomial vectors* $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x})$ *with* $d := \max_{0\leqslant j\leqslant n_d} \deg \mathbf{p}_j(\mathbf{x})$, *then there exists a dimension-agnostic quadratization with* $\ell \leqslant \binom{n_d+d}{d}$ *and* $L \leqslant n_d\binom{n_d+d}{d}$ *(in the notation of Definition 4.4).*

Proof. We construct such a quadratization explicitly. Let $w_{1,1}(\mathbf{x}), \dots, w_{1,\ell}(\mathbf{x})$ be the set of all monomials of degree at most d in $\mathbf{x}$, and let $w_{2,1}(\mathbf{x}, \tilde{\mathbf{x}}), \dots, w_{2,L}(\mathbf{x}, \tilde{\mathbf{x}})$ be the set of

all $2n$-variate monomials of degree at most one in $\tilde{\mathbf{x}}$ degree at most d in $\mathbf{x}$. Then $\ell = \binom{n_d+d}{d}$ and $L = n_d\binom{n_d+d}{d}$. We next show that these $\mathbf{w}_1 = [w_{1,1}(\mathbf{x}), \dots, w_{1,\ell}(\mathbf{x})]^\top$ and $\mathbf{w}_2 = [w_{2,1}(\mathbf{x}, \tilde{\mathbf{x}}), \dots, w_{2,L}(\mathbf{x}, \tilde{\mathbf{x}})]^\top$ provide a dimension-agnostic quadratization for the family defined by $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x})$.

In order to prove this, consider arbitrary n and matrices $\mathbf{D}_1, \dots, \mathbf{D}_{n_d} \in \mathbb{C}^{n\times n}$. We consider the corresponding set $\mathcal{M} := \mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}_1, \dots, \mathbf{D}_{n_d})$ as in (4.6). First we observe that every monomial on the right-hand side of $\mathcal{F}^{[n]}(\mathbf{p}_0, \dots, \mathbf{p}_{n_d}, \mathbf{D}_1, \dots, \mathbf{D}_{n_d})$ (see (4.5)) belongs to $\mathcal{M}$. Indeed, consider $\dot{\mathbf{x}}_i^{[n]}$ for some $1 \leqslant i \leqslant n$. If a monomial in $\dot{\mathbf{x}}_i^{[n]}$ comes from the $\mathbf{p}_0(\mathbf{x}_i^{[n]})$, then it is the form $m(\mathbf{x}_i^{[n]})$ with $\deg m \leqslant d$ and, thus, can be written as $w_{1,k}(\mathbf{x}_i^{[n]})$ for some $1 \leqslant k \leqslant \ell$. Otherwise, it is of the form $m(\mathbf{x}_i^{[n]})x_{i_0,j}$ for some $1 \leqslant j \leqslant n_d$ and $1 \leqslant i_0 \leqslant n$ such that $\dot{\mathbf{x}}_i^{[n]}$ depends on $\mathbf{x}_{i_0}^{[n]}$ and $\deg m \leqslant d$, and therefore can be presented as $w_{2,k}(\mathbf{x}_i^{[n]}, \mathbf{x}_{i_0}^{[n]})$ for some $1 \leqslant k \leqslant L$. Now we consider any element of $m(\mathbf{x}^{[n]}) \in \mathcal{M}$. It belongs to either $\mathbf{w}_1(\mathbf{x}_i^{[n]})$ for some i or to $\mathbf{w}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]})$ for some i_0, i_1. Consider the case of $\mathbf{w}_1(\mathbf{x}_i^{[n]})$:

$$\dot{\mathbf{w}}_1(\mathbf{x}_i^{[n]}) = \sum_{k=1}^{n_d} \dot{x}_{i,k}^{[n]} \frac{\partial \mathbf{w}_1(\mathbf{x}_i^{[n]})}{\partial x_{i,k}^{[n]}}.$$

Since every monomial of $\dot{x}_{i,k}^{[n]}$ belongs to $\mathcal{M}$ and every monomial in $\frac{\partial \mathbf{w}_1(\mathbf{x}_i^{[n]})}{\partial x_{i,k}^{[n]}}$ belongs to $\mathcal{M}$, the right-hand side of the equation above is quadratic in $\mathcal{M}$. The case of $m(\mathbf{x}^{[n]})$ belonging to $\mathbf{w}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]})$ is completely analogous. ∎

While this existence result is encouraging and yields a dimension-agnostic quadratization, the quadratization from the proof of Theorem 4.6 is large. To find a more compact quadratization, one would need to check if arbitrary $\mathbf{w}_1(\mathbf{x})$ and $\mathbf{w}_2(\mathbf{x}, \tilde{\mathbf{x}})$ yield a dimension-agnostic quadratization. Checking this directly from Definition 4.4 is not possible since the definition involves a universal quantifier. The next proposition gives a simple way of checking this. It turns out that it is sufficient to consider the case $n = 4$ with particular matrices $\mathbf{D}_i$. Note that, although the matrices $\mathbf{D}_i$ corresponding to a particular difference scheme of interest will be likely different from the ones given in the proposition below, it is guaranteed that a dimension-agnostic quadratization which works for the matrices in the proposition will work for any other matrices as well (for the way this works in practice, see subsection 5.3). The intuition behind this is that $n = 4$ is sufficient to exhibit all important combinatorial types of coupling, and, thus, a general matrix will always look locally as one of the fragments of the 4×4 matrix from the proposition. The proof of the next proposition formalizes this.

Proposition 4.7. *Consider a family of linearly coupled ODEs defined by $n_d + 1$ polynomial vectors $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x})$. Then polynomial vectors $\mathbf{w}_1(\mathbf{x})$ and $\mathbf{w}_2(\mathbf{x}, \tilde{\mathbf{x}})$ yield a monomial dimension-agnostic quadratization of the family if and only if $\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}_1^*, \dots, \mathbf{D}_{n_d}^*)$ from (4.6) is a quadratization for $\mathcal{F}^{[n]}(\mathbf{p}_0, \dots, \mathbf{p}_{n_d}, \mathbf{D}_1^*, \dots, \mathbf{D}_{n_d}^*)$, where $n = 4$ and $\mathbf{D}_1^*, \dots, \mathbf{D}_{n_d}^*$ are defined as follows:*

$$\mathbf{D}_i^* = \begin{pmatrix} a_i & b_i & 0 & c_i \\ 0 & d_i & e_i & 0 \\ 0 & 0 & f_i & 0 \\ 0 & 0 & 0 & g_i \end{pmatrix},$$

where $1 \leqslant i \leqslant n_d$ and $a_i, b_i, c_i, d_i, e_i, f_i, g_i$ are scalar parameters.

Proof. Assume that $\mathcal{M}_0 := \mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}_1^*, \dots, \mathbf{D}_{n_d}^*)$ is a quadratization for $\mathcal{F}_4 := \mathcal{F}^{[4]}(\mathbf{p}_0, \dots, \mathbf{p}_{n_d}, \mathbf{D}_1^*, \dots, \mathbf{D}_{n_d}^*)$ and consider arbitrary n and matrices $\mathbf{D}_1, \dots, \mathbf{D}_{n_d} \in \mathbb{C}^{n\times n}$. We set $\mathcal{M}_1 := \mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}_1, \dots, \mathbf{D}_{n_d})$. We show that $\mathcal{M}_1$ is a quadratization for $\mathcal{F}_n := \mathcal{F}^{[n]}(\mathbf{p}_0, \dots, \mathbf{p}_{n_d}, \mathbf{D}_1, \dots, \mathbf{D}_{n_d})$. Consider $1 \leqslant i_0 \neq i_1 \leqslant n$ such that $(\mathbf{D}_k)_{j_0,j_1} \neq 0$ for some k. We show that $\dot{\mathbf{w}}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]})$ is quadratic in $\mathcal{M}_1$. For every monomial $m(\mathbf{x}^{[n]})$ in $\dot{\mathbf{w}}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]})$, there are the following two options:

1. m depends only on $\mathbf{x}_{i_0}^{[n]}$ and $\mathbf{x}_{i_1}^{[n]}$. Then $m(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]})$ is present in $\mathcal{F}_4$ on the right-hand side of $\dot{\mathbf{w}}_2(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]})$. Hence, $m(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]})$ is quadratic in $\mathbf{w}_1(\mathbf{x}_1^{[4]}), \mathbf{w}_1(\mathbf{x}_2^{[4]})$, and $\mathbf{w}_2(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]})$. Since $\mathbf{w}_1(\mathbf{x}_{i_0}^{[n]}), \mathbf{w}_1(\mathbf{x}_{i_1}^{[n]}), \mathbf{w}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]}) \in \mathcal{M}_1$, we have that $m(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]})$ is quadratic in $\mathcal{M}_1$ as well.
2. m depends on $\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]}$, and $\mathbf{x}_{i_2}^{[n]}$ for some $i_2 \neq i_0, i_1$.
   - If m comes from $\frac{\partial \mathbf{w}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]})}{\partial x_{i_0,s}^{[n]}} \dot{x}_{i_0,s}^{[n]}$ for some s, we have that $(\mathbf{D}_k)_{i_0,i_2} \neq 0$ for some k. Then $m(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]}, \mathbf{x}_4^{[4]})$ appears in $\mathcal{F}_4$ on the right-hand side of $\dot{\mathbf{w}}_2(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]})$. Hence, $m(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]}, \mathbf{x}_4^{[4]})$ is quadratic in
   $$\mathbf{w}_1(\mathbf{x}_1^{[4]}),\ \mathbf{w}_1(\mathbf{x}_2^{[4]}),\ \mathbf{w}_1(\mathbf{x}_4^{[4]}),\ \mathbf{w}_2(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]}),\ \mathbf{w}_2(\mathbf{x}_1^{[4]}, \mathbf{x}_4^{[4]}).$$
   Getting back to $\mathcal{F}_n$, since
   $$\mathbf{w}_1(\mathbf{x}_{i_0}^{[n]}),\ \mathbf{w}_1(\mathbf{x}_{i_1}^{[n]}),\ \mathbf{w}_1(\mathbf{x}_{i_2}^{[n]}),\ \mathbf{w}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]}),\ \mathbf{w}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_2}^{[n]}) \in \mathcal{M}_1,$$
   we have that $m(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]}, \mathbf{x}_{i_2}^{[n]})$ is quadratic in $\mathcal{M}_1$.
   - If m comes from $\frac{\partial \mathbf{w}_2(\mathbf{x}_{i_0}^{[n]}, \mathbf{x}_{i_1}^{[n]})}{\partial x_{j_1,s}^{[n]}} \dot{x}_{i_1,s}^{[n]}$ for some s, we have that $(\mathbf{D}_k)_{i_1,i_2} \neq 0$ for some k. The argument is the same as in the previous case but using $m(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]}, \mathbf{x}_3)$ instead of $m(\mathbf{x}_1^{[4]}, \mathbf{x}_2^{[4]}, \mathbf{x}_4^{[4]})$.

The proof that $\dot{\mathbf{x}}^{[n]}$ and $\dot{\mathbf{w}}_1(\mathbf{x}_i^{[n]})$ are quadratic in $\mathcal{M}_1$ is analogous but simpler because only the first case (i.e., m depends on $\mathbf{x}_{i_0}^{[n]}$ and $\mathbf{x}_{i_1}^{[n]}$ for some i_0, i_1) is possible. Thus, $\mathcal{M}_1$ is a quadratization for $\mathcal{F}_n$. Since N and $\mathbf{D}_1, \dots, \mathbf{D}_n$ were chosen arbitrarily, $\mathbf{w}_1$ and $\mathbf{w}_2$ yield a dimension-agnostic quadratization for the family defined by $\mathbf{p}_0, \dots, \mathbf{p}_{n_d}$. ∎

5. The QBee algorithm and software for quadratization.

To the best of our knowledge, there are two software tools that generate quadratizations of different kind: Biocham [2] and QBee [14]. We focus on QBee because of its performance, stricter optimality guarantees [13, Table 3], and flexible algorithm design. In this work, we extend the capability of QBee according to the theory from the preceding sections. A Jupyter notebook with the examples from sections 5 to 7 is available online.[2]

5.1. Review of the original QBee algorithm.

The QBee algorithm from [14] takes as an input a system of polynomial ODEs and produces as an output an optimal monomial quadratization (in the sense of Definition 2.3). The QBee software is implemented in Python.

[2] Jupyter notebook: https://github.com/AndreyBychkov/QBee/blob/master/examples/Examples_BOPK2023.ipynb.

At the code level, QBee can be loaded by `import sympy as sp; from qbee import *`. After that, for example, the system

$$\dot{x}_1 = x_1^3 + x_2^2, \quad \dot{x}_2 = x_1 + x_2$$

can be quadratized by

```
x1, x2 = functions("x1, x2")
system = [ # pairs of the form (x, x')
    (x1, x1**3 + x2**2),
    (x2, x1 + x2)
]
quadratize(system).print()
```

where `system` defines a polynomial ODE system $\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x})$ as a list of pairs $(x_i, p_i(\mathbf{x}))$. QBee prints the quadratic symbolic form of the system and a list of auxiliary variables needed to do that. The list is guaranteed to be as short as possible. For the code above, QBee produces

```
    Introduced variables:
    w0 = x1**2
    w1 = x2**2

    x1' = x1*w0 + w1
    x2' = x1 + x2
    w0' = 2*x1*w1 + 2*w0**2
    w1' = 2*x1*x2 + 2*w1
```

The algorithms implemented in QBee are described in detail in [13]. To keep this paper self-contained, we briefly summarize the key ideas of the algorithm, which we build upon later. The algorithm follows the general branch-and-bound paradigm [48], and the computation of a quadratization is organized as a tree of recursive iterations. Each recursive iteration takes as input the original system, current set $\mathcal{S}$ of new variables ($\mathcal{S} = \varnothing$ at the first iteration), and the smallest order ℓ_0 of quadratization found so far, where initially we set ℓ_0 to be the size of quadratization from Theorem 2.5. If $\mathcal{S}$ is a quadratization, it is returned. If $|\mathcal{S}| \geqslant \ell_0$, this means that extending $\mathcal{S}$ does not lead to a better quadratization than already found, this branch is skipped. Furthermore, to keep the computation manageable, QBee implements additional pruning rules that allow the algorithm to decide that the branch can be safely skipped even if $|\mathcal{S}| < \ell_0$ [13, section 5]. Otherwise, the algorithm generates possible useful extensions of $\mathcal{S}$ by looking at not-yet-quadratized monomials in the derivatives of the original variables and elements of $\mathcal{S}$. It runs recursively on these extensions and updates ℓ_0 if any of these recursive iterations finds a better quadratization. The bound ℓ_0 forces the tree of recursive calls to be finite and thus implies that the algorithm terminates. The recursive step of the algorithm is summarized in Algorithm 5.1.

Algorithm 5.1. Outline of the original QBee algorithm from [13].

Inputs

- Symbolic form of the polynomial ODE system $\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x})$;
- A set of potential new variables $\mathcal{S} \subset \mathbb{C}[\mathbf{x}]$. If not specified, $\mathcal{S} = \varnothing$;
- The smallest order ℓ_0 of quadratization found so far. If not specified, we set $\ell_0 = \prod_{i=1}^{N}(d_i + 1)$, where $d_i = \deg_{x_i} \mathbf{p}(\mathbf{x})$ (see Theorem 2.5).

Output Optimal quadratization of $\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x})$ extending $\mathcal{S}$ with $|\mathcal{S}| < \ell_0$ or `nothing` if there is no such quadratization.

(Step 1) If $\mathcal{S}$ is a quadratization and $|\mathcal{S}| < \ell_0$, **return** $\mathcal{S}$.

(Step 2) If $|\mathcal{S}| \geqslant \ell_0$ or any pruning rule indicates that $\mathcal{S}$ cannot be extended to a quadratization of size $< \ell_0$, **return** `nothing`.

(Step 3) Generate a list $\mathcal{S}_1, \ldots, \mathcal{S}_r$ of possible extensions of $\mathcal{S}$.

(Step 4) Set $\mathcal{Q}_0 := $ `nothing`, $\ell := \ell_0$.

(Step 5) For $i = 1, \ldots, r$

(a) Run the algorithm recursively on $\mathcal{S}_i$ and ℓ.

(b) If the recursive call returns a quadratization $\mathcal{Q}$ with $|\mathcal{Q}| < \ell$, set $\mathcal{Q}_0 := \mathcal{Q}$ and $\ell := |\mathcal{Q}|$.

(Step 6) **Return** $\mathcal{Q}_0$.

Remark 5.1 (on Laurent polynomials in QBee). We have extended the original QBee algorithm outlined above to allow the input system to be defined not only by polynomials but also by *Laurent* polynomials, that is, polynomials with possible negative degrees. This is done by allowing negative degrees at (Step 3). In this case, we do not give termination and optimality guarantees but in practical examples (see subsubsection 6.1.2 and section 7) this approach works well. Therefore, all extensions of QBee presented here, such as Algorithms 5.2 and 5.3, can take Laurent polynomials as input but do not provide termination and optimality guarantees for this case.

We could instead first polynomialize the model using our polynomialization algorithm from subsection 5.4, but this may result in a quadratization of larger dimension, as the following example shows. Consider a Laurent polynomial model

$$\dot{x}_1 = x_2^2, \quad \dot{x}_2 = x_1 x_2^{-1}. \tag{5.1}$$

If we first polynomialize the model by introducing $x_3 = x_2^{-1}$, we obtain

$$\dot{x}_1 = x_2^2, \quad \dot{x}_2 = x_1 x_3, \quad \dot{x}_3 = -x_1 x_3^3. \tag{5.2}$$

Computation with QBee shows that at least two new variables are required to quadratize (5.2). On the other hand, QBee applied directly to (5.1) quadratizes the model with only two new variables, $w_1 = x_2^{-1}$ and $w_2 = x_1 x_2^{-2}$:

$$\dot{x}_1 = x_2^2, \quad \dot{x}_2 = x_1 w_1, \quad \dot{w}_1 = -w_1 w_2, \quad \dot{w}_2 = 1 - 2w_2^2.$$

This discrepancy arises because, when quadratizing (5.2), QBee cannot take into account the fact that $x_2 x_3 = 1$ as it is not a part of its input.

5.2. QBee for quadratization of polynomial ODEs with inputs. Herein, we describe how the results of section 3 can be used to add capabilities to Algorithm 5.1 for computing quadratizations for systems with inputs. This requires modifying the initial upper bound ℓ_0 and the procedure for computing the sets of variables for the recursive iterations at (Step 3). Consider the ODE system

$$\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x}, \mathbf{u}), \tag{5.3}$$

where $\mathbf{x} = \mathbf{x}(t) \in \mathbb{R}^N$ are the states and $\mathbf{u} = \mathbf{u}(t) \in \mathbb{R}^r$ are inputs and $\mathbf{p}(\mathbf{x}, \mathbf{u}) = [p_1(\mathbf{x}, \mathbf{u}), \ldots, p_N(\mathbf{x}, \mathbf{u})]^\top$ is a vector of polynomials.

Theorem 3.3 guarantees that it is always possible to find a quadratization with the new variables involving the inputs in at most zero order and, thus, the quadratized equations involving the inputs in at most first order (as in Definition 3.1). Since only zeroth- and first-order derivatives of inputs are involved in the quadratization process, we can impose an additional restriction that the inputs are linear,[3] that is, $\ddot{\mathbf{u}}(t) = 0$. With this additional restriction, the system can be written as an autonomous ODE system to which the original QBee algorithm can now be applied with two amendments: the bound ℓ_0 on the size of quadratization is now taken from Theorem 3.3, not from Theorem 2.5, and (Step 3) of Algorithm 5.1 is restricted to monomials in $\mathbf{x}$ and $\mathbf{u}$ (that is, using $\dot{\mathbf{u}}$ is not allowed). Theorem 3.3 guarantees that even under this restriction a quadratization exists, and QBee will find an optimal one.

To compute an input-free quadratization (for cases when $\mathbf{u}(t)$ is not differentiable), the approach above is modified as follows. First, we set $\ell_0 = \infty$. Second, we further restrict (Step 3) of Algorithm 5.1 by allowing only the extensions not involving either $\mathbf{u}$ or $\dot{\mathbf{u}}$. We note that, in general, the search for input-free quadratizations is not guaranteed to terminate, as there may not be input-free quadratizations. We refer the reader to subsection 3.2 for the criterion (Proposition 3.9) and special cases (Lemmas 3.8 and 3.13) where we expect the algorithm to terminate. We summarize the algorithm explained above in Algorithm 5.2.

Example 5.2 (use of QBee for systems with inputs). Consider again the polynomial model with input from Example 3.10. The QBee software finds a quadratization for that example with the following code:

```
x, u = functions("x, u")
system = [ # pairs of the form (x, x')
    (x, x**2 * u)
]
quadratize(system).print() # input-free=False by default
```

[3] Formally, this can be justified as follows. We introduce the new "state" variables: $\mathbf{u}^{(0)} = \mathbf{u}$ and $\mathbf{u}^{(1)} = \dot{\mathbf{u}}$, as the zeroth and first derivatives of $\mathbf{u}$, respectively. This allows us to formally write an *autonomous* ODE system $\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x}, \mathbf{u}^{(0)}), \dot{\mathbf{u}}^{(0)} = \mathbf{u}^{(1)}, \dot{\mathbf{u}}^{(1)} = \mathbf{0}$, where the last equation acts as a dummy equation. Since by Definition 3.1 the quadratizations of the nonautonomous system (5.3) involve only $\mathbf{x}$ and $\mathbf{u}$ but not derivatives of $\mathbf{u}$, there is a bijection between the quadratizations of (5.3) and the quadratizations of the autonomous system above involving $\mathbf{x}$ and $\mathbf{u}^{(0)}$ but not $\mathbf{u}^{(1)}$.

Algorithm 5.2. Extension of QBee to nonautonomous systems.

Input Nonautonomous polynomial ODE system $\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x}, \mathbf{u})$ with N states and r inputs and a Boolean flag `input-free`.

Output Optimal quadratization as defined in Definition 3.1 if `input-free` = `False` and optimal input-free quadratization as in Definition 3.4 if `input-free` = `True` and such quadratization exists.

(Step 1) Build an autonomous ODE system by restricting inputs to the linear ones, that is, adding $\ddot{u} = 0$ to the original system.

(Step 2) Run Algorithm 5.2 on the autonomous ODE system constructed in the previous step.

- If `input-free` = `False`, use ℓ_0 computed as in Theorem 3.3 and ensure that **(Step 3)** only selects the extensions not involving $\dot{\mathbf{u}}$.
- If `input-free` = `True`, use $\ell_0 = \infty$ and ensure that **(Step 3)** only selects the extensions not involving $\mathbf{u}, \dot{\mathbf{u}}$.

(Step 3) **Return** the quadratization found at the previous step.

The code produces

```
Introduced variables:
w0 = u*x

x' = x*w0
w0' = x*u' + w0**2
```

Example 5.3 (use of QBee for input-free quadratizations). Consider again the polynomial model from Example 3.11. The QBee software finds an input-free quadratization for that example with the following code:

```
x1, x2, u = functions("x1, x2, u")
system = [ # pairs of the form (x, x')
    (x1, x1 + x1 * u),
    (x2, x1**2 * u)
]
quadratize(system, input_free=True).print()
```

In the code above, `input_free=True` means that we are looking for an input-free quadratization. The code produces

```
Introduced variables:
w0 = x1**2

x1' = x1*u + x1
x2' = u*w0
w0' = 2*u*w0 + 2*w0
```

5.3. QBee for finding dimension-agnostic quadratizations. In section 4 we presented the theory for quadratizing families of linearly coupled ODE systems. By this, we mean that while the ODE dimension could be arbitrary, the equations have the same structure—so, in essence, we can find a quadratization for a lower-dimensional ODE, and this generalizes to adding more (similarly structured) equations.

We revisit Example 4.1, where we found a quadratization of a family of ODE systems of the form (4.3):

$$\dot{\mathbf{x}}^{[n]} = \mathbf{x}^{[n]} + (\mathbf{x}^{[n]})^2 \odot (\mathbf{D}\mathbf{x}^{[n]}).$$

To provide such a family as an input to our algorithm, we formally write this ODE system as a scalar equation,

$$\dot{x} = x + x^2 x_{\mathbf{D}}, \tag{5.4}$$

where $x_{\mathbf{D}}$ is a formal variable that we use as a placeholder for the linear coupling $\mathbf{D}\mathbf{x}^{[n]}$. We extend this notation to the general case of a family of linearly coupled ODE systems as in Definition 4.2. We represent the family symbolically as

$$\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x}, \mathbf{x}_{\mathbf{D}}), \tag{5.5}$$

where $\mathbf{x} = [x_1, \dots, x_{n_d}]^\top$, $\mathbf{x}_{\mathbf{D}} = [x_{\mathbf{D},1}, \dots, x_{\mathbf{D},n_d}]^\top$ are again formal variables which are placeholders for the coupling expressions $(\mathbf{D}_j \mathbf{x}^{[n]}_{*,j})_i$, and $\mathbf{p}(\mathbf{x}, \mathbf{x}_{\mathbf{D}})$ is affine in $\mathbf{x}_{\mathbf{D}}$, i.e.,

$$\mathbf{p}(\mathbf{x}, \mathbf{x}_{\mathbf{D}}) = \mathbf{p}_0(\mathbf{x}) + \sum_{i=1}^{n_d} \mathbf{p}_i(\mathbf{x}) x_{\mathbf{D},i}. \tag{5.6}$$

The vectors $\mathbf{p}_0(\mathbf{x}), \dots, \mathbf{p}_{n_d}(\mathbf{x})$ define the family of linearly coupled models as in Definition 4.2.

Given a family of linearly coupled ODE systems written in the form (5.5), Proposition 4.7 provides an effective condition to check whether given polynomial vectors $\mathbf{w}_1(\mathbf{x})$ and $\mathbf{w}_2(\mathbf{x}, \tilde{\mathbf{x}})$ provide a dimension-agnostic quadratization (see Definition 4.4) of the family: it is sufficient to take a particular member of the family, $\mathcal{F}^{[4]}_{\mathbf{P}}(\mathbf{p}_0, \dots, \mathbf{p}_{n_d}, \mathbf{D}^*_1, \dots, \mathbf{D}^*_{n_d})$, and check whether $\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}^*_1, \dots, \mathbf{D}^*_{n_d})$ is a quadratization of this specific ODE system. Moreover, we can use Proposition 4.7 to find a dimension-agnostic quadratization for the family by running Algorithm 5.1 on the discretization but restricting the extensions at (Step 3) to ones of the form $\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}^*_1, \dots, \mathbf{D}^*_{n_d})$. Success and termination of this search can be ensured by using the bounds for ℓ and L from Theorem 4.6 to provide the starting bound ℓ_0 in Algorithm 5.1: since there are three off-diagonal elements in the D_i's and $n = 4$, we take $\ell_0 = 4\ell + 3L = (3n_d + 4)\binom{n_d+d}{d}$. This approach is summarized in Algorithm 5.3.

Example 5.4 (use of QBee for dimension agnostic quadratization). Consider the family from Example 4.1. We have already written the corresponding family of ODE systems in the form (5.4) required by the algorithm. QBee can be used to find a dimension-agnostic quadratization by writing the following code:

```
x, Dx = functions("x Dx")
system = [
    (x, x + x**2 * Dx)
]
quadratize_dimension_agnostic(system)
```

Then QBee produces

```
Every ODE system in the family can be quadratized by adding the following variables
* For each i, we take variables
x_i**2
* For every i and j such that the variables from the j-th block (that is, x_j)
appear in the right-hand sides of the i-th block (that is, in x_i'), we take
x_i*x_j
```

The quadratized system like (4.4) in general depends on the shape of the $\mathbf{D}$ matrices used (see Example 4.5), and it is not produced automatically but has to be derived by hand using the knowledge of the quadratizing variables. The function `quadratize_dimension_agnostic` has a keyword parameter `print_intermediate`; if the value is set to be `True`, the quadratization for the ODE system from Proposition 4.7 is printed, which can be helpful when producing the quadratized system.

Algorithm 5.3. Extension of QBee to dimension-agnostic quadratizations.

Input A family of linearly coupled ODE systems (4.5) presented as

$$\dot{\mathbf{x}} = \mathbf{p}(\mathbf{x}, \mathbf{x}_{\mathbf{D}}),$$

where $\mathbf{x} = [x_1, \dots, x_{n_d}]^\top$, $\mathbf{x}_{\mathbf{D}} = [x_{\mathbf{D},1}, \dots, x_{\mathbf{D},n_d}]$ are formal variables which are placeholders for the coupling, and $\mathbf{p}(\mathbf{x}, \mathbf{x}_{\mathbf{D}})$ is affine in $\mathbf{x}_{\mathbf{D}}$.

Output A dimension-agnostic quadratization of the family (see Definition 4.4).

(Step 1) Set $d := \deg_{\mathbf{x}} \mathbf{p}(\mathbf{x}, \mathbf{x}_{\mathbf{D}})$.

(Step 2) Write $\mathbf{p}(\mathbf{x}, \mathbf{x}_{\mathbf{D}})$ in the form (5.6) and extract the vectors $\mathbf{p}_0, \mathbf{p}_1, \dots, \mathbf{p}_{n_d}$.

(Step 3) Construct an ODE system $\mathcal{F}^{[4]}_{\mathbf{P}}(\mathbf{p}_0, \dots, \mathbf{p}_{n_d}, \mathbf{D}^*_1, \dots, \mathbf{D}^*_{n_d})$ from Proposition 4.7.

(Step 4) Apply Algorithm 5.1 to the produced ODE system with $\ell_0 = (3n_d + 4)\binom{n_d+d}{d}$ and **(Step 3)** selecting extensions of the form $\mathcal{M}(\mathbf{w}_1, \mathbf{w}_2; \mathbf{D}^*_1, \dots, \mathbf{D}^*_{n_d})$.

(Step 5) From the produced quadratization, derive the polynomial vectors $\mathbf{w}_1(\mathbf{x})$ and $\mathbf{w}_2(\mathbf{x}, \tilde{\mathbf{x}})$ and **return** them.

Since (Step 4) of Algorithm 5.3 applies QBee to $\mathcal{F}^{[4]}(\mathbf{p}_0, \dots, \mathbf{p}_{n_d}, \mathbf{D}^*_1, \dots, \mathbf{D}^*_{n_d})$, the resulting dimension-agnostic quadratization is not guaranteed to be optimal for all choices of n and $\mathbf{D}_1, \dots, \mathbf{D}_{n_d}$. In all the examples we have considered so far, and the forthcoming examples in sections 6 and 7, the dimension-agnostic quadratization produced by the algorithm was always of the same or lower dimension as the ones found by hand.

5.4. Polynomialization as a preprocessing step for quadratization. This work focuses on the task of quadratizing polynomial ODEs. However, many models in engineering and science are nonpolynomial. The problem of optimal quadratization of nonpolynomial systems remains open, but practical polynomialization algorithms can nevertheless be devised. We design and implement a polynomialization algorithm that can be used as a preprocessing step before quadratization with Algorithms 5.1 and 5.2. The algorithm follows the general approach described in [24, 31]: at each step the algorithm finds a nonpolynomial term and adds a new

variable corresponding to it. The main difference of our approach from the prior algorithms is that we again employ the branch-and-bound approach as in Algorithm 5.1, that is, the algorithm does not perform a fixed sequence of substitutions but explores different sequences in a recursive fashion. While this may require more iterations, the resulting system can be of smaller dimensions, as the next example shows.

Example 5.5. Consider the scalar nonpolynomial equation $\dot{x} = e^{-x} + e^{-2x}$. Here, the Biocham [2] software introduces two new variables, e^{-x} and e^{-2x}, by using the algorithm from [31]. In contrast, our QBee algorithm finds that only one new variable, $w := e^{-x}$, is sufficient to make the system polynomial:

$$\dot{x} = w + w^2,$$
$$\dot{w} = -w^2 - w^3.$$

The following QBee code produces the above polynomialization of order one:

```
x = functions("x")
system = [(x, sp.exp(-x) + sp.exp(-2 * x))] # list of pairs (x, x')
print(polynomialize(system))
```

The QBee output is

```
w_0  = exp(-x)

  x' = w_0**2 + w_0
w_0' = -w_0*(w_0**2 + w_0)
```

5.5. On optimality guarantees of the presented algorithms. We summarize the optimality guarantees which are available for the algorithms we presented herein.

- If the input is a polynomial ODE system of the form (2.1) or (3.2), then the original QBee (Algorithm 5.1) and its extension to the nonautonomous systems (Algorithm 5.2) are guaranteed to produce an optimal monomial quadratization.
- For the polynomialization algorithm (subsection 5.4), the algorithm for computing dimension-agnostic quadratizations (Algorithm 5.3), as well as Algorithms 5.1 and 5.2 when allowing for Laurent polynomials in the model form, we do not guarantee optimality. However, our algorithms always reduce the number of new variables by applying the branch-and-bound approach from QBee and, in particular, manage to produce quadratizations of smaller dimensions than was possible before (see sections 6 and 7)

6. Applications of quadratization. This section applies the QBee algorithm to quadratize a wide range of nonlinear systems of ODEs. Here, we revisit examples from the literature and demonstrate that we can improve the order of quadratization (i.e., we need fewer variables to do so). Subsection 6.1 considers two different tubular reactor models, both with variable dimension and external inputs In subsection 6.2 we consider a two-step rocket combustion process which is nonpolynomial.

6.1. Tubular reactor model. We consider a nonadiabatic tubular reactor model with a single reaction as in [29] and follow the discretization and setup in [63, 43]. The spatially discretized model describes the evolution of the species concentration vector $\psi(t)$ and

temperature vector $\boldsymbol{\theta}(t)$ over time $t > 0$. We consider two cases of different complexity. Both cases are nonautonomous and of variable dimension (depending on the size of discretization).

6.1.1. Polynomial reaction model. We consider the polynomial ODE in $2n$ (i.e., $n_d = 2$) dimensions:

$$\begin{aligned} \dot{\boldsymbol{\psi}} &= \mathbf{A}_\psi \boldsymbol{\psi} + \mathbf{b}_\psi - \mathcal{D}\boldsymbol{\psi} \odot (\mathbf{c}_0 + \mathbf{c}_1 \odot \boldsymbol{\theta} + \mathbf{c}_2 \odot \boldsymbol{\theta}^2 + \mathbf{c}_3 \odot \boldsymbol{\theta}^3), \\ \dot{\boldsymbol{\theta}} &= \mathbf{A}_\theta \boldsymbol{\theta} + \mathbf{b}_\theta + \mathbf{b}u + \mathcal{B}\mathcal{D}\boldsymbol{\psi} \odot (\mathbf{c}_0 + \mathbf{c}_1 \odot \boldsymbol{\theta} + \mathbf{c}_2 \odot \boldsymbol{\theta}^2 + \mathbf{c}_3 \odot \boldsymbol{\theta}^3), \end{aligned} \tag{6.1}$$

where $u = u(t)$ is a scalar input function, $\mathbf{A}_\psi$ and $\mathbf{A}_\theta$ are $n \times n$ constant matrices, and $\mathbf{b}, \mathbf{b}_\psi$, and $\mathbf{b}_\theta$ are n-dimensional constant vectors. The Damköhler number $\mathcal{D}$ and $\mathcal{B}$ are the known constants. This ODE system has a polynomial reaction model, i.e., the chemical reaction term is $f(\boldsymbol{\psi}, \boldsymbol{\theta}; \gamma) = \psi(c_0 + c_1\boldsymbol{\theta} + c_2\boldsymbol{\theta}^2 + c_3\boldsymbol{\theta}^3)$ following [43, equations (17)–(18)]. In [43] this model was quadratized with $5n$ additional variables, so the lifted system dimension was $7n$.

In order to use QBee, we rewrite (6.1) in the form (5.5), that is, as a formal three-dimensional ODE system with additional variables $\psi_{\mathbf{D}}$ and $\theta_{\mathbf{D}}$ instead of $\mathbf{A}_\psi\boldsymbol{\psi}$ and $\mathbf{A}_\theta\boldsymbol{\theta}$:

$$\begin{aligned} \dot{\psi} &= \psi_{\mathbf{D}} + b_\psi - \mathcal{D}\psi(c_0 + c_1\theta + c_2\theta^2 + c_3\theta^3), \\ \dot{\theta} &= \theta_{\mathbf{D}} + b_\theta + bu + \mathcal{B}\mathcal{D}\psi(c_0 + c_1\theta + c_2\theta^2 + c_3\theta^3). \end{aligned} \tag{6.2}$$

Then we use QBee to search for a discretization-agnostic quadratization as described in subsection 5.3. QBee finds that for every n and for all matrices $\mathbf{A}_\psi$ and $\mathbf{A}_\theta$, the system can be quadratized using the following $4n$ additional variables:

$$\mathbf{w}_1 := \boldsymbol{\theta}^2, \quad \mathbf{w}_2 := \boldsymbol{\theta}^3, \quad \mathbf{w}_3 := \boldsymbol{\psi} \odot \boldsymbol{\theta}, \quad \mathbf{w}_4 := \boldsymbol{\psi} \odot \boldsymbol{\theta}^2. \tag{6.3}$$

One can use these new variables to write the quadratized system:

$$\begin{aligned} \dot{\boldsymbol{\psi}} &= \mathbf{A}_\psi\boldsymbol{\psi} + \boldsymbol{b}_\psi - \mathcal{D}(\mathbf{c}_0 \odot \boldsymbol{\psi} + \mathbf{c}_1 \odot \mathbf{w}_3 + \mathbf{c}_2 \odot \mathbf{w}_4 + \mathbf{c}_3 \odot \mathbf{w}_1 \odot \mathbf{w}_3), \\ \dot{\boldsymbol{\theta}} &= \mathbf{A}_\theta\boldsymbol{\theta} + \boldsymbol{b}_\theta + \boldsymbol{b}u + \mathcal{B}\mathcal{D}(\mathbf{c}_0 \odot \boldsymbol{\psi} + \mathbf{c}_1 \odot \mathbf{w}_3 + \mathbf{c}_2 \odot \mathbf{w}_4 + \mathbf{c}_3 \odot \mathbf{w}_1 \odot \mathbf{w}_3), \\ \dot{\mathbf{w}}_1 &= 2\boldsymbol{\theta}(\mathbf{A}_\theta\boldsymbol{\theta} + \boldsymbol{b}_\theta + \boldsymbol{b}u) + 2\mathcal{B}\mathcal{D}(\mathbf{c}_1 \odot \mathbf{w}_4 + \mathbf{w}_3 \odot (\mathbf{c}_0 + \mathbf{c}_2 \odot \mathbf{w}_1 + \mathbf{c}_3 \odot \mathbf{w}_2)), \\ \dot{\mathbf{w}}_2 &= 3\boldsymbol{\theta}(\mathbf{A}_\theta\boldsymbol{\theta} + \boldsymbol{b}_\theta + \boldsymbol{b}u) + 3\mathcal{B}\mathcal{D}(\mathbf{w}_4 \odot (\mathbf{c}_0 + \mathbf{c}_3 \odot \mathbf{w}_2) + \mathbf{w}_3 \odot (\mathbf{c}_1 \odot \mathbf{w}_1 + \mathbf{c}_2 \odot \mathbf{w}_2)), \\ \dot{\mathbf{w}}_3 &= \boldsymbol{\psi}(\mathbf{A}_\theta\boldsymbol{\theta} + \boldsymbol{b}_\theta + \boldsymbol{b}u) + \boldsymbol{\theta}(\mathbf{A}_\psi\boldsymbol{\psi} + \boldsymbol{b}_\psi) - \mathcal{D}(\mathbf{c}_1 \odot \mathbf{w}_4 + \mathbf{w}_3 \odot (\mathbf{c}_0 + \mathbf{c}_2 \odot \mathbf{w}_1 + \mathbf{c}_3 \odot \mathbf{w}_2)) \\ &\quad + \mathcal{B}\mathcal{D}(\mathbf{c}_0 \odot \boldsymbol{\psi}^2 + \mathbf{w}_3 \odot (\mathbf{c}_1 \odot \boldsymbol{\psi} + \mathbf{c}_2 \odot \mathbf{w}_3 + \mathbf{c}_3 \odot \mathbf{w}_4)), \\ \dot{\mathbf{w}}_4 &= \mathbf{w}_1(\mathbf{A}_\psi\boldsymbol{\psi} + \boldsymbol{b}_\psi) + 2\mathbf{w}_3(\mathbf{A}_\theta\boldsymbol{\theta} + \boldsymbol{b}_\theta + \boldsymbol{b}u) \\ &\quad - \mathcal{D}(\mathbf{c}_0 \odot \mathbf{w}_4 + \mathbf{c}_1 \odot \mathbf{w}_1 \odot \mathbf{w}_3 + \mathbf{w}_2 \odot (\mathbf{c}_2 \odot \mathbf{w}_3 + \mathbf{c}_3 \odot \mathbf{w}_4)) \\ &\quad + 2\mathcal{B}\mathcal{D}(\mathbf{c}_0 \odot \boldsymbol{\psi} \odot \mathbf{w}_3 + \mathbf{c}_1 \odot \mathbf{w}_3^2 + \mathbf{c}_2 \odot \mathbf{w}_3 \odot \mathbf{w}_4 + \mathbf{c}_3 \odot \mathbf{w}_4^2). \end{aligned} \tag{6.4}$$

Thus, QBee produced a lower-dimensional quadratization ($4n$ extra variables) than has been previously reported in the literature ($5n$ extra variables).

6.1.2. Exponential reaction terms. We now consider a nonpolynomial nonautonomous ODE system. The model is taken from [42, sect. V] and includes an Arrhenius reaction term that is of exponential kind, i.e., the system of nonpolynomial nonautonomous ODEs is

$$\begin{aligned}\dot{\boldsymbol{\psi}} &= \mathbf{A}_\psi \boldsymbol{\psi} + \mathbf{b}_\psi - \mathcal{D}\boldsymbol{\psi} \odot e^{\gamma - \gamma/\boldsymbol{\theta}},\\ \dot{\boldsymbol{\theta}} &= \mathbf{A}_\theta \boldsymbol{\theta} + \mathbf{b}_\theta + \mathbf{b}u(t) + \mathcal{B}\mathcal{D}\boldsymbol{\psi} \odot e^{\gamma - \gamma/\boldsymbol{\theta}}.\end{aligned} \tag{6.5}$$

In [42] two lifting transformations are derived: (i) the system is polynomialized with $3n$ extra variables, $\mathbf{w}_1 := \boldsymbol{\theta}^{-1}, \mathbf{w}_2 := \boldsymbol{\theta}^{-2}$, and $\mathbf{w}_3 := e^{\gamma - \gamma/\boldsymbol{\theta}}$; (ii) the system is further transformed into a quadratic system of differential-algebraic equations (DAEs) by adding $3n$ additional variables, thus, adding $6n$ variables in total.

With QBee, we improve on these results. First, we manually introduce only one variable $\mathbf{w}_1 := e^{\gamma - \gamma/\boldsymbol{\theta}}$ and obtain a $3n$-dimensional system with Laurent polynomials on the right-hand side:

$$\begin{aligned}\dot{\boldsymbol{\psi}} &= \mathbf{A}_\psi \boldsymbol{\psi} + \mathbf{b}_\psi - \mathcal{D}\boldsymbol{\psi} \odot \mathbf{w}_1,\\ \dot{\boldsymbol{\theta}} &= \mathbf{A}_\theta \boldsymbol{\theta} + \mathbf{b}_\theta + \mathbf{b}u(t) + \mathcal{B}\mathcal{D}\boldsymbol{\psi} \odot \mathbf{w}_1,\\ \dot{\mathbf{w}}_1 &= \gamma \dot{\boldsymbol{\theta}} \odot \frac{1}{\boldsymbol{\theta}^2} \odot \boldsymbol{w}_1.\end{aligned} \tag{6.6}$$

In order to use QBee, we rewrite (6.6) in the form (5.5), that is, as a formal three-dimensional ODE system with additional variables $\psi_{\mathbf{D}}$ and $\theta_{\mathbf{D}}$ instead of $\mathbf{A}_\psi \boldsymbol{\psi}$ and $\mathbf{A}_\theta \boldsymbol{\theta}$:

$$\begin{aligned}\dot{\psi} &= \psi_{\mathbf{D}} + b_\psi - \mathcal{D}\psi w_1,\\ \dot{\theta} &= \theta_{\mathbf{D}} + b_\theta u + \mathcal{B}\mathcal{D}\psi w_1,\\ \dot{w}_1 &= \gamma \frac{w_1}{\theta^2}(\theta_{\mathbf{D}} + b_\theta + bu + \mathcal{B}\mathcal{D}\psi w_1).\end{aligned}$$

Then we use QBee to search for a discretization-agnostic quadratization as described in subsection 5.3. As explained in Remark 5.1, this can be done even though (6.6) is a Laurent polynomial system. QBee finds that for every n and for all matrices $\mathbf{A}_\psi$ and $\mathbf{A}_\theta$, a quadratization requires the additional variables

$$\begin{aligned}\mathbf{w}_2 &:= \frac{1}{\boldsymbol{\theta}}, & \mathbf{w}_4 &:= \frac{u(t)}{\boldsymbol{\theta}}, & \mathbf{w}_8 &:= \frac{\boldsymbol{\psi} \odot \mathbf{w}_1}{\boldsymbol{\theta}^2},\\ \mathbf{w}_3 &:= \frac{1}{\boldsymbol{\theta}^2}, & \mathbf{w}_5 &:= \frac{u(t)}{\boldsymbol{\theta}^2}, & \mathbf{w}_7 &:= \frac{\boldsymbol{\psi} \odot \mathbf{w}_1}{\boldsymbol{\theta}},\end{aligned}$$

and we additionally require for every $1 \leqslant i \neq j \leqslant n$, where $(\mathbf{A}_\psi)_{i,j} \neq 0$ or $(\mathbf{A}_\theta)_{i,j} \neq 0$:

$$w_{8,i,j} := \frac{\psi_j}{\theta_i}, \quad w_{9,i,j} := \frac{\psi_j}{\theta_i^2}, \quad w_{10,i,j} := \frac{\theta_j}{\theta_i}, \quad w_{11,i,j} := \frac{\theta_j}{\theta_i^2}.$$

Therefore, if we denote by M the number of nonzero off-diagonal elements in $\boldsymbol{A}_\theta$ and $\mathbf{A}_\psi$, we add $7n + 4M$ new variables. While these are more additional variables than the aforementioned $6n$ new variables from [42], our transformation yields an ODE (not a DAE) which is generally much more advantageous to work with for analysis, simulation, and control.

6.2. Species' reaction model for combustion. We consider the rocket engine combustion model from [61, equations (A1a)–(A1d)]. At each spatial grid point, the species' molar concentrations x_1, x_2, x_3, and x_4 are determined by the following equations:

$$\begin{aligned} \dot{x}_1 &= -A \exp\Big(-\frac{E}{Ru(t)}\Big) x_1^{0.2} x_2^{1.3}, \\ \dot{x}_2 &= 2\dot{x}_1, \\ \dot{x}_3 &= -\dot{x}_1, \\ \dot{x}_4 &= -2\dot{x}_1, \end{aligned} \tag{6.7}$$

where $u(t)$ is a time-dependent input standing for the temperature, and A, E, and R are known parameters. Our goal is to compute a polynomialization and a quadratization via QBee and compare the results to [61], where this was done by hand. QBee performs polynomialization and quadratization of the model automatically. It first finds three variables with which the system can be lifted using Laurent polynomials:

$$w_1 = x_2^{1.3}, \quad w_2 = x_1^{0.2}, \quad w_3 = e^{-E/(Ru(t))}$$

The lifted system will be

$$\begin{aligned} &\dot{x}_1 = -Aw_1w_2w_3, && \dot{x}_2 = -2Aw_1w_2w_3, && \dot{x}_3 = Aw_1w_2w_3, && \dot{x}_4 = 2Aw_1w_2w_3, \\ &\dot{w}_1 = -2.6Aw_1^2w_2w_3x_2^{-1}, && \dot{w}_2 = -0.2Aw_1w_2^2w_3x_1^{-1}, && \dot{w}_3 = \frac{E\dot{u}(t)w_3}{Ru^2(t)}. \end{aligned}$$

Then this system is quadratized using seven more variables:

$$\begin{aligned} &w_4 = w_1w_2, && w_5 = \frac{\dot{u}(t)}{u(t)^2}, && w_6 = \frac{1}{u(t)^2}, && w_7 = \frac{\dot{u}(t)}{u(t)}, \\ &w_8 = \frac{1}{u(t)}, && w_9 = w_1w_2w_3x_1^{-1}, && w_{10} = w_1w_2w_3x_2^{-1}. \end{aligned}$$

The resulting quadratic system is

$$\begin{aligned} &\dot{x}_1 = -Aw_1w_4, && \dot{w}_2 = -0.2Aw_2w_9, && \dot{w}_7 = \ddot{u}(t)w_8 - w_7^2, \\ &\dot{x}_2 = -2Aw_1w_4, && \dot{w}_3 = \frac{Ew_3w_5}{R}, && \dot{w}_8 = -w_7w_8, \\ &\dot{x}_3 = Aw_1w_4, && \dot{w}_4 = -0.2Aw_4w_9 + \frac{Ew_4w_5}{R}, && \dot{w}_9 = Aw_9\left(0.8w_9 - 2.6w_{10} + \frac{Ew_5}{AR}\right), \\ &\dot{x}_4 = 2Aw_1w_4, && \dot{w}_5 = \ddot{u}(t)w_6 - 2w_5w_7, && \dot{w}_{10} = -Aw_{10}\left(0.2w_9 + 0.6w_{10} - \frac{Ew_5}{AR}\right), \\ &\dot{w}_1 = -2.6Aw_1w_{10}, && \dot{w}_6 = -2w_6w_7. \end{aligned}$$

In total, we need to add ten new variables to the original system (6.7) to obtain a quadratic ODE system. The authors in [61] did not find a quadratized form for this system by hand due to the complexity of finding such quadratizations. This illustrates a significant advantage of the automated polynomialization and quadratization implemented in QBee. Furthermore, one

can observe that the equations for the new variables do not involve x_1 and x_2, and x_1 and x_2 can be expressed as $w_1w_2w_3w_9^{-1}$ and $w_1w_2w_3w_{10}^{-1}$, respectively. Thus, one can omit x_1 and x_2 and work with only 12-dimensional systems.

This example exhibits the same subtlety as noted in Remark 5.1: if we first transformed the system to a polynomial (not Laurent polynomial) one and then quadratized, then we would add eleven variables. We also note that it is not possible to find a quadratization without the appearance of $\ddot{u}(t)$ in the resulting system; this can be deduced by applying Proposition 3.9 to the polynomialized system.

7. Model learning for solar wind prediction with quadratized variables.

In this section, we illustrate the advantages of quadratization in the use case of data-driven reduced-order modeling for solar wind prediction. The benefit of a quadratization of the dynamical system is that it provides a direct parametrization of the model and avoids hyperreduction, as discussed in the introduction. Thus, the free parameters (the coefficients matrices for the polynomial terms) can be learned from trajectory data; see [61, 53, 54, 36, 59] for a variety of applications of this approach. Subsection 7.1 briefly introduces the model, subsection 7.3 gives implementation details, subsection 7.2 presents the results of the QBee quadratization, and subsection 7.4 presents the numerical result on the simulated quadratized model.

7.1. Model.

The Heliospheric Upwind Extrapolation (HUX) model [56] is a two-dimensional nonlinear scalar homogeneous time-stationary PDE of the solar wind radial velocity in the heliospheric domain, where the independent variables are the radial distance from the Sun r and Carrington longitude ϕ and the dependent variable is the solar wind velocity in the radial direction $v(r,\phi)$. The angular frequency of the Sun's rotation is evaluated at a constant Carrington latitude $\hat{\theta}$; if we consider the Sun's equatorial plane ($\hat{\theta}=0$), then $\Omega_{\text{rot}}(0)=\frac{2\pi}{25.38}1/\text{days}$ at the solar equator. The initial condition $v(r_0,\phi)=v_0(\phi)$ is defined on the periodic domain $0\le\phi\le 2\pi$ and $r\ge 0.14\text{AU}$.

After semidiscretization via the upwind scheme, and an r-dependent linear shift of the longitude to account for advection (see [35] for details), the finite-dimensional nonlinear model is

$$\frac{\mathrm{d}\mathbf{v}(r)}{\mathrm{d}r}=\mathbf{D}\ln[\mathbf{v}(r)]-\frac{\xi}{\Omega_{\text{rot}}(\hat{\theta})}\mathbf{D}\mathbf{v}(r), \tag{7.1}$$

where $\mathbf{v}(r)=[v(r,\phi_1),v(r,\phi_2),\dots,v(r,\phi_n)]^\top\in\mathbb{R}^n$ is the state vector discretized over N points in longitude at heliocentric distance r, ξ is the shift velocity which is fixed for a given Carrington rotation, and the sparse matrix $\mathbf{D}\in\mathbb{R}^{n\times n}$ is

$$\mathbf{D}=\frac{\Omega_{\text{rot}}(\hat{\theta})}{\Delta\phi}\begin{bmatrix}-1&1&0&&&&\\0&-1&1&&&&\\&\ddots&\ddots&\ddots&\ddots&&\\&&&-1&1&0\\&&&&-1&1\\1&&&&0&-1\end{bmatrix}\in\mathbb{R}^{n\times n}. \tag{7.2}$$

7.2. Quadratization of the model. To obtain a system of quadratic ODEs, we start with the original model (7.1) and eliminate the logarithmic function by adding a new variable $\mathbf{w}_0 := \ln(\mathbf{v})$. Let $C_1 := \frac{\xi}{\Omega_{\text{rot}}(\hat{\theta})}$, so that in the variables $\mathbf{v}$ and $\mathbf{w}_0$, the system (7.1) becomes

$$\begin{aligned} \frac{\mathrm{d}\mathbf{v}}{\mathrm{d}r} &= \mathbf{D}\mathbf{w}_0 - C_1\mathbf{D}\mathbf{v}, \\ \frac{\mathrm{d}\mathbf{w}_0}{\mathrm{d}r} &= \frac{1}{\mathbf{v}}\mathbf{D}\mathbf{w}_0 - \frac{C_1}{\mathbf{v}}\mathbf{D}\mathbf{v}. \end{aligned} \tag{7.3}$$

In order to use QBee, we rewrite (7.3) in the form (5.5), that is, as a formal two-dimensional ($n_d = 2$) ODE system with additional variables $v_{\mathbf{D}}$ and $w_{0,\mathbf{D}}$ instead of $\mathbf{D}\mathbf{v}$ and $\mathbf{D}\mathbf{w}_0$:

$$\begin{aligned} \dot{v} &= w_{0,\mathbf{D}} - C_1 v_{\mathbf{D}}, \\ \dot{w}_0 &= \frac{w_{0,\mathbf{D}}}{v} - C_1\frac{v_{\mathbf{D}}}{v}. \end{aligned}$$

Then we use QBee to search for a discretization-agnostic quadratization of (7.3) as described in section 4. As explained in Remark 5.1, this can be done even though (7.3) is a Laurent polynomial system. The dimension-agnostic quadratization returned by Algorithm 5.3 is

$$\mathbf{w}_1 = \left[\frac{1}{v}, \frac{w_0}{v}\right]^\top \quad \text{and} \quad \mathbf{w}_2 = \left[\frac{\tilde{v}}{v}, \frac{\tilde{w}_0}{v}\right]^\top.$$

We can now specialize this dimension-agnostic quadratization to the matrix $\mathbf{D}$ from (7.2). Since the off-diagonal entries of $\mathbf{D}$ are on the shifted diagonal, $\tilde{v}$ and $\tilde{w}_0$ will be replaced with $\mathbf{S}\mathbf{v}$ and $\mathbf{S}\mathbf{w}_0$, respectively, where $\mathbf{S}$ denotes the cyclic shift operator sending any vector $[a_1, \ldots, a_n]^\top$ to $[a_n, a_1, \ldots, a_{n-1}]^\top$. Thus, we will obtain the following $4n$ variables:

$$\mathbf{w}_{1,1} = \frac{1}{\mathbf{v}}, \quad \mathbf{w}_{1,2} = \frac{\mathbf{w}_0}{\mathbf{v}}, \quad \mathbf{w}_{2,1} = \frac{\mathbf{S}\mathbf{v}}{\mathbf{v}}, \quad \mathbf{w}_{2,2} = \frac{\mathbf{S}\mathbf{w}_0}{\mathbf{v}}.$$

Let $C_2 := \frac{\Omega_{\text{rot}}(\hat{\theta})}{\Delta\phi}$; then the quadratic system in these new variables is

$$\begin{aligned} \frac{\mathrm{d}\mathbf{w}_{1,1}}{\mathrm{d}r} &= -C_2\mathbf{w}_{1,1} \odot \mathbf{W}, \\ \frac{\mathrm{d}\mathbf{w}_{1,2}}{\mathrm{d}r} &= C_2\mathbf{W} \odot (\mathbf{w}_{1,1} - \mathbf{w}_{1,2}), \\ \frac{\mathrm{d}\mathbf{w}_{2,1}}{\mathrm{d}r} &= C_2\mathbf{w}_{2,1} \odot (\mathbf{S}\mathbf{W} - \mathbf{W}), \\ \frac{\mathrm{d}\mathbf{w}_{2,2}}{\mathrm{d}r} &= C_2\mathbf{w}_{1,1} \odot \mathbf{S}\mathbf{W} - C_2\mathbf{w}_{2,2} \odot \mathbf{W}, \end{aligned} \tag{7.4}$$

where we denote $\mathbf{W} = \mathbf{w}_{2,2} - \mathbf{w}_{1,2} + C_1\mathbf{1} - C_1\mathbf{w}_{2,1}$ and $\mathbf{1} = [1, \ldots, 1]^\top \in \mathbb{R}^n$. Furthermore, we see that the equations (7.4) do not involve $\mathbf{v}$ and $\mathbf{w}_0$. Since the values of both $\mathbf{v}$ and $\mathbf{w}_0 = \ln(\mathbf{v})$ can be computed from $\mathbf{w}_{1,1} = \frac{1}{\mathbf{v}}$, we henceforth only work with equations (7.4) in

variables $\mathbf{w}_{1,1}, \mathbf{w}_{1,2}, \mathbf{w}_{2,1}, \mathbf{w}_{2,2}$. We observe that system (7.4) can be further reduced by hand via introducing $\mathbf{w}_3 = \mathbf{w}_{2,2} - \mathbf{w}_{1,2}$. Then $\mathbf{W}$ can be written as $\mathbf{w}_3 + C_1\mathbf{1} - C_1\mathbf{w}_{2,1}$ and

$$\begin{aligned}\frac{\mathrm{d}\mathbf{w}_3}{\mathrm{d}r} &= C_2\mathbf{w}_{1,1} \odot \mathbf{S}\mathbf{W} - C_2\mathbf{w}_{2,2} \odot \mathbf{W} - C_2\mathbf{W} \odot (\mathbf{w}_{1,1} - \mathbf{w}_{1,2}) \\ &= C_2\mathbf{w}_{1,1} \odot (\mathbf{S}\mathbf{W} - \mathbf{W}) - C_2\mathbf{w}_3 \odot \mathbf{W}.\end{aligned}$$

Therefore, we can replace the $4n$-dimensional system (7.4) with a $3n$-dimensional

$$\begin{aligned}\frac{\mathrm{d}\mathbf{w}_{1,1}}{\mathrm{d}r} &= -C_2\mathbf{w}_{1,1} \odot \mathbf{W}, \\ \frac{\mathrm{d}\mathbf{w}_{2,1}}{\mathrm{d}r} &= C_2\mathbf{w}_{2,1} \odot (\mathbf{S}\mathbf{W} - \mathbf{W}), \\ \frac{\mathrm{d}\mathbf{w}_3}{\mathrm{d}r} &= C_2\mathbf{w}_{1,1}(\mathbf{S}\mathbf{W} - \mathbf{W}) - C_2\mathbf{w}_3 \odot \mathbf{W},\end{aligned} \tag{7.5}$$

where $\mathbf{W} = \mathbf{w}_3 + C_1\mathbf{1} - C_1\mathbf{w}_{2,1}$. Note that this system still contains $\mathbf{w}_{1,1} = \frac{1}{\mathbf{v}}$, so the original trajectory for v can be reconstructed once (7.5) is solved. We can rewrite this system compactly in quadratic form as

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{H}\left(\mathbf{x} \otimes' \mathbf{x}\right), \tag{7.6}$$

where $\otimes'$ denotes the compact Kronecker product, $\mathbf{x} = \mathbf{x}(r) = \begin{bmatrix}\mathbf{w}_{1,1}^\top(r) & \mathbf{w}_{2,1}^\top(r) & \mathbf{w}_3^\top(r)\end{bmatrix}^\top \in \mathbb{R}^{3n}$, $\mathbf{A} \in \mathbb{R}^{3n\times 3n}$ is the linear operator, and $\mathbf{H} \in \mathbb{R}^{3n\times\frac{1}{2}(3n)(3n+1)}$ is the quadratic operator. In sum, we started with an n-dimensional nonpolynomial (with logarithmic terms) system (7.1) and through QBee are able to replace it with a $3n$-dimensional quadratic system (7.5).

7.3. Data-driven reduced-order model implementation details. We derived the lifted system in (7.5) with the goal to learn a reduced-order model (ROM) from simulated trajectory data—we do not simulate the lifted system. We learn a ROM via the Operator Inference [52] framework and use Python package version 1.2.1 [1] to implement the model learning and prediction. The numerical results in this section are shown for Carrington Rotation 2210, which occurred from 26 October to 23 November 2018, during solar minimum. For more details of the ROM method for this solar wind application, see [35].

We next provide some numerical details. The state vector of the full-order model which we simulate to generate data is $\mathbf{v} \in \mathbb{R}^{129}$. We take $n_r = 400$ uniform steps in the r variables for discretization. We simulate the full-order model via the forward-Euler scheme and the ROM via an implicit multistep variable method based on a backward differentiation formula using the `scipy.integrate.solve_ivp()` Python function. The implemented method is based on a shift of the independent variable to account for advection. We compute the shift function $c(r)$ and its derivative ξ via the cross-correlation extrapolation method; see [35] for more details. Note that the training data is obtained by shifting $\mathbf{v}(r)$ instead of directly solving the shifted-lifted equations. Note that the nonlifted ROM is trained on velocity data in units of km/s, and the lifted ROM is trained on data in units of AU/days. We choose to change the units in order to avoid very large or small scales of the lifted variables. The change in units also indicates the change in the scales of the regularization coefficients.

7.4. Numerical results. Here, we seek to provide data to answer the following question: *"Does lifting the nonlinear system* (7.1) *to quadratic form, i.e.,* (7.5)*, improve the ROM learning?"* We set to answer this question by comparing the numerical results of a quadratic ROM learned from trajectories of the original states, $\mathbf{v}$ from (7.1), to a quadratic ROM learned from the lifted state variables, $\mathbf{w}_{1,1}, \mathbf{w}_{2,1}, \mathbf{w}_3$ in (7.5).

We use 70% of our total snapshots (280 snapshots) for training to compute the basis of the low-dimensional model and 30% for testing (120 snapshots). Therefore, the training domain is $[0.14\text{AU}, 0.813\text{AU}]$ and the testing domain is $[0.813\text{AU}, 1.1\text{AU}]$. We choose the reduced dimension to be $\ell = 6$ for all learned ROMs. The ROM learning framework requires regularization for each of the learned matrices/tensors, i.e., $\mathbf{A}$ and $\mathbf{H}$ require regularization coefficients λ_1, λ_2. We choose those from the logarithmically spaced set, i.e., $\{10^0, 10^1, \ldots, 10^9\}$, such that the best coefficients minimize the mean relative error over the training regime. The optimal coefficients for the shifted ROM in the training regime are $\{\lambda_1 = 1, \lambda_2 = 10^4\}$ and the optimal coefficients for the shifted lifted ROM in the training regime are $\{\lambda_1 = 1, \lambda_2 = 10\}$.

Figure 7.1 shows the solutions obtained from both quadratic ROMs, the one learned from nonlifted data and the one from lifted data. In both cases, the numerical results show that overall both methods are accurate in the training data (less than 1% relative error). However, the ROM learned from the original data becomes unstable and produces spurious solutions (note the different error bars). We compare the relative error in the training and testing regime in Figure 7.2. While both methods are almost equally accurate in the training domain, the

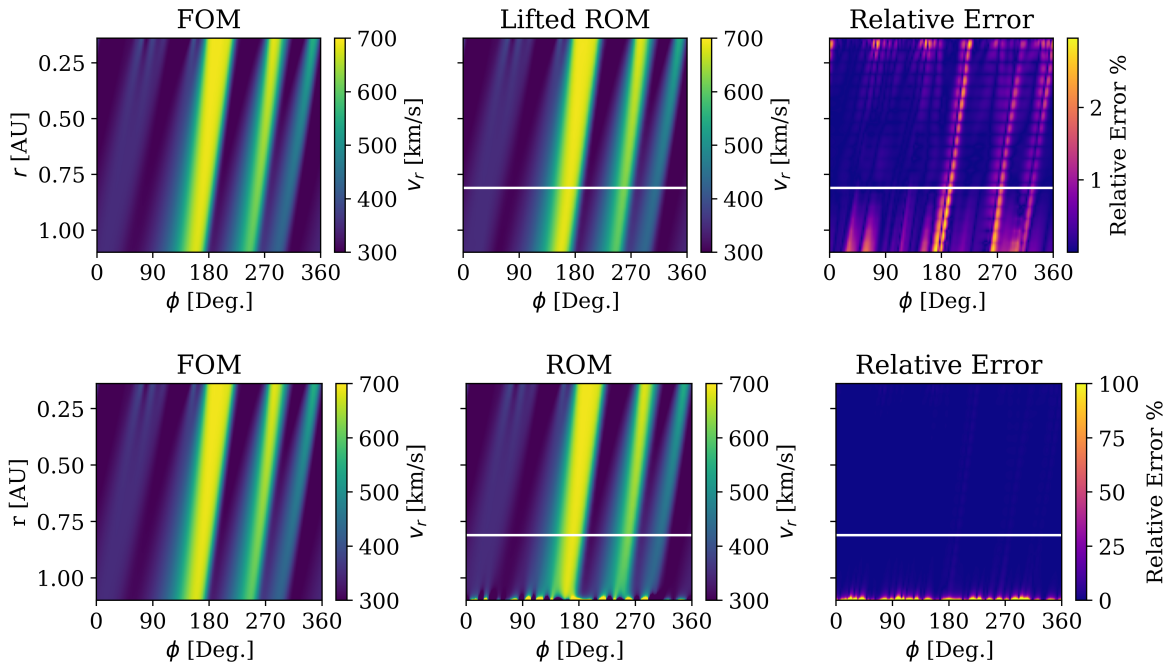

Figure 7.1. *(Top) Solution of the quadratic ROM in the lifted variables, of the form* $\dot{\mathbf{x}}(r) = \mathbf{A}\mathbf{x}(r) + \mathbf{H}[\mathbf{x}(r) \otimes' \mathbf{x}(r)]$ *from* (7.6)*. (Bottom) Solution of the quadratic ROM in the original variables, of the form* $\dot{\mathbf{v}}(r) = \mathbf{A}\mathbf{v}(r) + \mathbf{H}[\mathbf{v}(r) \otimes' \mathbf{v}(r)]$.

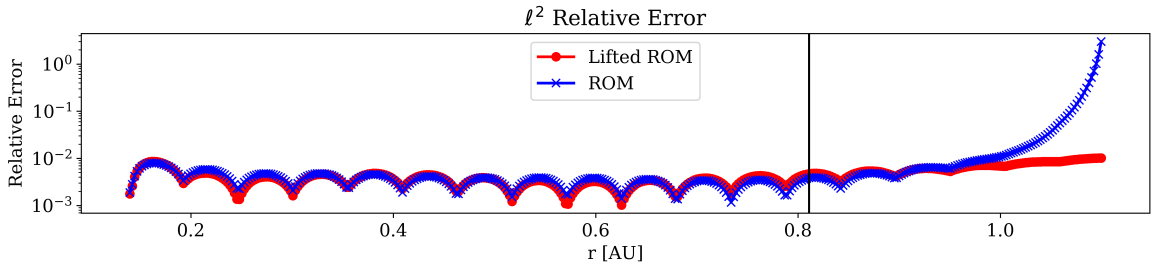

Figure 7.2. *Relative error of the learned quadratic ROMs in both lifted variables and original variables.*

quadratic ROM learned from the original variables produces unstable results in the testing data—it is apparent that lifting the dynamics improves the ROM's accuracy in the testing regime. In conclusion, for predictive computation, the variable lifting makes a significant difference.

8. Conclusions and outlook. We presented novel theory for quadratization of nonautonomous ODEs and ODEs with arbitrary dimension. In particular, we provided existence results, depending on the regularity of the input function, for cases when a polynomial control-affine system can be rewritten as a quadratic-bilinear system. In another thrust, we also developed an algorithm that generalizes the process of quadratization for systems with arbitrary dimension that retain the nonlinear structure when the dimension grows. A specific example is semidiscretized PDEs, where the nonlinear terms remain identical when the discretization size increases. As an important aspect of this research, we extended the QBee software with capabilities for nonautonomous systems of ODEs as well as for ODEs with arbitrary dimension, with rigorous optimality guarantees in the former case. We presented a suite of ODEs that were previously reported in the literature, and where our new algorithms outperform previously reported lifting transformations. We also highlight an important area of lifting transformations: reduced-order model learning can benefit significantly in working in the correct and optimal lifting variables, where learning quadratic models is relatively straightforward.

We anticipate that this research will influence various disciplines and different use cases that rely on quadratization, as mentioned in the introduction. Particularly, the research will spur new directions in data-driven modeling, as novel quadratizations of nonlinear dynamical systems are discovered, which can then be used for model learning. Moreover, this research will make feasible system-theoretic model reduction for a larger class of nonlinear systems, as the optimal quadratizations are the starting point for quadratic-bilinear model reduction.

Despite this progress, there remain several open challenges. First, there is a need for more theory (existence and optimality) of quadratization of nonpolynomial systems of ODEs, alongside with practical and fast algorithms to find those. Second, an interesting question that we currently investigate is whether the symbolic computing tools used for ODEs can be carried over to the PDE setting. Third, since an optimal quadratization is typically not unique, a natural question to ask is how to find optimal quadratizations with attractive numerical properties (stable, preserves equilibria, etc.).

Acknowledgments. We thank the anonymous referee for the valuable suggestion to add more examples to the manuscript. B.K. would also like to thank the Isaac Newton Institute for Mathematical Sciences, Cambridge, for support and hospitality during the program "The Mathematical and Statistical Foundation of Future Data-Driven Engineering," where part of the work on this paper was undertaken.

REFERENCES

[1] *Operator Inference package (version* 0.4.2*)*, https://pypi.org/project/opinf/ (accessed 22 February 2023).
[2] *Biocham4: Biochemical Abstract Machine*, 2021, http://contraintes.inria.fr/biocham/.
[3] F. Alauddin, *Quadratization of ODEs: Monomial vs. non-monomial*, Undergrad. Res. Online (SIURO), 14 (2021), https://doi.org/10.1137/20s1360578.
[4] G. G. Appelroth, *General form for a system of algebraic differential equations*, Sb. Math., 23 (1902), pp. 12–23 (in Russian), http://mi.mathnet.ru/sm6683.
[5] P. Astrid, S. Weiland, K. Willcox, and T. Backx, *Missing point estimation in models described by proper orthogonal decomposition*, IEEE Trans. Automat. Control, 53 (2008), pp. 2237–2251.
[6] M. Balajewicz, I. Tezaur, and E. Dowell, *Minimal subspace rotation on the Stiefel manifold for stabilization and enhancement of projection-based reduced order models for the compressible Navier–Stokes equations*, J. Comput. Phys., 321 (2016), pp. 224–241.
[7] M. Barrault, Y. Maday, N. C. Nguyen, and A. T. Patera, *An empirical interpolation method: Application to efficient reduced-basis discretization of partial differential equations*, C. R. Math., 339 (2004), pp. 667–672.
[8] P. Benner and T. Breiten, *Two-sided projection methods for nonlinear model order reduction*, SIAM J. Sci. Comput., 37 (2015), pp. B239–B260, https://doi.org/10.1137/14097255X.
[9] P. Benner, P. Goyal, and S. Gugercin, *$\mathcal{H}_2$-quasi-optimal model order reduction for quadratic-bilinear control systems*, SIAM J. Matrix Anal. Appl., 39 (2018), pp. 983–1032, https://doi.org/10.1137/16M1098280.
[10] M. Bergmann, C. Bruneau, and A. Iollo, *Enablers for robust POD models*, J. Comput. Phys., 228 (2009), pp. 516–538.
[11] O. Bournez, M. L. Campagnolo, D. S. Graça, and E. Hainry, *Polynomial differential equations compute all real computable functions on computable compact intervals*, J. Complexity, 23 (2007), pp. 317–335.
[12] L. Brenig, *Reducing nonlinear dynamical systems to canonical forms*, Philos. Trans. Roy. Soc. A, 376 (2018), 20170384.
[13] A. Bychkov and G. Pogudin, *Optimal monomial quadratization for ODE systems*, in Combinatorial Algorithms, P. Flocchini and L. Moura, eds., Springer, Cham, 2021, pp. 122–136.
[14] A. Bychkov and G. Pogudin, *QBee*, 2021, https://github.com/AndreyBychkov/QBee.
[15] K. Carlberg, C. Farhat, J. Cortial, and D. Amsallem, *The GNAT method for nonlinear model reduction: Effective implementation and application to computational fluid dynamics and turbulent flows*, J. Comput. Phys., 242 (2013), pp. 623–647.
[16] D. C. Carothers, G. E. Parker, J. S. Sochacki, and P. G. Warne, *Some properties of solutions to polynomial systems of differential equations*, Electron. J. Differential Equations, 2005 (2005), pp. 1–17.
[17] F. Carravetta, *Global exact quadratization of continuous-time nonlinear control systems*, SIAM J. Control Optim., 53 (2015), pp. 235–261, https://doi.org/10.1137/130915418.
[18] F. Carravetta, *On the solution calculation of nonlinear ordinary differential equations via exact quadratization*, J. Differential Equations, 269 (2020), pp. 11328–11365.
[19] S. Chaturantabut and D. C. Sorensen, *Nonlinear model reduction via discrete empirical interpolation*, SIAM J. Sci. Comput., 32 (2010), pp. 2737–2764, https://doi.org/10.1137/090766498.
[20] J. D. Cole, *On a quasi-linear parabolic equation occurring in aerodynamics*, Quart. Appl. Math., 9 (1951), pp. 225–236.
[21] A. V. D. Essen, *Locally finite and locally nilpotent derivations with applications to polynomial flows and polynomial morphism*, Proc. Amer. Math. Soc., 116 (1992), pp. 861–871.

[22] F. Fages, G. Le Guludec, O. Bournez, and A. Pouly, *Strong Turing completeness of continuous chemical reaction networks and compilation of mixed analog-digital programs*, in Computational Methods in Systems Biology: 15th International Conference, CMSB 2017, Springer, 2017, pp. 108–127.
[23] I. V. Gosea and A. C. Antoulas, *Data-driven model order reduction of quadratic-bilinear systems*, Numer. Linear Algebra Appl., 25 (2018), e2200.
[24] C. Gu, *QLMOR: A projection-based nonlinear model order reduction approach using quadratic-linear representation of nonlinear systems*, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst., 30 (2011), pp. 1307–1320.
[25] L. Guillot, B. Cochelin, and C. Vergez, *A generic and efficient Taylor series–based continuation method using a quadratic recast of smooth nonlinear systems*, Internat. J. Numer. Methods Engrg., 119 (2019), pp. 261–280.
[26] R. Haberman, *Applied Partial Differential Equations with Fourier Series and Boundary Value Problems*, Pearson, 2019.
[27] F. D. Halpern, I. Sfiligoi, M. Kostuk, R. Stefan, and R. E. Waltz, *Simulations of plasmas and fluids using anti-symmetric models*, J. Comput. Phys., 445 (2021), 110631.
[28] S. Hassler, A. M. Ranno, and M. Behr, *Finite-element formulation for advection–reaction equations with change of variable and discontinuity capturing*, Comput. Methods Appl. Mech. Engrg., 369 (2020), 113171.
[29] R. F. Heinemann and A. B. Poore, *Multiplicity, stability, and oscillatory dynamics of the tubular reactor*, Chem. Eng. Sci., 36 (1981), pp. 1411–1419.
[30] M. Hemery, F. Fages, and S. Soliman, *On the complexity of quadratization for polynomial differential equations*, in Computational Methods in Systems Biology, A. Abate, T. Petrov, and V. Wolf, eds., Springer, Cham, 2020, pp. 120–140.
[31] M. Hemery, F. Fages, and S. Soliman, *Compiling elementary mathematical functions into finite chemical reaction networks via a polynomialization algorithm for ODEs*, in Computational Methods in Systems Biology, E. Cinquemani and L. Paulevé, eds., Springer, Cham, 2021, pp. 74–90.
[32] E. Hopf, *The partial differential equation* $u_t + uu_x = \mu_{xx}$, Comm. Pure Appl. Math., 3 (1950), pp. 201–230.
[33] C. Huang, J. Xu, K. Duraisamy, and C. Merkle, *Exploration of reduced-order models for rocket combustion applications*, in 2018 AIAA Aerospace Sciences Meeting, American Institute of Aeronautics and Astronautics, 2018, 1183.
[34] T. J. Hughes, L. Franca, and M. Mallet, *A new finite element formulation for computational fluid dynamics:* I. *Symmetric forms of the compressible Euler and Navier-Stokes equations and the second law of thermodynamics*, Comput. Methods Appl. Mech. Engrg., 54 (1986), pp. 223–234.
[35] O. Issan and B. Kramer, *Predicting solar wind streams from the inner-heliosphere to Earth via shifted operator inference*, J. Comput. Phys., 3473 (2022), 111689.
[36] P. Jain, S. A. McQuarrie, and B. Kramer, *Performance comparison of data-driven reduced models for a single-injector combustion process*, in AIAA Propulsion and Energy Forum and Exposition, American Institute of Aeronautics and Astronautics, 2021.
[37] B. Jakubczyk and W. Respondek, *On linearization of control systems*, Bull. Acad. Polonaise Sci. Ser. Sci. Math., 28 (1980), pp. 517–522.
[38] I. Kalashnikova and M. Barone, *Stable and efficient Galerkin reduced order models for non-linear fluid flow*, in 6th AIAA Theoretical Fluid Mechanics Conference, American Institute of Aeronautics and Astronautics, 2011, 3110.
[39] E. H. Kerner, *Universal formats for nonlinear ordinary differential systems*, J. Math. Phys., 22 (1981), pp. 1366–1371.
[40] H. K. Khalil, *Nonlinear Systems*, 3rd ed., Prentice-Hall, Upper Saddle River, NJ, 2002.
[41] B. Kramer, *Stability domains for quadratic-bilinear reduced-order models*, SIAM J. Appl. Dyn. Syst., 20 (2021), pp. 981–996, https://doi.org/10.1137/20M1364849.
[42] B. Kramer and K. Willcox, *Nonlinear model order reduction via lifting transformations and proper orthogonal decomposition*, AIAA J., 57 (2019), pp. 2297–2307, https://doi.org/10.2514/1.J057791.
[43] B. Kramer and K. Willcox, *Balanced truncation model reduction for lifted nonlinear systems*, in Realization and Model Reduction of Dynamical Systems: A Festschrift in Honor of the 70th Birthday

of Thanos Antoulas, C. Beattie, P. Benner, M. Embree, S. Gugercin, and S. Lefteriu, eds., Springer, Cham, 2022, pp. 157–174, https://doi.org/10.1007/978-3-030-95157-3_9.

[44] M. N. Lagutinskii, *On the simplest forms of a system of ordinary differential equations*, Sb. Math., 27 (1911), pp. 420–423 (in Russian), http://mi.mathnet.ru/sm6583.

[45] J. Liu, N. Zhan, H. Zhao, and L. Zou, *Abstraction of elementary hybrid systems by variable transformation*, in International Symposium on Formal Methods, Springer, 2015, pp. 360–377.

[46] G. P. McCormick, *Computability of global solutions to factorable nonconvex programs: Part* I*–convex underestimating problems*, Math. Program., 10 (1976), pp. 147–175.

[47] S. A. McQuarrie, C. Huang, and K. E. Willcox, *Data-driven reduced-order models via regularised operator inference for a single-injector combustion process*, J. Roy. Soc. New Zealand, 51 (2021), pp. 194–211.

[48] D. R. Morrison, S. H. Jacobson, J. J. Sauppe, and E. C. Sewell, *Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning*, Discrete Optim., 19 (2016), pp. 79–102, https://doi.org/10.1016/j.disopt.2016.01.005.

[49] J. Nam, M. Behr, and M. Pasquali, *Space-time least-squares finite element method for convection-reaction system with transformed variables*, Comput. Methods Appl. Mech. Engrg., 200 (2011), pp. 2562–2576.

[50] M. Netto, Y. Susuki, V. Krishnan, and Y. Zhang, *On analytical construction of observable functions in extended dynamic mode decomposition for nonlinear estimation and prediction*, in 2021 American Control Conference (ACC), IEEE, 2021, pp. 4190–4195.

[51] N. Nguyen, A. Patera, and J. Peraire, *A best points interpolation method for efficient approximation of parametrized functions*, Internat. J. Numer. Methods Engrg., 73 (2008), pp. 521–543.

[52] B. Peherstorfer and K. Willcox, *Data-driven operator inference for nonintrusive projection-based model reduction*, Comput. Methods Appl. Mech. Engrg., 306 (2016), pp. 196–215.

[53] E. Qian, B. Kramer, A. Marques, and K. Willcox, *Transform & learn: A data-driven approach to nonlinear model reduction*, in AIAA Aviation and Aeronautics Forum and Exposition, American Institute of Aeronautics and Astronautics, 2019, https://doi.org/10.2514/6.2019-3707.

[54] E. Qian, B. Kramer, B. Peherstorfer, and K. Willcox, *Lift & learn: Physics-informed machine learning for large-scale nonlinear dynamical systems*, Phys. D, 406 (2020), 132401, https://doi.org/10.1016/j.physd.2020.132401.

[55] E. Rezaian and M. Wei, *Impact of symmetrization on the robustness of POD-Galerkin ROMs for compressible flows*, in AIAA Scitech 2020 Forum, American Institute of Aeronautics and Astronautics, 2020, 1318.

[56] P. Riley and R. Lionello, *Mapping solar wind streams from the Sun to* 1 *AU: A comparison of techniques*, Solar Phys., 270 (2011), pp. 575–592, https://doi.org/10.1007/s11207-011-9766-x.

[57] C. W. Rowley, I. Mezić, S. Bagheri, P. Schlatter, and D. S. Henningson, *Spectral analysis of nonlinear flows*, J. Fluid Mech., 641 (2009), pp. 115–127.

[58] M. A. Savageau and E. O. Voit, *Recasting nonlinear differential equations as S-systems: A canonical nonlinear form*, Math. Biosci., 87 (1987), pp. 83–115.

[59] N. Sawant, B. Kramer, and B. Peherstorfer, *Physics-informed regularization and structure preservation for learning stable reduced models from data with operator inference*, Comput. Methods Appl. Mech. Engrg., 404 (2023), 115836.

[60] P. J. Schmid, *Dynamic mode decomposition of numerical and experimental data*, J. Fluid Mech., 656 (2010), pp. 5–28.

[61] R. Swischuk, B. Kramer, C. Huang, and K. Willcox, *Learning physics-based reduced-order models for a single-injector combustion process*, AIAA J., 58 (2020), pp. 2658–2672, https://doi.org/10.2514/1.J058943.

[62] M. O. Williams, I. G. Kevrekidis, and C. W. Rowley, *A data–driven approximation of the Koopman operator: Extending dynamic mode decomposition*, J. Nonlinear Sci., 25 (2015), pp. 1307–1346.

[63] Y. B. Zhou, *Model Reduction for Nonlinear Dynamical Systems with Parametric Uncertainties*, Ph.D. thesis, Massachusetts Institute of Technology, 2012.