Received: February 7, 2025; **Accepted:** May 21, 2025; **Published:** July 1, 2025.
DOI. 10.30473/coam.2025.74498.1307
Summer-Autumn (2025) Volume 10, Issue 2, (135-152)
Research Article

Open Access

Control and Optimization in Applied Mathematics - COAM

A Computer Algebra Approach to Linear ODE Systems with Parametric Coefficients

Mahdi Dehghani Darmian ✉

Department of Basic Sciences, Technical and Vocational University (TVU), Tehran, Iran.

✉ **Correspondence:**
Mahdi Dehghani Darmian
E-mail:
mdehghanidarmian@tvu.ac.ir

How to Cite
Dehghani Darmian, M. (2025). "A computer algebra approach to linear ODE systems with parametric coefficients", Control and Optimization in Applied Mathematics, 10(2): 135-152, doi: 10.30473/coam.2025.74498.1307.

Abstract. This paper analyzes systems of linear first-order ordinary differential equations (ODEs) with parametric coefficients, a class of problems that arises in control theory, optimization, and applied mathematics. We introduce the notion of a comprehensive solution system for such parametric ODEs, constructed using Gröbner systems from computer algebra. Our approach partitions the parameter space into finitely many cells and associates an explicit solution with each cell. Furthermore, we present an algorithm that computes a comprehensive solution system for any given parametric system. To address the computational challenges inherent in Gröbner systems, we adopt the GES algorithm, a parametric variant of Gaussian elimination, which eliminates the need for Gröbner bases. This method builds upon the LDS algorithm proposed in 2017. Both algorithms have been implemented in `Maple`, and we illustrate the structural framework of the main algorithm with a straightforward example. The results highlight the practicality and effectiveness of the proposed methods for solving parametric linear first-order ODE systems.

Keywords. Parametric linear ODE systems, Gröbner system, Gaussian elimination system algorithm, Comprehensive solution system, Parameter space.

MSC. 13P10; 93C15.

1 Introduction

Systems of differential equations are foundational in modeling dynamic processes across control theory, optimization, and applied mathematics [13, 14]. This paper focuses on linear first-order ordinary differential equations (ODEs) with parametric coefficients, a class of problems that could be critical to optimal control design [1], sensitivity analysis [20], and robust stability under parameter variations [28]. Such systems may arise frequently in engineering and scientific applications, where optimizing performance or ensuring stability might benefit from explicit solutions for arbitrary parameter configurations [23].

Traditional methods for solving parametric ODEs face limitations in scalability and generality, particularly when parameters influence system behavior discontinuously. Our work addresses this gap by integrating computational algebra with control-oriented frameworks, enabling systematic analysis of parametric dependencies. Specifically, we focus on systems where the number of unknown functions equals the number of equations, a structure common in state-space control systems [19], expressed as

$$\frac{dx_j}{dt} = a_{j1}(t)x_1 + \cdots + a_{jn}(t)x_n + g_j(t), \quad j = 1, \ldots, n,$$

where $a_{ji}(t)$ and $g_j(t)$ are arbitrary functions of the independent variable t, and n is a positive integer. A first-order linear system of ODEs may be written in matrix form:

$$\frac{d}{dt}\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ a_{21} & \cdots & a_{2n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix} + \begin{bmatrix} g_1 \\ g_2 \\ \vdots \\ g_n \end{bmatrix},$$

or simply

$$\dot{\mathbf{x}}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{g}(t).$$

When the coefficients $a_{ji}(t)$ are constant, the general solution to the system $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{g}(t)$ is the sum of the general solution to the associated homogeneous system $\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t)$ and a particular solution to the non-homogeneous system.

The general solution to the homogeneous system is constructed from the eigenvalues and eigenvectors of the matrix $\mathbf{A}$. If $\mathbf{A}$ is diagonalizable (e.g., when it has n distinct eigenvalues), this solution is given by:

$$\mathbf{x}_h(t) = C_1\mathbf{v}_1 e^{\lambda_1 t} + C_2\mathbf{v}_2 e^{\lambda_2 t} + \cdots + C_n\mathbf{v}_n e^{\lambda_n t},$$

where $\lambda_1, \lambda_2, \ldots, \lambda_n$ are the eigenvalues and $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ are the corresponding eigenvectors. Cases with fewer distinct eigenvalues required different treatment.

A particular solution, $\mathbf{x}_p(t)$, to the non-homogeneous system can be found by methods such as undetermined coefficients or variation of parameters. The general solution is then:

$$\mathbf{x}(t) = \mathbf{x}_h(t) + \mathbf{x}_p(t).$$

Several methods exist to solve systems of linear first-order differential equations, including the matrix method (finding the eigenvalues of A), the elimination method, the operator method D, the Laplace transform method, and so on [14]. Furthermore, Ritt's characteristic set method, developed by Ritt

[21, 22] and Wu [26, 27], offers a constructive approach to solving polynomial systems, including differential equations. This method analyzes and solves these systems by decomposing them into simpler, triangular forms known as ascending chains.

Any differential algebraic system's zero-set decomposes into chains' zero-sets. Our method's key advantage lies in its ability to handle differential equations with parametric coefficients, particularly in finding solutions for first-order parametric ODEs.

Numerous engineering and scientific problems can be modeled using differential equations with parametric coefficients, necessitating repeated analysis for various parameter values. Therefore, this paper focuses solely on the solution of first-order parametric ODEs, which has not, to our knowledge, been explored in the literature. The following example shows that the mentioned traditional solution approach may not be used for such systems of ODEs.

Example 1. Consider the first-order linear system of equations where a, b, c, d are parameters in $\mathbb{R}$:

$$sys = \begin{cases} \dot{x}(t) = (a-1)x(t) + by(t) + d, \\ \dot{y}(t) = (b-c)x(t) + ay(t) - 1. \end{cases}$$

The solution set is computed using one of the above well-known methods and the function `PDEtools:-dsolve(sys)` in `Maple 18`:

$$x := e^{\left(a-\frac{1}{2}+\frac{1}{2}\sqrt{4b^2-4bc+1}\right)t} C_2 + e^{\left(a-\frac{1}{2}-\frac{1}{2}\sqrt{4b^2-4bc+1}\right)t} C_1 + \frac{-ad-b}{a^2-b^2+bc-a}$$

$$\begin{aligned} y := \frac{1}{2}\frac{1}{(a^2-b^2+bc-a)\,b}\Big(& e^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t} C_2 \sqrt{4b^2-4bc+1}\,(a^2-b^2) \\ & + e^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t} C_2 \sqrt{4b^2-4bc+1}\,bc - e^{-\frac{1}{2}t\left(-2a+1+\sqrt{4b^2-4bc+1}\right)} C_1 \sqrt{4b^2-4bc+1}\,a^2 \\ & + e^{-\frac{1}{2}t\left(-2a+1+\sqrt{4b^2-4bc+1}\right)} C_1 \sqrt{4b^2-4bc+1}\,b^2 - e^{-\frac{1}{2}t\left(-2a+1+\sqrt{4b^2-4bc+1}\right)} C_1 \sqrt{4b^2-4bc+1}\,bc \\ & - e^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t} C_2 \sqrt{4b^2-4bc+1}\,a + e^{-\frac{1}{2}t\left(-2a+1+\sqrt{4b^2-4bc+1}\right)} C_1 \sqrt{4b^2-4bc+1}\,a \\ & + e^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t} C_2 a^2 - e^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t} C_2 b^2 + e^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t} C_2 bc \\ & + e^{\frac{1}{2}t\left(-2a+1+\sqrt{4b^2-4bc+1}\right)} C_1 a^2 - e^{\frac{1}{2}t\left(-2a+1+\sqrt{4b^2-4bc+1}\right)} C_1 b^2 + e^{-\frac{1}{2}t\left(-2a+1+\sqrt{4b^2-4bc+1}\right)} C_1 bc \\ & - e^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t} C_2 a - e^{-\frac{1}{2}t\left(-2a+1+\sqrt{4b^2-4bc+1}\right)} C_1 a + 2b^2 d - 2bcd + 2ba - 2b \Big) \end{aligned}$$

Figure 1: Solution set of the parametric system of ODEs computed using `Maple`.

Substituting $a = 1, b = 2, c = 2$, and $d = -1$ yields:

$$sys|_{a=1,b=2,c=2,d=-1} = \begin{cases} \dot{x}(t) = 2y(t) - 1, \\ \dot{y}(t) = y(t) - 1, \end{cases}$$

resulting in the solution set:

$$\{x(t) = 2C_2 e^t + t + C_1,\ y(t) = C_2 e^t + 1\}. \tag{1}$$

However, this evaluation of parameters a, b, c, d does not align with x and y, the `Maple` output in Figure 1, since these are undefined for $a = 1, b = 2, c = 2$, and $d = -1$.

One may be interested in determining the values of parameters s.t. $\sigma(x(t))$ and $\sigma(y(t))$ are the solution of $\sigma(sys)$ where $\sigma : \mathbb{R}[\mathbf{a}] \to \mathbb{R}' \supseteq \mathbb{R}$ is a specialization involving parameter assignments; more especially, one may seek to understand the solution space structure. In this direction, we introduce the concept of a *comprehensive solution system* for a system of parametric first-order ODEs and design an algorithm for its computation, utilizing the key idea of *Gröbner systems*.

2 Gröbner Systems

Gröbner systems and their algorithms were introduced by Weispfenning in 1992 [25]. For a parametric polynomial ideal, computing its Gröbner system partitions the parameter space into a finite set of cells, each linked to a specific set of polynomials. By identifying the cell corresponding to given parameter values, we obtain a Gröbner basis corresponding to those values.
Let $\mathcal{R} = \mathbb{K}[x_1, \ldots, x_n]$ denote the polynomial ring in variables $x_1, \ldots, x_n$ over a field $\mathbb{K}$. For $g \in \mathcal{R}$, denote by $\mathrm{LM}_\prec(g)$, $\mathrm{LC}_\prec(g)$, and $\mathrm{LT}_\prec(g)$ the leading monomial, leading coefficient, and leading term of g w.r.t. the monomial ordering $\prec$, respectively. Given an ideal $\mathcal{J} = \langle f_1, \ldots, f_k \rangle \subset \mathcal{R}$, the leading monomial ideal of $\mathcal{J}$ is $\mathrm{LM}_\prec(\mathcal{J}) = \langle \mathrm{LM}_\prec(g) \mid g \in \mathcal{J} \rangle$. finally, $G = \{g_1, \ldots, g_m\} \subset \mathcal{J}$ is a Gröbner basis for $\mathcal{J}$ w.r.t. $\prec$ if $\mathrm{LM}_\prec(\mathcal{J}) = \langle \mathrm{LM}_\prec(g_1), \ldots, \mathrm{LM}_\prec(g_m) \rangle$. See [2] for further details concerning Gröbner bases.

Using these notations, we define Gröbner bases for ideals with polynomial generators that include parametric coefficients, referred to as Gröbner systems. Gröbner systems are widely utilized in mathematics, science, and engineering, playing crucial roles in areas such as parametric linear algebra [5, 6, 10], automated geometry theorem proving [15], robotics [16, 17], algebraic geometry [8, 15, 17, 25], and electrical networks [17, 18], and more. These applications often require repeated analysis with varying parameter values.

Essentially, Gröbner systems extend the concept of Gröbner bases for polynomial ideals over fields to those with parametric coefficients. Consider the polynomial ring $\mathcal{S} = \mathbb{K}[\mathbf{a}, \mathbf{x}]$, where $\mathbf{a} = (a_1, \ldots, a_m)$ are parameters and $\mathbf{x} = (x_1, \ldots, x_n)$ are variables. The sets $\{\mathbf{x}\}$ and $\{\mathbf{a}\}$ are disjoint. Hence, any term in $\mathcal{S}$ takes the form $\mathbf{a}^\gamma \mathbf{x}^\alpha$, where $\mathbf{a}^\gamma \in \mathbb{K}[\mathbf{a}]$ acts as a coefficient of $\mathbf{x}^\alpha$. We define two monomial orderings, $\prec_\mathbf{x}$ for variables and $\prec_\mathbf{a}$ for parameters. To compute Gröbner systems, we establish a product order $\prec_{\mathbf{x},\mathbf{a}}$, defined as follows: for any $\gamma, \delta \in \mathbb{N}^m$ and $\alpha, \beta \in \mathbb{N}^n$, we have $\mathbf{a}^\gamma \mathbf{x}^\alpha \prec_{\mathbf{x},\mathbf{a}} \mathbf{a}^\delta \mathbf{x}^\beta$ if either $\mathbf{x}^\alpha \prec_\mathbf{x} \mathbf{x}^\beta$ or both $\mathbf{x}^\alpha = \mathbf{x}^\beta$ and $\mathbf{a}^\gamma \prec_\mathbf{a} \mathbf{a}^\delta$. Moreover, a specialization of parameters is a morphism $\sigma : \mathbb{K}[\mathbf{a}] \to \overline{\mathbb{K}}$, where $\overline{\mathbb{K}}$ denotes the algebraic closure of $\mathbb{K}$. A parametric polynomial ideal consists of a system of parametric polynomial equations generated by polynomials with parametric coefficients. Additionally, a parametric linear polynomial is defined as a polynomial where the power of each variable x_i is at most one, allowing parametric coefficients to be polynomials of any power in $\mathbb{K}[\mathbf{a}]$.

Let us revisit the definition of a Gröbner system for polynomial ideals with parametric coefficients, where Weispfenning established their existence for all such ideals [25, Proposition 3.4 and Theorem 2.7] and proposed an algorithm for their computation [25, Theorem 3.6]. Since then, various algorithms have

been developed to compute these Gröbner systems, each with its advantages and drawbacks [3, 4, 6, 7, 8, 9, 11, 12, 17, 24].

Definition 1. Let $\mathcal{G} = \{(N_i, W_i, G_i)\}_{i=1}^{\ell}$ be a set of triples where $G_i \subset \mathcal{S}$ and (N_i, W_i) is a conditions pair in $\mathbb{K}[\mathbf{a}] \times \mathbb{K}[\mathbf{a}]$ for $i = 1, \ldots, \ell$. The set $\mathcal{G}$ is a Gröbner system for $\langle F \rangle \subset \mathcal{S}$ with respect to $\prec_{\mathbf{x},\mathbf{a}}$ over $\mathcal{V} \subseteq \overline{\mathbb{K}}^m$ if, for each i, the following holds:

- $\sigma(G_i)$ is a Gröbner basis of $\langle \sigma(F) \rangle$ w.r.t. $\prec_{\mathbf{x}}$, for any specialization $\sigma : \mathbb{K}[\mathbf{a}] \to \overline{\mathbb{K}}$ satisfying (N_i, W_i),
- $\mathcal{V} \subseteq \bigcup_{i=1}^{\ell} \mathbb{V}(N_i) \setminus \mathbb{V}(W_i)$, where $\mathbb{V}(N)$ denotes the common zeros of a set of generators for N.

Each (N_i, W_i, G_i) is referred to as a segment or branch of the Gröbner system $\mathcal{G}$ for $1 \leq i \leq \ell$. Each pair (N_i, W_i) is called a specification, with N_i and W_i referred to as the null and non-null condition sets, respectively. $\mathcal{G}$ is generally termed a Gröbner system of F, if affine variety $\mathcal{V} = \overline{\mathbb{K}}^m$. A condition pair (N, W) is deemed inconsistent if, for every polynomial $f \in W$, f lies in the radical ideal generated by N. Consistency of parametric constraints can be efficiently determined by combining the ICheck and CCheck algorithms (Kapur, et al. [12]) with Rabinovitch's trick for radical membership testing, which involves introducing a new variable in worst-case scenarios. See [12, Section 5] for details.

It is worth noting that there are two general approaches to computing Gröbner systems. One approach branches in the parameter space during Gröbner basis construction in the variable ring. The other computes a Gröbner basis in the combined variable and parameter ring at each iteration, branching based on the leading coefficients in terms of the parameters. In both cases, the parameter space is automatically partitioned according to the algorithm's structure.

Example 2. Let

$$F = \{(a_1 - 1)x_1x_3 - a_3x_2^2, -a_2x_2 + x_1^2 + 1\} \subset \mathbb{K}[a_1, a_2, a_3][x_1, x_2, x_3],$$

where a_1, a_2, a_3 are parameters and x_1, x_2, x_3 are variables. Using our `Maple` implementation of the PF_4 algorithm [4], we compute a Gröbner system for $\langle F \rangle$ with respect to the lexicographic term order

$$x_3 \prec x_2 \prec x_1 \quad \text{and} \quad a_3 \prec a_2 \prec a_1.$$

The following Gröbner system has five branches while assuming $a_3 = 3$, $a_2 = 2$, and $a_1 = 1$ randomly, the first triple satisfying these parameter values, and so a Gröbner basis of $\langle F \rangle \,|_{a_3=3,a_2=2,a_1=1}$ is $\{a_3x_2^2, -a_2x_2 + x_1^2 + 1\}$.

$$\begin{cases} ([a_1 - 1], [a_3], & [a_3x_2^2, -a_2x_2 + x_1^2 + 1]) \\ ([a_3, a_2], [a_1 - 1], & [a_1x_3 - x_3, x_1^2 + 1]) \\ ([a_3, a_1 - 1], [\,], & [-a_2x_2 + x_1^2 + 1]) \\ ([a_3], [a_2, a_1 - 1], & [a_1a_2x_2x_3 - a_2x_2x_3 - a_1x_3 + x_3, a_1x_1x_3 - x_1x_3, -a_2x_2 + x_1^2 + 1]) \\ ([\,], [a_3^2(a_1 - 1)], & [-a_1^2a_2x_2x_3^2 + a_3^2x_2^4 + 2a_1a_2x_2x_3^2 + a_1^2x_3^2 - a_2x_2x_3^2 - 2a_1x_3^2 + x_3^2, \\ & a_1x_1x_3 - a_3x_2^2 - x_1x_3, -a_1a_2x_2x_3 + a_3x_1x_2^2 + a_2x_2x_3 + a_1x_3 - x_3, \\ & -a_2x_2 + x_1^2 + 1]). \end{cases}$$

This raises the question: "how can the Gröbner system be applied to solving systems of parametric first-order ODEs? "

3 Comprehensive Solution System

In this section, we present a comprehensive solution system for parametric first-order ODEs. We begin by linearizing the ODEs through the substitution of all functional variables and their derivatives with new variables A_i, then rearranging the equations to move the right-hand sides to the left, setting them to zero. Next, we construct an ideal in $\mathbb{R}[\mathbf{a}, \mathbf{A}]$ that corresponds to the ODEs and compute its Gröbner system. Here, $\mathbf{a}$ shows the sequence of parameters, and $\mathbf{A}$ denotes the sequence of new tag variables A_i.

Definition 2. Let sys be a system of first-order ODEs with parametric coefficients. A finite triple set $L = \{(N_i, W_i, S_i)\}_{i=1}^{\ell}$ is defined as a *comprehensive solution system* of sys if, for each i and any specialization $\sigma : \mathbb{R}[\mathbf{a}] \to \mathbb{R}' \supseteq \mathbb{R}$ satisfying the parametric constraints (N_i, W_i), the solution set of $\sigma(S_i)$ is equivalent to the solution set of $\sigma(sys)$, where σ satisfies the parametric conditions (N_i, W_i).

Based on this definition and the preceding discussion, we present the CSS algorithm to compute a comprehensive solution. In this algorithm, the variable **LIST** starts as an empty set and eventually represents a comprehensive solution system. Each entry in **LIST** is a triple (N_i, W_i, S_i), where S_i is the solution set for the input system of parametric ODEs based on (N_i, W_i). We denote the i-th element of L as $L[i]$. The bijective linear transformation ψ maps functional variables m_i and their derivatives to tag variables A_i, with ψ^{-1} performing the inverse mapping.

Theorem 1. The CSS algorithm terminates after a finite number of iterations and accurately computes a comprehensive solution system.

Proof. The termination of the CSS algorithm is guaranteed by the termination of Gröbner system computations. Its correctness is based on established classical methods for solving systems of ODEs with constant coefficients. Specifically, let (N_i, W_i, G_i) represent a triple of the Gröbner system G at the i-th step of the **for**-loop. During this step, all parametric coefficients in G_i are non-zero, allowing us to apply classical methods after transforming G_i back into the original functional variables and their derivatives to obtain S_i, the solution set of $\sigma(sys)$, based on the conditions (N_i, W_i). Thus, each S_i is the solution set of the input system sys under the parametric constraint (N_i, W_i). □

We illustrate this technique with an example.

Example 3. Consider the following system of two parametric first-order ODEs mentioned in Example 1:

$$sys = \begin{cases} \dot{x}(t) = (a-1)x(t) + by(t) + d, \\ \dot{y}(t) = (b-c)x(t) + ay(t) - 1. \end{cases}$$

Initially, we define $F = \{\dot{x}(t) + (1-a)x(t) - by(t) - d = 0, \dot{y}(t) - (b-c)x(t) - ay(t) + 1 = 0\}$. We then apply a bijective linear transformation ψ to obtain the linearization

$$L = \psi(F) = \{A_3 + (1-a)A_1 - bA_2 - dA_5, A_4 - (b-c)A_1 - aA_2 + A_5\}.$$

Using the efficient GES-GVW-CGS algorithms [3], we derive a Gröbner system of L:

Algorithm 4 CSS (Comprehensive Solution System).

Inputs: A system of parametric first-order ODEs

$$sys = \left\{ \frac{dx_j}{dt} = a_{j1}x_1 + \cdots + a_{jn}x_n + g_j(t) \right\}_{j=1}^{n}$$

Outputs: A Comprehensive Solution System of sys.

1. Construct the set
$$F = \left\{ \frac{dx_j}{dt} - (a_{j1}x_1 + \cdots + a_{jn}x_n + g_j(t)) \right\}_{j=1}^{n}.$$
2. Compute a linearization of F, denoted by $L := \psi(F)$.
3. Determine a Gröbner system of L, namely
$$G = \{(N_i, W_i, G_i)\}_{i=1}^{\ell}.$$
4. Initialize an empty list, LIST := $\{\}$.
5. For each $i = 1, \ldots, \ell$, perform the following steps: Define the system
$$F_i' = \{\psi^{-1}(G_i[j]) = 0 \mid j = 1, \ldots, n\}.$$
Solve F_i' using any standard classical method, and denote the solution set by S_i.
Update the list:
$$\text{LIST} := \text{LIST} \cup \{(N_i, W_i, S_i)\}.$$
6. Return LIST.

$$\begin{cases}
([a-1], [c-b, b], & [bA_2 + dA_5 - A_3, bA_1 - cA_1 - A_4 + A_2 - A_5]) \\
([b, 1-a], [c], & [-dA_5 + A_3, cA_1 - A_2 + A_4 + A_5]) \\
([1-a, b, c], [1], & [-dA_5 + A_3, A_2 - A_4 - A_5]) \\
([b-c, a-1], [c], & [-cA_4 - cA_5 - dA_5 + A_3, A_2 - A_5 - A_4]) \\
([a^2 - b^2 + bc - a], [1-a, b-c, b], & [-adA_5 + aA_3 - bA_4 - bA_5, aA_1 + bA_2 + dA_5 - A_1 - A_3]) \\
([b, a], [c], & [-cdA_5 + cA_3 - A_4 - A_5, A_1 - dA_5 + A_3]) \\
([c, b, a], [1], & [A_5 + A_4, -dA_5 + A_1 + A_3]) \\
([b-c, a], [c], & [A_5 + A_4, -cA_2 - dA_5 + A_1 + A_3]) \\
 & \\
([\,], [(1-a)(a^2 - b^2 + bc - a)], & [(a^2 - b^2 + bc - a)A_2 + (1 - bd + cd - a)A_5 + (1-a)A_4 + \\
 & (b-c)A_3, aA_1 + bA_2 + dA_5 - A_1 - A_3]).
\end{cases}$$

Next, we transform all G_i back into the original functions and their derivatives, resulting in a system containing nine triples.

$$\begin{cases}
([a-1],[c-b,b], & [by(t)+d-\dot{x}(t), bx(t)-\dot{y}(t)-1-cx(t)+y(t)]) \\
([b,1-a],[c], & [\dot{x}(t)-d, -y(t)+cx(t)+\dot{y}(t)+1]) \\
([1-a,b,c],[1], & [\dot{x}(t)-d, y(t)-\dot{y}(t)-1]) \\
([b-c,a-1],[c], & [-c\dot{y}(t)-c1-d+\dot{x}(t), y(t)-\dot{y}(t)-1]) \\
([a^2-b^2+bc-a],[b,a-1,b-c], & [-ad+a\dot{x}(t)-b\dot{y}(t)-b1, ax(t)+by(t)-\dot{x}(t)+d-x(t)]) \\
([b,a],[c], & [-cd+c\dot{x}(t)-\dot{y}(t)-1, x(t)-d+\dot{x}(t)]) \\
([c,b,a],[1], & [\dot{y}(t)+1, -d+\dot{x}(t)+x(t)]) \\
([b-c,a],[c], & [\dot{y}(t)+1, -cy(t)+\dot{x}(t)-d+x(t)]) \\
 & \\
([\,],[(1-a)(a^2-b^2+bc-a)], & [(a^2-b^2+bc-a)y(t)+(1-bd+cd-a)+(1-a)\dot{y}(t)+ \\
 & (b-c)\dot{x}(t), ax(t)+by(t)+d-x(t)-\dot{x}(t)]).
\end{cases}$$

The solution sets S_i can be calculated using any established method applied to the third component of the branches (We use the function `PDEtools:-dsolve(sys)` in `Maple 18`).

$$S_1 = \left\{ x(t) = C_2 \mathrm{e}^{\left(\frac{1}{2}+\frac{1}{2}\sqrt{4b^2-4bc+1}\right)t} + C_1 \mathrm{e}^{\left(\frac{1}{2}-\frac{1}{2}\sqrt{4b^2-4bc+1}\right)t} + \frac{b+d}{b(b-c)}, \right.$$
$$\left. y(t) = \frac{1}{2}\frac{-2d+\left(1+\sqrt{4b^2-4bc+1}\right)C_2 \mathrm{e}^{\frac{1}{2}\left(1+\sqrt{4b^2-4bc+1}\right)t} + \left(1-\sqrt{4b^2-4bc+1}\right)C_1 \mathrm{e}^{\frac{1}{2}\left(1-\sqrt{4b^2-4bc+1}\right)t}}{b} \right\}$$

$$S_2 = \left\{ x(t) = C_2 + dt, y(t) = cdt + C_2 c + cd + 1 + C_1 \mathrm{e}^t \right\}$$

$$S_3 = \left\{ x(t) = C_2 + dt, y(t) = 1 + C_1 \mathrm{e}^t \right\}$$

$$S_4 = \left\{ x(t) = C_2 c \mathrm{e}^t + ct + dt + C_1, y(t) = 1 + C_2 \mathrm{e}^t \right\}$$

$$S_5 = \left\{ x(t) = \frac{a^3 dt + C_1 ba \mathrm{e}^{(2a-1)t} - 2C_2 a^2 b + a^2 bt - a^2 dt - C_1 b \mathrm{e}^{(2a-1)t} + C_2 ab - a^2 d - abt + ab + ad}{(2a-1)a(a-1)}, \right.$$
$$\left. y(t) = \frac{-a^2 dt + C_1 b \mathrm{e}^{(2a-1)t} + 2C_2 ab - abt + adt - C_2 b + tb}{b(2a-1)} \right\}$$

$$S_6 = \left\{ x(t) = d + C_2 \mathrm{e}^{-t}, y(t) = C_2 c \mathrm{e}^{-t} - cdt + C_1 - t \right\}$$

$$S_7 = \left\{ x(t) = d + C_1 \mathrm{e}^{-t}, y(t) = -t + C_2 \right\}$$

$$S_8 = \left\{ x(t) = C_2 c - ct + c + d + C_1 \mathrm{e}^{-t}, y(t) = -t + C_2 \right\}$$

$$S_9 = \left\{ x(t) = \frac{a\left(C_2 \mathrm{e}^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t} + C_1 \mathrm{e}^{\frac{1}{2}\left(2a-1-\sqrt{4b^2-4bc+1}\right)t}\right)}{-b+c} + \frac{(c-b)ad+a-a^2}{\left(2a^2-2b^2+2bc-2a\right)(b-c)} + \right.$$
$$\frac{C_2(2a-1+\sqrt{4b^2-4bc+1})\,\mathrm{e}^{\frac{1}{2}\left(2a-1+\sqrt{4b^2-4bc+1}\right)t}}{2(b-c)} + \frac{C_1\left(2a-1-\sqrt{4b^2-4bc+1}\right)\mathrm{e}^{\frac{1}{2}\left(2a-1-\sqrt{4b^2-4bc+1}\right)t}}{2(b-c)} +$$
$$\left. \frac{1}{b-c}, y(t) = C_2 \mathrm{e}^{\left(a-\frac{1}{2}+\frac{1}{2}\sqrt{4b^2-4bc+1}\right)t} + C_1 \mathrm{e}^{\left(a-\frac{1}{2}-\frac{1}{2}\sqrt{4b^2-4bc+1}\right)t} + \frac{(b-c)d+a-1}{a^2-b^2+bc-a} \right\}$$

To conserve space and due to the length and complexity of some branches' solutions, we focus on the fourth branch's solution set, which pertains to the challenging values in Example 1. The solution set for the fourth triple is given with conditions $b-c=0$, $a-1=0$, and $c\neq 0$ (satisfying the parametric constraint (N_4, W_4)):

$$\{x(t) = C_2(ce^t) + (c+d)t + C_1,\ y(t) = C_2 e^t + 1\}.$$

For example, if $a=1$, $b=c=2$, and $d=-1$ (an assignment in (N_4, W_4)), the resulting solution set is:

$$\{x(t) = 2C_2e^t + t + C_1,\ y(t) = C_2e^t + 1\},$$

which coincides with the solution set in (1) from Example 1. This means that for any $1 \leq i \leq 9$ and each specialization σ corresponding to (N_i, W_i), we can obtain a solution set for a system of parametric first-order ODEs, referred to as a comprehensive solution system.

4 Improvement of CSS Algorithm

The computation of comprehensive solution systems through the use of Gröbner systems is not without its challenges and drawbacks. As highlighted by Suzuki and Sato in their work [24], the process of computing Gröbner systems for parametric linear ideals tends to be "generally slow and inefficient." This inefficiency is further illustrated in the comparison table presented on [6, Page 48]. Because of these limitations, we have decided to adopt a slight modification of the GES algorithm, which is detailed in [6, Algorithm 3]. This algorithm functions as a parametric variant of the Gaussian elimination method. and notably avoids the complications associated with Gröbner systems. Instead, we utilize the GES approach to compute comprehensive solution systems that are based on the LDS algorithm introduced in [6], which serves as a foundation for our efforts. As a crucial initial step in this direction, we must provide a brief review of the parametric linear algebra techniques to achieve our objective outlined in effectively cite[Section 3]pfglm. The GES algorithm is designed to return a Gaussian elimination system suitable for a matrix that contains parametric entries, which can be equated to a linear system with parametric coefficients. However, our specific application of this algorithm is geared toward a system of first-order ordinary differential equations (ODEs); through this application, we aim to establish the solutions set concerning the corresponding specifications presented. To execute this process properly, the input ODEs need to be linearized. This linearization involves the crucial step of replacing all the functional variables that appear within the ODEs with newly introduced variables, thereby simplifying the system.

The LDS algorithm [6, Algorithm 2], a crucial sub-algorithm within GES, efficiently addresses the parametric linear dependency check. It determines dependencies within a linear polynomial containing parameters relative to a set of parametric polynomials, bypassing the need for Gröbner systems. This approach allows for a more streamlined analysis of the problem at hand. For the reader's convenience, we present the following example to further illustrate these concepts. Additionally, for those seeking more in-depth information related to the LDS algorithm and its applications, we direct readers to consult [6] for further details and explanations.

Example 4. Consider a linear Gröbner basis $L = [a_1x_5 + x_1, a_2x_4 + x_2, x_3]$ under the conditions pair $(N, W) = ([\,], [a_1 - 1, a_2 - 1, a_3])$. Let $f = (a_1 - 2)x_1 + a_5x_2 + a_3x_3 + du + (3 - a_2)x_5$ be a polynomial with parametric coefficients.

Utilizing our `Maple` implementation of the LDS algorithm, we derive the following linear dependency system of f on L with respect to $a_5 \prec_{lex} a_4 \prec_{lex} a_3 \prec_{lex} a_2 \prec_{lex} a_1$ and $x_5 \prec_{lex} x_4 \prec_{lex} x_3 \prec_{lex} x_2 \prec_{lex} x_1$ according to (N, W):

$$\begin{cases} ([\,], [a_1 - 1, a_2 - 1, a_3, -a_2a_5], & [false, [a_1 - 2, a_5, a_3], -a_2a_5x_4 + \\ & (-a_1^2 + 2a_1 - a_2 + 3)x_5 + a_4x_4]), \\ ([-a_2a_5], [a_3, a_1 - 1, a_2 - 1, a_1^2 - 2a_1 + a_2 - 3], & [false, [a_1 - 2, a_5, a_3], -a_2a_5x_4 + \\ & (-a_1^2 + 2a_1 - a_2 + 3)x_5 + a_4x_4]), \\ ([-a_2a_5, -a_1^2 + 2a_1 - a_2 + 3], [a_3, a_1 - 1, a_2 - 1], & [false, [a_1 - 2, a_5, a_3], -a_2a_5x_4 + a_4x_4]), \\ ([-a_2a_5, -a_1^2 + 2a_1 - a_2 + 3, a_4x_4], [\,], & [true, [a_1 - 2, a_5, a_3], -a_2a_5x_4]). \end{cases}$$

We can now apply a modified version of the GES algorithm using the LDS algorithm to compute a Gaussian elimination system for a set of parametric polynomials.

In the GES algorithm, Sys represents a global variable initially as an empty set, which ultimately becomes a Gaussian elimination system. Each entry in Sys is a triple (N, W, G), where (N, W) is a parametric constraint and G is the Gaussian elimination form of the input parametric polynomials corresponding to (N, W). Additionally, if B is a list or set, its i-th element is denoted as $B[i]$.

Algorithm 5 GES Algorithm.

Input: F as a subset of $\mathbb{K}[\mathbf{a}, \mathbf{A}]$, where **a** denotes the parameters a_i, **A** denotes the variables A_i, and (N, W) is a pair of parametric conditions.
Output: Sys, a Gaussian-elimination system for F.

- **Initialization:** Sys $:= \emptyset$
- **Initial queue:** $B := (N, W, , T[1], T)$
- **Main loop:** While $B \neq \emptyset$, do
  1. Let $b := B[1]$ and set $B := B \setminus b$
  2. If $b[5] = \{\}$ then
     $$\text{Sys} := \text{Sys} \cup (b[1], b[2], b[3]).$$
  3. Else
     $$G := b[5] \setminus b[4], \quad g := G[1].$$
     $$P := \text{LDS}(b[1], b[2], b[3], b[4]).$$
  4. For $i = 1, \ldots, |P|$, do
     - Let $P[i] = (N_1, W_1, [\text{flag}, Q, f])$
     - If flag $=$ true then
       $$B := B \cup (N_1, W_1, b[3], g, G),$$
       else
       $$B := B \cup (N_1, W_1, b[3] \cup f, g, G)$$
- **Return:** Sys

Theorem 2. The GES algorithm eventually terminates after a finite number of iterations and computes a linear Gröbner basis.

Proof. GES algorithm termination and correction are addressed in [6, Theorem 16]. In more detail, the algorithm terminates because F and each linear dependency system associated with $f \in F$ are finite. Correctness follows from the correctness of the LDS algorithm. Specifically, for each new polynomial $f \in F$, the LDS algorithm determines its linear dependency on the current basis. If dependent, f is discarded; otherwise, its normal form w.r.t. the basis is added. Consequently, each branch yields a Gaussian elimination form (or linear Gröbner basis) of the input matrix (or linear system) under the corresponding conditions. □

Dehghani Darmian and Hashemi illustrated the algorithm's behavior in detail [6, Example 15]; its output is presented:

Example 5. Consider the polynomial set $F = \{(a_1-a_2)x_1x_2+x_3-1, a_1x_2+x_1^2-2, (a_1-1)x_2^3-x_1^2-a_2x_3\} \subset \mathbb{K}[a_1,a_2][x_1,x_2,x_3]$. Applying the `Maple` implementation of GES to F yields the Gaussian elimination system below.

$$\begin{cases} ([\,], [a_1-a_2, 1-a_1], & [(a_1-a_2)x_1x_2+x_3-1, a_1x_2+x_1^2-2, a_1x_2^3-x_2^3+a_1x_2-a_2x_3-2]), \\ ([1-a_1], [1-a_2], & [-a_2x_1x_2+x_1x_2+x_3-1, x_1^2+x_2-2, -a_2x_3+x_2-2]), \\ ([a_2-a_1], [1-a_2], & [x_3-1, a_2x_2+x_1^2-2, a_2x_2^3-x_2^3+a_2x_2-a_2-2]), \\ ([a_2-a_1, 1-a_2], [\,], & [x_3-1, x_1^2+x_2-2, x_2-3]). \end{cases}$$

The GES algorithm efficiently computes Gröbner systems for parametric linear (homogeneous) polynomial ideals. We implemented both GES and PGBMain [12] in `Maple` and compared their performance using the examples from [6, Page 48]. The following table summarizes the results, showing CPU time (seconds) and memory usage (MB). The monomial orderings used were $c_3 \prec_{lex} c_2 \prec_{lex} c_1 \prec_{lex} \cdots \prec_{lex} a_2 \prec_{lex} a_1$ and $u \prec z \prec y \prec x$.

- Ex. 1: $[(3-a_1)x+(2-a_1a_2)y+b_1a_3z, (b_1-1)x+a_2b_2y+a_1b_3z]$
- Ex. 2: $[a_1x+a_1a_2y+a_1a_2a_3z, b_1x+b_1b_2y+b_1b_2b_3z, c_1x+c_1c_2y+c_1c_2c_3z]$
- Ex. 3: $[a_1x+(a_1-a_2)y+(a_2-a_3)z, b_1x+(b_1-b_2)y+(b_2-b_3)z, c_1x+(c_1-c_2)y+(c_2-c_3)z]$
- Ex. 4: $[a_1a_2x+(a_1a_2-a_3)z, b_1b_2x+(b_1b_2-b_3)y, c_1c_2y+(c_1c_2-c_3)z]$
- Ex. 5: $[(3-a_1a_2a_3)z+a_1u, (2-b_2)^2x+b_1b_2b_3u, (1-c_1c_3)x+(c_1-c_2)y+(c_2c_3-c_3)u, c_1x-c_2u]$

Installing the GES algorithm on the CSS algorithm instead of using the Gröbner system computations, we enhance the CSS algorithm, resulting in the Improved-CSS algorithm.

Theorem 3. The Improved-CSS algorithm terminates after finite iterations and effectively computes a complete solution system.

Proof. The termination of the Improved-CSS algorithm is guaranteed by the termination of the GES algorithm and its correctness is the same as the correctness of the CSS algorithm. □

Table 1: Performance comparison of the GES and PGBMain algorithms.

Example	Algorithm	Time (sec.)	Used Memory (MB)
Ex. 1	GES	0.37	19.47
	PGBMain	1.58	112.8
Ex. 2	GES	0.34	24.99
	PGBMain	1.49	107.1
Ex. 3	GES	1.04	60.37
	PGBMain	4.95	302.6
Ex. 4	GES	0.44	21.31
	PGBMain	1.48	111.54
Ex. 5	GES	0.78	39.73
	PGBMain	3.92	250.69

5 Comparison and Conclusion

In this section, we compare the performance of the Improved-CSS and CSS algorithms, implemented in `Maple`. We tested both algorithms on parametric first-order ODE systems, where parameters belong to the polynomial ring $\mathbb{K}[a, b, c, d, m, n]$, and aimed to compute comprehensive solution systems for each with respect to the parameter ordering $n \prec_{lex} m \prec_{lex} d \prec_{lex} c \prec_{lex} b \prec_{lex} a$.

- EX.1 = $[\dot{x}(t) = (a-1)x(t) + by(t) + d, \dot{y}(t) = (b-c)x(t) + ay(t) - 1]$
- EX.2 = $[\dot{x}(t) = (ab-1)x(t) + (n^3-1)y(t) + z(t), \dot{y}(t) = (cd-mn)x(t) + y(t) - z(t), \dot{z}(t) = x(t) + (-cd+1)z(t) - ny(t)]$
- EX.3 = $[\dot{x}(t) = (a^2-1)x(t) + y(t) + (b-1)z(t), \dot{y}(t) = (b-c)x(t) + y(t) - az(t), \dot{z}(t) = (1-c)x(t) - y(t) - z(t)]$
- EX.4 = $[\dot{x}(t) = (a-b+1)x(t) + c + u(t), \dot{y}(t) = cx(t) + y(t) - a^2z(t), \dot{z}(t) = (1-d)x(t) - y(t) - bu(t), \dot{u}(t) = 2x(t) - (b-d)u(t)]$
- EX.5 = $[\dot{x}(t) = (-d^4+1)u(t) + x(t) + 1, \dot{y}(t) = c^2 - a^2 + u(t) - y(t), \dot{z}(t) = -dv(t) - u(t), \dot{u}(t) = u(t) - ay(t) - z(t), \dot{v}(t) = x(t) - y(t) + (b-1)v(t)]$
- EX.6 = $[\dot{x}(t) = (1+c)v(t)+x(t), \dot{y}(t) = (c-a)u(t)-y(t), \dot{z}(t) = -dv(t)-u(t)-by(t), \dot{u}(t) = bu(t) - y(t) - z(t), \dot{v}(t) = -y(t) + (b-1)z(t)]$
- EX.7 = $[\dot{x}(t) = z(t)+(mn-m)y(t)+(-a^2+bc)x(t), \dot{y}(t) = (-m^3+1)z(t)+y(t)-x(t), \dot{z}(t) = x(t) + (-d^2+1)y(t) - z(t)]$
- EX.8 = $[\dot{u}(t) = x(t) + (1-m)y(t) - u(t), \dot{x}(t) = (b-1)y(t) - 2u(t) - x(t), \dot{y}(t) = (a-b)u(t) + y(t) - z(t), \dot{z}(t) = (a-2)x(t) - z(t) + y(t)]$
- EX.9 = $[\dot{x}(t) = (-m^3+mn)x(t) + (-a^2+bc)y(t), \dot{y}(t) = (a^3-m^3)x(t) + (n^3-d^2)y(t)]$

Algorithm 6 Improved-CSS

Input: System of parametric first-order ODEs

$$\text{sys} = \left\{ \frac{dx_j}{dt} = a_{j1}x_1 + \ldots + a_{jn}x_n + g_j(t) \right\}_{j=1}^{n}.$$

Output: A comprehensive solution system of sys.

Procedure:

- Define
$$F := \left\{ \frac{dx_j}{dt} - (a_{j1}x_1 + \ldots + a_{jn}x_n + g_j(t)) \right\}_{j=1}^{n}.$$
- Compute a linearization of F:
$$L \leftarrow \psi(F),$$
where ψ denotes the linearization operator (and $\psi(F)$ is its linearization).
- Obtain a Gaussian elimination system:
$$G \leftarrow \{(N_i, W_i, G_i)\}_{i=1}^{\ell},$$
which is a Gaussian elimination system of L using the GES algorithm.
- Initialize an empty list:
$$\text{LIST} \leftarrow \emptyset.$$
- **For** $i = 1, \ldots, \ell$ **do**:
  - Define
$$F_i' := \left\{ \psi^{-1}(G_i[j]) = 0 \,\middle|, j = 1, \ldots, n \right\}.$$
  - Compute the solution set
$$S_i \leftarrow \text{the solution set of } F_i' \text{ using any well-known classical methods.}$$
  - Update the list:
$$\text{LIST} \leftarrow \text{LIST} \cup \{(N_i, W_i, S_i)\}.$$

Return: LIST.

The following results were obtained from a Ryzen 6800 PC (8GB RAM, 64-bit Windows 10): Columns 3 and 4 show CPU time (s) and memory (GB). The last column indicates the number of solution branches. "—" signifies that the CSS algorithm failed to compute a comprehensive solution system within 300 seconds.

Table 2: Performance comparison of the improved-CSS and CSS algorithms.

Example	Algorithm	Time (Sec)	Used Memory (GB)	Branch
	Improved-CSS	0.31	0.046	7
EX.1	CSS	0.51	0.069	9
	Improved-CSS	74.18	12.76	16
EX.2	CSS	120.18	30.36	16
	Improved-CSS	21.52	3.51	13
EX.3	CSS	—	—	—
	Improved-CSS	9.51	1.93	18
EX.4	CSS	—	—	—
	Improved-CSS	6.83	1.84	2
EX.5	CSS	—	—	—
	Improved-CSS	5.47	0.98	8
EX.6	CSS	279.82	41.35	9
	Improved-CSS	34.79	5.14	12
EX.7	CSS	41.67	6.83	7
	Improved-CSS	25.03	4.87	6
EX.8	CSS	167.92	38.67	8
	Improved-CSS	0.48	0.52	9
EX.9	CSS	2.65	0.91	10

A comparison of the timing columns and our tests across various examples highlights the efficiency of our implemented Improved-CSS algorithm. Benchmark examples results demonstrate that the Improved-CSS algorithm, which employs parametric linear algebra techniques, significantly outperforms the CSS algorithm that relies on Gröbner system computations. Nevertheless, Gröbner systems remain effective tools for analyzing problems in mathematics, science, and engineering, such as solving systems of first-order ordinary differential equations with parametric coefficients. They facilitate the transformation of complex multivariable polynomial problems into simpler forms, leading to clearer solutions. This technique is applied to solving the system of parametric first-order ODEs, where partitioning the parameter space can ensure and enhance computational efficiency. Eventually, computational complexity increases significantly with the number and degree of parameters and larger system dimensions. While Gröbner bases computations and consistency checks contribute most to the complexity of the CSS algorithm, consistency checks remain the bottleneck in the improved-CSS algorithm. Increasing parameter length and the number/degree of parameters introduces practical problems for both CSS and improved-CSS algorithms.

6 Conclusion

This paper presented a novel algorithmic framework for solving systems of linear first-order ordinary differential equations (ODEs) with parametric coefficients. By leveraging Gröbner systems and parametric linear algebra techniques, we introduced a *comprehensive solution system* that partitions the parameter space into computationally tractable cells, each associated with a closed-form solution. The Improved-CSS algorithm, based on the GES method, significantly outperformed traditional Gröbner-based approaches in efficiency, as demonstrated by benchmark examples in `Maple`. Our work may offer insights into challenges in control theory and optimization, where parametric ODEs could model systems with uncertain or variable parameters (e.g., gain scheduling, robust stability analysis). Potential advantages might include optimization under uncertainty, where the cell-partitioning strategy could enable systematic analysis of parameter-dependent solutions, potentially supporting the design of controllers or estimators that may remain stable across parameter ranges. Additionally, the method appears to improve computational efficiency by avoiding Gröbner systems in favor of the GES algorithm, possibly reducing the complexity of solving parametric ODEs and potentially making it viable for real-time applications such as adaptive control and sensitivity analysis. While the Improved-CSS algorithm shows promise for linear systems, extensions to nonlinear parametric ODEs or hybrid dynamical systems (e.g., switched control systems) may present significant challenges. Future work could explore coupling the method with optimal control techniques such as Pontryagin's principle to potentially solve boundary-value problems, as well as applications to data-driven optimization, where parameter cells might inform machine learning models of dynamical systems. It is possible that this work could inspire further collaborations between computer algebra and control communities, particularly in scenarios requiring rigorous handling of parametric dependencies.

Declarations

Availability of Supporting Data
All data generated or analyzed during this study are included in this published paper.

Funding
The author conducted this research without any funding, grants, or support.

Competing Interests
The author declares that there are no competing interests relevant to the content of this paper.

References

[1] Åström, K.J., Murray, R.M. (2008). "Feedback systems: An introduction for scientists and engineers". *Princeton University Press*, doi:https://doi.org/10.1515/9781400828739.

[2] Cox, D., Little, J.B., O'Shea, D.B. (2007). "Ideals, varieties, and algorithms. An introduction to computational algebraic geometry and commutative algebra". *3rd ed. New York, NY: Springer*, doi:http://dx.doi.org/10.1007/978-3-319-16721-3.

[3] Dehghani Darmian, M. (2024). "Improvement of an incremental signature-based comprehensive Gröbner system algorithm. *Mathematics in Computer Science*, 18(12), doi:http://dx.doi.org/10.1007/s11786-024-00587-w.

[4] Dehghani Darmian, M., Hashemi, A. (2024). "A Parametric F_4 Algorithm". *Iranian Journal of Mathematical Sciences and Informatics*, 19(1), 117-133, doi:http://dx.doi.org/10.61186/ijmsi.19.1.117.

[5] Dehghani Darmian, M. (2024). "Efficient algorithm for computing inverse of parametric matrices." *Scientific Annals of Computer Science*, 34(1), 1-22, doi:http://dx.doi.org/10.47743/SACS.2024.1.1.

[6] Dehghani Darmian, M., Hashemi, A. (2017). "Parametric FGLM algorithm." *Journal of Symbolic Computation*, 82, 38-56, doi:https://doi.org/10.1016/j.jsc.2016.12.006.

[7] Dehghani Darmian, M., Hashemi, A., Montes, A. (2011). Erratum to "A new algorithm for discussing Gröbner bases with parameters" [J. Symbolic Comput. 33 (1-2) (2002) 183-208]. *Journal of Symbolic Computation*, 46(10), 1187-1188, doi:https://doi.org/10.1016/j.jsc.2011.05.002.

[8] Hashemi, A., Dehghani Darmian, M., Barkhordar, M. (2017). "Gröbner systems conversion." *Mathematics in Computer Science*, 11(1), 61-77, doi:https://doi.org/10.1007/s11786-017-0295-3.

[9] Hashemi, A., Dehghani Darmian, M., Barkhordar, M. (2018). "Universal Gröbner basis for parametric polynomial ideals." *Mathematical software - ICMS 2018. 6th International Conference, South Bend, IN, USA, 2018. Proceedings*, pages 191-199. Cham: Springer, doi:http://dx.doi.org/10.1007/978-3-319-96418-8_23.

[10] Hashemi, A., M.-Alizadeh, B., Dehghani Darmian, M. (2013). "Minimal polynomial systems for parametric matrices." *Linear and Multilinear Algebra*, 61(2), 265-272, doi:https://doi.org/10.1080/03081087.2012.670235.

[11] Kapur, D., Sun, Y., Wang, D. (2011). "Computing comprehensive Gröbner systems and comprehensive Gröbner bases simultaneously." *ISSAC '11: Proceedings of the 36th International Symposium on Symbolic and Algebraic Computation*, 193-200, doi:http://dx.doi.org/10.1145/1993886.1993918.

[12] Kapur, D., Sun, Y., Wang, D. (2010). "A new algorithm for computing comprehensive Gröbner systems." *Proceedings of the 35th International Symposium on Symbolic and Algebraic Computation, ISSAC 2010, Munich, Germany*, pages 29-36. New York, NY: Association for Computing Machinery (ACM), doi:https://doi.org/10.1145/1837934.1837946.

[13] Khalil, H.K. (2002). "Nonlinear systems". *3rd Edition, Prentice Hall, Upper Saddle River*.

[14] Logan, J.D. (2015). "A first course in differential equations." *Undergraduate Texts Mathematics, Cham: Springer, 3rd edition,* doi:https://doi.org/10.1007/978-3-319-17852-3.

[15] Manubens, M., Montes, A. (2009). "Minimal canonical comprehensive Gröbner systems." *Journal of Symbolic Computation*, 44(5), 463-478, doi:https://doi.org/10.1016/j.jsc.2007.07.022.

[16] Manubens, M., Montes, A. (2006). "Improving the DisPGB algorithm using the discriminant ideal." *Journal of Symbolic Computation*, 41(11), 1245-1263, doi:https://doi.org/10.1016/j.jsc.2005.09.013.

[17] Montes, A. (2002). "A new algorithm for discussing Gröbner bases with parameters." *Journal of Symbolic Computation*, 33(2), 183-208, doi:https://doi.org/10.1006/jsco.2001.0504.

[18] Montes, A., Castro, J. (1995). "Solving the load flow problem using the Gröbner basis." *ACM SIGSAM Bulletin*, 29(1), 1-13, doi:https://doi.org/10.1145/216685.216686.

[19] Ogata, K. (2010). "Modern control engineering." *Prentice-Hall Electrical Engineering Series. Instrumentation and Controls Series.*

[20] Rawlings, J.B., Mayne, D.Q., Diehl, M. (2017). "Model predictive control: Theory, computation, and design." *Nob Hill Publishing.*

[21] Ritt, J.F. (1932). "Differential equations from the algebraic standpoint". *Nature*, 131, page 456, doi:https://doi.org/10.1038/131456a0.

[22] Ritt, J.F. (1950). "Differential algebra, *Colloquium Publications, American Mathematical Society*. Volume 33, doi:https://doi.org/10.1090/coll/033.

[23] Skogestad, S., Postlethwaite, I. (2005). "Multivariable feedback control: Analysis and design." *Wiley-Interscience*.

[24] Suzuki, A., Sato, Y. (2006). "A simple algorithm to compute comprehensive Gröbner bases using Gröbner bases." *Proceedings of the 2006 International Symposium on Symbolic and Algebraic Computation*, ISSAC '06, Genova, Italy, pages 326-331. New York, NY: ACM Press, doi:http://dx.doi.org/10.1145/1145768.1145821.

[25] Weispfenning, V. (1992). "Comprehensive Gröbner bases." *Journal of Symbolic Computation*, 14(1), 1-29, doi:https://doi.org/10.1016/0747-7171(92)90023-W.

[26] Wu, W-T. (1978)."On the decision problem and the mechanization of theorem-proving in elementary geometry." *Scientia Sinica*, 21(2) , 117-138, doi:https://doi.org/10.1142/9789812791085_0008.

[27] Wu, W.T. (1986). "On zeros of algebraic equations - An application of Ritt principle." *Kexue Tongbao, Science Bulletin*, 31, 225-229, doi:https://doi.org/10.1142/9789812791085_0013.

[28] Zhou, K., Doyle, J.C., Glover, K. (1996). "Robust and optimal control". *Feher/Prentice Hall Digital and Prentice Hall*.