Scientific Annals of Computer Science vol. 34 (1), 2024, pp. 1–22
doi: 10.47743/SACS.2024.1.1

Efficient Algorithm for Computing Inverse of Parametric Matrices

Mahdi Dehghani Darmian[1,2]

Abstract

In this paper, we study and compute the inverse of matrices with parametric entries. We demonstrate that the Gauss-Jordan method can be extended to compute the inverse of parametric matrices, offering a powerful tool for solving systems of linear equations and analyzing parametric systems. Using this new expansion (so-called Gauss-Jordan systems) and also utilizing linearly dependency systems for linear systems involving parameters [4, 5], we introduce the notion of an inverse matrix system for a parametric matrix. In doing so, we decompose the space of parameters into a finite partition and for each partition, we give the corresponding inverse matrix without applying Gröbner systems. We also present an algorithm for computing an inverse system for a given parametric matrix. All mentioned algorithms have been implemented in `Maple`, and their efficiency and behavior have been experimented on a set of benchmark matrices.

Keywords: Parametric matrices, GES algorithm, IMS algorithm, Gauss-Jordan system, Inverse matrix system.

This research was in part supported by a grant from IPM (No.1402130044).

[1]Department of Mathematics, Technical and Vocational University (TVU), Tehran, Iran.

[2]School of Mathematics, Institute for Research in Fundamental Sciences (IPM), P.O. Box: 19395- 5746, Tehran, Iran.

Email: `m.dehghanidarmian@ipm.ir` & `m.dehghanidarmian@gmail.com`

1 Introduction

The inverse matrix computation is a fundamental operation in linear algebra with profound applications across various fields, including engineering, physics, computer science, and statistics. The inverse matrix is utilized as a powerful tool for solving systems of linear equations, providing insights into the underlying structures of mathematical models. Among the myriad techniques available for computing the inverse of a matrix, the Gauss-Jordan method stands out as a versatile approach that has found widespread use due to its ease of implementation.

Often, matrices with parametric entries, where elements of the matrix are expressed as parameters, arise in modeling real-world problems. This paper explores the application of the Gauss-Jordan method to compute the inverse of matrices with parametric entries. We demonstrate how the Gauss-Jordan method can be extended to handle matrices with parametric entries, empowering researchers and practitioners to solve complex problems in a variety of domains. The following example shows that the common approach for computing the inverse of matrices may not be used in parametric cases.

Example 1 *Let*

$$A = \begin{bmatrix} -b & 1 & a+1 \\ 0 & c & b+1 \\ -1 & 3+c & 1 \end{bmatrix}$$

where a, b and c are parameters in the complex numbers. The matrix inverse of A computed by the function `LinearAlgebra:-MatrixInverse(A)` *of* `Maple` *is equal to*

$$A^{-1} = \begin{bmatrix} -\frac{bc+3\,b+3}{b^2c+ac+3\,b^2+2\,b+c-1} & \frac{ac+3\,a+c+2}{b^2c+ac+3\,b^2+2\,b+c-1} & -\frac{ac-b+c-1}{b^2c+ac+3\,b^2+2\,b+c-1} \\ -\frac{b+1}{b^2c+ac+3\,b^2+2\,b+c-1} & \frac{a-b+1}{b^2c+ac+3\,b^2+2\,b+c-1} & \frac{(b+1)b}{b^2c+ac+3\,b^2+2\,b+c-1} \\ \frac{c}{b^2c+ac+3\,b^2+2\,b+c-1} & \frac{bc+3\,b-1}{b^2c+ac+3\,b^2+2\,b+c-1} & -\frac{bc}{b^2c+ac+3\,b^2+2\,b+c-1} \end{bmatrix}$$

Considering $\Delta = b^2c + ac + 3b^2 + 2b + c - 1$ as the determinant of A the matrix A^{-1} is defined when $\Delta \neq 0$. For instance if $a = 1$, $b = 0$, and $c = 0$ then A and A^{-1} are as follow:

$$A|_{\{a=1,b=0,c=0\}} = \begin{bmatrix} 0 & 1 & 2 \\ 0 & 0 & 1 \\ -1 & 3 & 1 \end{bmatrix}$$

$$A^{-1}|_{\{a=1,b=0,c=0\}} = \begin{bmatrix} 3 & -5 & -1 \\ 1 & -2 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

However, A^{-1} undefined for the values $a = 1, b = \frac{1}{3}$ and $c = 0$, since $\Delta = b^2c + ac + 3b^2 + 2b + c - 1 = 0$ and $A|_{\{a=1,b=\frac{1}{3},c=0\}}$ is the following Matrix with rank 2.

$$A|_{\{a=-2,b=-1,c=0\}} = \begin{bmatrix} -1/3 & 1 & 2 \\ 0 & 0 & 4/3 \\ -1 & 3 & 1 \end{bmatrix}.$$

In fact, one may be interested in identifying all possible parametric relations between a, b, and c where A is invertible or singular. With more details, we want to partition $\mathbb{V}(\Delta)$ and also $\mathbb{C}^3 \setminus \mathbb{V}(\Delta)$ into subvarieties. For this purpose, we introduce the *inverse matrix system* for parametric matrices and present an algorithm for computing it. In doing so, we can use two ideas; *Gröbner systems* and *Gauss-Jordan systems* based on *linearly dependency systems*.

2 Gröbner Systems

The concepts of Gröbner systems together with their algorithms were introduced by Weispfenning [13] in 1992. Gröbner systems can be considered as an extension of Gröbner bases to polynomials with parametric coefficients. Roughly speaking, for a parametric ideal, by computing its Gröbner system, we partition the space of parameters into a finite set of cells, and for each cell, we provide a set of polynomials. For any values of parameters, we find out the cell containing these values, and the corresponding polynomial set is a Gröbner basis of the ideal for the given values of parameters.

Throughout this article, we consider $\mathcal{R} = \mathbb{K}[x_1, \ldots, x_n]$ the polynomial ring in terms of $x_1, \ldots, x_n$ over a field $\mathbb{K}$. Let $\mathcal{I} = \langle f_1, \ldots, f_k \rangle \subset \mathcal{R}$ be the polynomial ideal generated by the f_i's. For any $f \in \mathcal{R}$, the *leading monomial* of f, denoted by $\mathrm{LM}_\prec(f)$, is the greatest monomial (w.r.t. the monomial ordering $\prec$) appearing in f and its coefficient is the *leading coefficient* of f which denoted by $\mathrm{LC}_\prec(f)$. The *leading term* of f w.r.t. $\prec$ is the product $\mathrm{LT}_\prec(f) = \mathrm{LC}_\prec(f)\mathrm{LM}_\prec(f)$. The *leading monomial ideal* of $\mathcal{I}$ is defined to be $\mathrm{LM}_\prec(\mathcal{I}) = \langle \mathrm{LM}_\prec(f) \mid f \in \mathcal{I} \rangle$. A finite subset $\{g_1, \ldots, g_m\} \subset \mathcal{I}$ is

called a *Gröbner basis* for $\mathcal{I}$ w.r.t. $\prec$ if $\mathrm{LM}_{\prec}(\mathcal{I}) = \langle \mathrm{LM}_{\prec}(g_1), \ldots, \mathrm{LM}_{\prec}(g_m) \rangle$. Gröbner bases together with the first algorithm to compute them were introduced by Buchberger in his Ph.D. thesis under the supervision of Gröbner [1]. For more details, we refer the interested readers to [2].

Using these notations, we recall the definition of Gröbner bases for ideals containing polynomial generators with parametric coefficients, namely Gröbner systems. Roughly speaking, Gröbner systems can be considered as an extension of Gröbner bases for polynomial ideals over fields to polynomial ideals with parametric coefficients. For this purpose, let us consider $\mathcal{S} = \mathbb{K}[\mathbf{a}, \mathbf{x}]$ as a polynomial ring with parametric coefficients where $\mathbf{x} = x_1, \ldots, x_n$ is a sequence of variables and $\mathbf{a} = a_1, \ldots, a_m$ is a sequence of parameters. Also, $\{\mathbf{x}\}$ and $\{\mathbf{a}\}$ are disjoint sets ($\{\mathbf{x}\} \cap \{\mathbf{a}\} = \emptyset$) which means that $x_i \neq a_j$ for any $i = 1, \ldots, n$ and $j = 1, \ldots, m$. Thus any monomial in $\mathcal{S}$ is in the form $\mathbf{x}^{\alpha} = x_1^{\alpha_1} \cdots x_n^{\alpha_n}$ where $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{N}^n$. Additionally, a term in $\mathcal{S}$ is $\mathbf{a}^{\gamma}\mathbf{x}^{\alpha}$ which $\mathbf{a}^{\gamma} \in \mathbb{K}[\mathbf{a}]$ is a monomial as a coefficient of $\mathbf{x}^{\alpha}$. Let us consider $\prec_{\mathbf{x}}$ and $\prec_{\mathbf{a}}$ as two monomial orderings on the variables and parameters, respectively. To define and compute Gröbner systems, we also need to recall a product order to specify an ordering on $\mathcal{S}$. The product ordering $\prec_{\mathbf{x},\mathbf{a}}$, the product of two monomial orderings $\prec_{\mathbf{x}}$ and $\prec_{\mathbf{a}}$, is defined as follows: For any $\gamma, \delta \in \mathbb{N}^m$ and $\alpha, \beta \in \mathbb{N}^n$, we can write $\mathbf{a}^{\gamma}\mathbf{x}^{\alpha} \prec_{\mathbf{x},\mathbf{a}} \mathbf{a}^{\delta}\mathbf{x}^{\beta}$ if either $\mathbf{x}^{\alpha} \prec_{\mathbf{x}} \mathbf{x}^{\beta}$ or $\{\mathbf{x}^{\alpha} = \mathbf{x}^{\beta} \text{and } \mathbf{a}^{\gamma} \prec_{\mathbf{a}} \mathbf{a}^{\delta}\}$. Furthermore, an specialization of parameters is a morphism $\sigma : \mathbb{K}[\mathbf{a}] \to \overline{\mathbb{K}}$ which in $\overline{\mathbb{K}}$ denotes the algebraic closure of $\mathbb{K}$. So, for any $m-$tuple $\mathbf{a} = t_1, \ldots, t_m$ and for each polynomial $f \in \mathcal{S}$ one may be write $\sigma(f) = f|_{\mathbf{a}=t_1,\ldots,t_m}$ where $\sigma(a_i) = t_i$. Moreover, a parametric polynomial ideal is a type of polynomial ideal defined by a system of parametric polynomial equations. This ideal is generated by polynomials with parameter coefficients. Additionally, a parametric linear polynomial refers to a polynomial where the power of the variables x_i's is at most one, and the parametric coefficients can be polynomials of any power in $\mathbb{K}[\mathbf{a}]$.

Now, let's review the definition of a Gröbner system for parametric polynomial ideals. The concept of Gröbner system was introduced by Weispfenning in [13]. He proved that any parametric polynomial ideal has a Gröbner system [13, Proposition 3.4 and Theorem 2.7] and described an algorithm to compute it [13, Theorem 3.6]. Since then, several different algorithms have been designed to compute the Gröbner system of an ideal, each of which has advantages and faults [4, 5, 3, 6, 7, 8, 9, 10, 11, 12].

Definition 1 *Let $F = \{f_1, \ldots, f_k\} \subset \mathcal{S}$ and $\mathcal{G} = \{(N_i, W_i, G_i)\}_{i=1}^{\ell}$ be a finite triple set where $N_i, W_i \subset \mathbb{K}[\mathbf{a}]$ and $G_i \subset \mathcal{S}$ are finite for $i = 1, \ldots, \ell$.*

The set $\mathcal{G}$ is called a Gröbner system for $\langle F\rangle$ w.r.t. to $\prec_{\mathbf{x},\mathbf{a}}$ over $\mathcal{V}\subseteq\overline{\mathbb{K}}^m$ if for any i we have

- *$\sigma(G_i)$ is a Gröbner basis of $\langle\sigma(F)\rangle$ with respect to $\prec_{\mathbf{x}}$, for any specialization $\sigma:\mathbb{K}[\mathbf{a}]\to\overline{\mathbb{K}}$ satisfying (N_i,W_i); (if $\sigma(p)=0$ for all $p\in N_i$ and $\sigma(q)\neq 0$ for all $q\in W_i$ then σ satisfies $(N_i,W_i)\subset\mathbb{K}[\mathbf{a}]\times\mathbb{K}[\mathbf{a}]$).*
- *$\mathcal{V}\subseteq\bigcup_{i=1}^{\ell}\mathbb{V}(N_i)\setminus\mathbb{V}(W_i)$ which $\mathbb{V}(N)$ is the common zeros of a set of polynomial generators for N.*

Any triple (N_i,W_i,G_i) is said a branch (segment) of the Gröbner system $\mathcal{G}$ for each $1\leq i\leq\ell$. Moreover, if $\mathcal{V}=\overline{\mathbb{K}}^m$ then $\mathcal{G}$ is generally called a Gröbner system of F. Also, each pair (N_i,W_i) is called a specification or a parametric constraint (N_i and W_i are called the null and non-null condition sets, respectively). Additionally, a parametric constraint (N,W) is called inconsistent if and only if for any polynomial $f\in W$, f belongs in the radical ideal generated by N. An effective technique for deciding the consistency of parametric constraints involves using an efficient combination of the ICheck and CCheck algorithms (introduced in [8]), along with the incorporation of Rabinovitch's trick for testing radical membership in the worst-case scenario; this is achieved by introducing a new variable. The interested reader can be referred to [8] for more details.

Example 2 *Let $F=\{(a-b)xy+(b-1)y,x^2+(2-b)y\}\subset\mathbb{K}[a,b,x,y]$ where x,y are variables and a,b are parameters. We consider the monomial orderings $z\prec_{drl}y\prec_{drl}x$ and $c\prec_{drl}b\prec_{drl}a$. Using our implementation of the* PF_4 *algorithm [5], we can compute a Gröbner system for $\langle F\rangle$ as follows*

$$\begin{cases}
([b-1,a-1],[\,], & [x^2+y])\\
([b-2,a-2],[\,], & [y,x^2])\\
([b-a],[b-2,b-1], & [by-y,x^2])\\
([b-2],[a-2], & [axy-2xy+y,x^2,y])\\
([\,],[a-b,b-2], & [axy-bxy+by-y,-by+x^2+2y,\\
 & (2b^2-a^2b+2ab^2-b^3+2a^2-4ab)y^2+(b^2-2b+1)y]).
\end{cases}$$

The above Gröbner system has five branches while assuming $a=3$ and $b=2$ randomly, the fourth branch corresponds to these values of parameters, and so $\{3xy-2xy+y,x^2,y\}$ is a Gröbner basis for the ideal $\langle F\rangle\,|_{a=3,b=2}$.

The question that may arise is: how one could use Gröbner system to decide the invertibility or singularity of a parametric matrix A? For

this purpose, we shall compute a Gröbner system $\{(N_i, W_i, G_i)\}_{i=1}^{\ell}$ for the corresponding parametric linear ideal $\langle \mathcal{I} \rangle$. Now, for each i if $|G_i| = |\mathcal{I}|$ then A is invertible provided that N_i and W_i are satisfied, and it is singular otherwise. We illustrate this technique with a simple example.

Example 3 *Consider the following matrix A with parametric entries from $\mathbb{K}[r, s, t]$.*

$$\begin{bmatrix} r-1 & 1 & t+1 \\ 2 & 2 & -s \\ r+1 & 3 & -1-s \end{bmatrix}$$

which corresponds to the following parametric linear ideal:

$$\mathcal{I} = \langle (r-1)x + y + (t+1)z, -sz + 2x + 2y, x(r+1) + 3y + (-1-s)z \rangle.$$

A Gröbner system of $\mathcal{I}$ w.r.t. the product ordering of $z \prec_{lex} y \prec_{lex} x$ and $t \prec_{lex} s \prec_{lex} r$ by using our implementation of PF_4 *algorithm [5] in* `Maple` *is as follows:*

$$\begin{cases} ([\,], & [2+t, r-2, r-1], & [(r-1)x + y + (t+1)z, \\ & & \frac{(2r-4)y-(rs-s+2t+2)z}{r-1}, (-2-t)z]), \\ ([r-1], & [2+t], & [2x + (-s-2t-2)z, y + (t+1)z, \\ & & (-2-t)z]), \\ ([2+t], & [r-2, r-1], & [(r-1)x + y - z, \frac{(2r-4)y-(rs-s-2)z}{r-1}]), \\ ([r-2], & [s+2t+2], & [x + y + (t+1)z, \frac{(-s-2t-2)z}{r-1}]), \\ ([s+2t+2, r-2], & [2+t], & [x + y + (t+1)z, (-2-t)z]), \\ ([2+t, r-1], & [\,], & [2x + (-s+2)z, y - z]]), \\ ([2+t, s-2, r-2], & [\,], & [x + y - z]). \end{cases}$$

This follows the following rank system of A (introduced in [4]):

$$\begin{cases} ([\,], & [2+t, r-2, r-1], & 3), \\ ([r-1], & [2+t], & 3), \\ ([2+t], & [r-2, r-1], & 2), \\ ([r-2], & [s+2t+2], & 2), \\ ([s+2t+2, r-2], & [2+t], & 2), \\ ([2+t, r-1], & [\,], & 2), \\ ([2+t, s-2, r-2], & [\,], & 1). \end{cases}$$

The rank of a matrix and its determinant are related in the sense that the determinant of a square matrix is non-zero if and only if the matrix has full

rank. In other words, if the determinant of a square matrix is non-zero, then the matrix is invertible and has full rank. Conversely, if the determinant is zero, then the matrix is singular and does not have full rank. According to this, we can conclude the following system for deciding the invertibility or singularity of A.

$$\begin{cases} ([\,], & [2+t, r-2, r-1], & \text{Invertible}), \\ ([r-1], & [2+t], & \text{Invertible}), \\ ([2+t], & [r-2, r-1], & \text{Singular}), \\ ([r-2], & [s+2t+2], & \text{Singular}), \\ ([s+2t+2, r-2], & [2+t], & \text{Singular}), \\ ([2+t, r-1], & [\,], & \text{Singular}), \\ ([2+t, s-2, r-2], & [\,], & \text{Singular}). \end{cases}$$

For example, if $s=-1, r=2$, and $t=-\frac{3}{2}$ the fourth branch corresponds to these values of parameters, and so $A|_{s=-1,r=2,t=-\frac{3}{2}}$ is singular.

3 Inverse Matrix Systems

The computation of the inverse matrix system via Gröbner system has two disadvantages: first, when the third component of any triple is "Invertible", the inverse matrix is not calculated and only the invertibility is reported. On the other hand, Suzuki and Sato [12] mentioned that the performance of the computation of Gröbner systems of parametric linear ideals "are generally slow and inefficient" (It is worth noting that a parametric linear ideal is generated by linear parametric polynomials. In the continuation of this paper, a linear parametric polynomial refers to a polynomial that is linear with respect to the main variables $\mathbf{x}$, and its parametric coefficients may be linear or non-linear). That is why, we apply the efficient algorithm GJS to compute the Gauss-Jordan system (without using Gröbner systems) for parametric matrices based on LDS and GES algorithms proposed in [4]. In this direction, we first restate the LDS algorithm to determine the dependency of a linear parametric polynomial on a given set of parametric polynomials without utilizing the Gröbner system and then recall the GES algorithm for computing a Gaussian elimination form for a parametric matrix. LDS and GES algorithms are crucial in the design of the GJS algorithm for computing the Gauss-Jordan form of a matrix with parametric entries. The LDS algorithm is essential for solving the problem of *parametric linearly dependency check*.

More precisely, let us consider $g \in \mathcal{S}$ as a parametric linear polynomial and a triple (N, W, G) such that $G \subset \mathcal{S}$ is a linear reduced Gröbner basis w.r.t. a given monomial ordering according to the constraint sets $N, W \subset \mathbb{K}[\mathbf{a}]$. The question that may arise is: how to specify the dependency of g on the triple (N, W, G)? To answer this question, we described the following LDS algorithm in [4]. Below, we use the NormalForm function which takes a polynomial p, a Gröbner basis $G = \{g_1, \ldots, g_m\}$, and a monomial ordering $\prec$ on the parameters or variables. It returns f and $\mathbb{Q} = [q_1, \ldots, q_m]$ such that $p = q_1 g_1 + \cdots + q_m g_m + f$, where f is the normal form of p by G.

Algorithm 1 LDS (Linearly Dependency System)

Require: $G \subset \mathcal{S}$; A linear reduced Gröbner basis w.r.t. the product of the monomial orderings $\prec_{\mathbf{x}}$ and $\prec_{\mathbf{a}}$ provided that a specification pair (N, W) is satisfied and $g \in \mathcal{S}$; a linear polynomial with parametric coefficients

Ensure: A linearly dependency system of g on (N, W, G)

Sys:= {}
f, Q :=NormalForm(g,GröbnerBasis($N, \prec_{\mathbf{a}}$), $\prec_{\mathbf{a}}$)
f', Q' :=NormalForm($f, G, \prec_{\mathbf{x}}$)
if $f' = 0$ **then**
  Sys:=Sys $\bigcup\{(N, W, [true, Q', 0])\}$
else
  $A := \{a_{i_1}, \ldots, a_{i_t}\}$ where $f' = a_{i_1} x_{i_1} + \cdots + a_{i_t} x_{i_t}$
  with $a_{i_j} \neq 0$ and $x_{i_1} \succ_{\mathbf{x}} \cdots \succ_{\mathbf{x}} x_{i_t}$
  for j **from** 1 **to** t **do**
    if a_{i_j} is not constant **then**
      Sys:=Sys
       $\bigcup\{(N \cup \{a_{i_1}, \ldots, a_{i_{j-1}}\}, W \cup \{a_{i_j}\}, [false, Q', f'|_{a_{i_1}=0,\ldots,a_{i_{j-1}}=0}])\}$
    else
      Sys:=Sys
       $\bigcup\{(N \cup \{a_{i_1}, \ldots, a_{i_{j-1}}\}, W, [false, Q', f'|_{a_{i_1}=0,\ldots,a_{i_{j-1}}=0}])\}$
      Return(Sys)
    end if
  end for
  Sys:=Sys $\bigcup\{(N \bigcup A, W, [true, Q', 0])\}$
end if
Return(Sys)

The linear dependency system of the following example is computed utilizing our `Maple` implementation of the above algorithm.

Example 4 *Consider* $(N, W) = ([m, n], [a - 1])$ *as a specification,* $G = [u+v, by-cz, bx-acz]$ *a linear Gröbner basis, and* $g = av-bw+mz+nx-u-y$ *a parametric polynomial. We fix the monomial orderings* $n \prec_{lex} m \prec_{lex} d \prec_{lex} c \prec_{lex} b \prec_{lex} a$ *and* $w \prec_{lex} v \prec_{lex} u \prec_{lex} z \prec_{lex} y \prec_{lex} x$ *on the parameters and the variables, respectively. So a linear dependency system of* g *on* G *according to* (N, W) *is as follow:*

$$\text{Sys} = \begin{cases} ([m,n],[c,a-1], & [false,[-1,\frac{-1}{b},0],-\frac{cz}{b}+(a+1)v-bw]), \\ ([m,n,-c],[a-1,a+1], & [false,[-1,\frac{-1}{b},0],-\frac{cz}{b}+(a+1)v-bw]), \\ ([m,n,-c,a+1],[b], & [false,[-1,\frac{-1}{b},0],-\frac{cz}{b}-bw]), \\ ([m,n,-c,a+1,b],[\,], & [true,[-1,\frac{-1}{b},0],0]). \end{cases}$$

The behavior of the above algorithm is illustrated step by step for a simple example in [4, Example 15]. Using this algorithm, we re-explain the Gaussian elimination for parametric matrices (so-called Gaussian elimination systems [4, Definition 10]) and also the efficient GES algorithm to compute these systems [4, Algorithm 3].

Definition 2 *Let* A *be a parametric matrix over* $\mathbb{K}[\mathbf{a}] = \mathbb{K}[a_1, \ldots, a_m]$. *A* Gaussian elimination system *of* A *is a finite set of triples* $\{(N_i, W_i, M_i)\}_{i=1}^{\ell}$ *such that for any specialization* $\sigma : \mathbb{K}[a] \to \bar{\mathbb{K}}$, *there exists an* i *satisfying the following conditions:*

- *The matrix* $\sigma(M_i)$ *is a Gaussian elimination form of* $\sigma(A)$,
- $\sigma(p) = 0$ *for each* $p \in N_i$ *and* $\sigma(q) \neq 0$ *for each* $q \in W_i$,
- $\bigcup_{i=1}^{\ell} \mathbb{V}(N_i) \setminus \mathbb{V}(\prod_{w \in W_i} w) = \overline{\mathbb{K}}^m$.

In the following, we restate the efficient GES algorithm for computing the Gaussian elimination systems [4, Algorithm 3].

Algorithm 2 GES (Gaussian Elimination System)

Require: M; A matrix with parametric entries over $\mathbb{K}[\mathbf{a}]$
Ensure: A Gaussian elimination system for M
  Sys:= {}
  $F :=$ the linear system in $\mathcal{S}$ corresponding to M
  $A := \{([\,], [\,], \{\}, F[1], F)\}$
  while $A \neq \{\}$ **do**
    $a := A[1]$ and $A := A \setminus \{a\}$
    if $a[5] = \{\}$ **then**
      Sys:=Sys $\bigcup\{(a[1], a[2], a[3])\}$

else
$G := a[5] \setminus \{a[4]\}$ and $g := G[1]$ and $P :=$LDS$(a[1], a[2], a[3], a[4])$
for i **from** 1 **to** $|P|$ **do**
$P[i] = (N, W, [flag, Q, f])$
if $flag = true$ **then**
$A := A \cup \{(N, W, a[3], g, G)\}$
else
$A := A \cup \{(N, W, a[3] \cup \{f\}, g, G)\}$
end if
end for
end if
end while
Return(Sys)

Example 5 *Let us consider the parametric matrix A in example 3. We get the following Gaussian elimination system of A using our* `Maple` *implementation of the above GES algorithm.*

$$\left\{\begin{array}{lll}
([\,], & [2+t, r-2, r-1], & \begin{bmatrix} r-1 & 1 & t+1 \\ 0 & \frac{2r-4}{r-1} & -\frac{rs-s+2t+2}{r-1} \\ 0 & 0 & -t-2 \end{bmatrix}), \\
([r-1], & [2+t], & \begin{bmatrix} 2 & 0 & -s-2t-2 \\ 0 & 1 & t+1 \\ 0 & 0 & -t-2 \end{bmatrix}), \\
([2+t], & [r-2, r-1], & \begin{bmatrix} r-1 & 1 & -1 \\ 0 & \frac{2r-4}{r-1} & -\frac{rs-s-2}{r-1} \\ 0 & 0 & 0 \end{bmatrix}), \\
([r-2], & [s+2t+2], & \begin{bmatrix} 1 & 1 & t+1 \\ 0 & 0 & \frac{-s-2t-2}{r-1} \\ 0 & 0 & 0 \end{bmatrix}), \\
([s+2t+2, r-2], & [2+t], & \begin{bmatrix} 1 & 1 & t+1 \\ 0 & 0 & -t-2 \\ 0 & 0 & 0 \end{bmatrix}), \\
([2+t, r-1], & [\,], & \begin{bmatrix} 2 & 0 & -s+2 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}), \\
([2+t, s-2, r-2], & [\,], & \begin{bmatrix} 1 & 1 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}).
\end{array}\right.$$

A matrix is said to be in reduced row echelon form (rref) when it is in row echelon form and vectors having one entry equal to 1 and all the other entries equal to 0. More precisely, a matrix is in reduced row echelon form if it satisfies the following conditions:

- The left-most nonzero entry in each row (known as the pivot or leading entry) is 1, and the column containing this pivot has all other entries equal to 0.
- The pivots in the next rows are to the right of the pivot in the row just above.
- Any zero-row (a row consisting entirely of zeros) is at the bottom.

The standard algorithm used to transform a matrix into a reduced row echelon form is called Gauss-Jordan elimination. However, the Gauss-Jordan elimination of a parametric matrix has not, to our knowledge, been studied in the literature. On the other hand, many problems in science and engineering can be modeled by parametric matrices, and have to be repeatedly solved for different values of parameters. So, the Gauss-Jordan of a parametric matrix may have a wide range of applications.

The following example shows that the traditional approach for computing the Gauss-Jordan form of matrices may not be used for such a matrix.

Example 6 *Let*

$$A = \begin{bmatrix} a-1 & 0 & c-2 & 1 \\ 2 & 0 & -1 & b-1 \\ a & b+c & 0 & -1 \end{bmatrix}$$

where a, b and c are parameters in the real numbers. The reduced row echelon form of A computed by the function `MTM:-rref(A)` *of* `Maple` *is equal to*

$$B = \begin{bmatrix} 1 & 0 & 0 & \frac{bc-2b-c+3}{a-5+2c} \\ 0 & 1 & 0 & \frac{abc-2ab-ac+4a+2c-5}{(a-5+2c)(b+c)} \\ 0 & 0 & 1 & -\frac{ab-a-b-1}{a-5+2c} \end{bmatrix}$$

which is a matrix of rank 3 for all values of parameters. However, this is undefined for the values $a = 3, b = 2$ and $c = 1$, where the Gauss-Jordan form of $A|_{\{a=3,b=2,c=1\}}$ is the following Matrix with rank 2.

$$\begin{bmatrix} 1 & 0 & -\frac{1}{2} & \frac{1}{2} \\ 0 & 1 & \frac{1}{2} & -\frac{5}{6} \\ 0 & 0 & 0 & 0 \end{bmatrix}$$

In fact, one may be interested in finding for what values of parameters, the Gauss-Jordan form of the above matrix is of rank 2 and for what values it is of rank 3. In doing so, we introduce the notion of the Gauss-Jordan system for a parametric matrix, and we present an algorithm for computing it.

Definition 3 *Let A be a parametric matrix over $\mathbb{K}[a_1, \ldots, a_m]$. A triples set $\{(N_i, W_i, M_i)\}_{i=1}^{\ell}$ is called a* Gauss-Jordan system *of A if*

- $\bigcup_{i=1}^{\ell} \mathbb{V}(N_i) \setminus \mathbb{V}(\prod_{w \in W_i} w) = \overline{\mathbb{K}}^m$
- *for each specialization $\sigma : \mathbb{K}[\mathbf{a}] \to \bar{\mathbb{K}}$, there is an i such that $\sigma(M_i)$ is a Gauss-Jordan form of $\sigma(A)$.*

Using the LDS and GES algorithms proposed in [4], we present an efficient algorithm to compute a Gauss-Jordan system for parametric matrices. Below, given an $n \times m$ parametric matrix M over $\mathbb{K}[\mathbf{a}]$, we associate a linear parametric polynomial system $F \subset \mathcal{S}$.

Algorithm 3 GJS (Gauss-Jordan System)

Require: M; a parametric matrix over $\mathbb{K}[\mathbf{a}]$
Ensure: A Gauss-Jordan system for M
  Compute a Gaussian Elimination System $\{(N_i, W_i, M_i)\}_{i=1}^{\ell}$
    for M using GES algorithm [4]
  Sys:= { }
  for i **from** 1 **to** ℓ **do**
    B_i := the reduced row echelon form of M_i
    Sys :=Sys $\bigcup$ $\{(N_i, W_i, B_i)\}$
  end for
  Return(Sys)

Theorem 4 *The* GJS *algorithm terminates and accurately computes a Gauss-Jordan system.*

Proof: The termination and correctness of the GES algorithm ensure the termination and correctness of the GJS algorithm. Specifically, for each triple (N_i, W_i, M_i), the first entry of any nonzero row from the left of M_i is nonzero according to specification (N_i, W_i). Thus, the Gauss-Jordan form can be completed for each triple. Of course, if a row has all zero entries, then it doesn't have a leading entry. Therefore, Sys is a Gauss-Jordan system for M at the end. □

Example 7 *Consider the matrix A mentioned in example 3. Using our* `Maple` *implementation of the GJS algorithm we obtain the following Gauss-Jordan system of A.*

$$\left\{\begin{array}{lll}
([\,], & [2+t, r-2, r-1], & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}), \\
([r-1], & [2+t], & \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}), \\
([2+t], & [r-2, r-1], & \begin{bmatrix} 1 & 0 & \frac{s-2}{2r-4} \\ 0 & 1 & \frac{rs-s-2}{4-2r} \\ 0 & 0 & 0 \end{bmatrix}), \\
([r-2], & [s+2t+2], & \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}), \\
([s+2t+2, r-2], & [2+t], & \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \end{bmatrix}), \\
([2+t, r-1], & [\,], & \begin{bmatrix} 1 & 0 & \frac{-s}{2}+1 \\ 0 & 1 & -1 \\ 0 & 0 & 0 \end{bmatrix}), \\
([2+t, s-2, r-2], & [\,], & \begin{bmatrix} 1 & 1 & -1 \\ 0 & 0 & 0 \\ 0 & 0 & 0 \end{bmatrix}).
\end{array}\right.$$

For example, the parameter values $r = 3, t = -2$, and $s = -4$ hold at the parametric constraints of the third branch of this system, and so the reduced row echelon form of $A|_{r=3,t=-2,s=-4}$ is as follow:

$$\text{rref}(\begin{bmatrix} 2 & 1 & -1 \\ 2 & 2 & 4 \\ 4 & 3 & 3 \end{bmatrix}) = \begin{bmatrix} 1 & 0 & -3 \\ 0 & 1 & 5 \\ 0 & 0 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 0 & \frac{s-2}{2r-4} \\ 0 & 1 & \frac{rs-s-2}{4-2r} \\ 0 & 0 & 0 \end{bmatrix}_{r=3,t=-2,s=-4}$$

In the following, first, we will briefly provide an overview of the Gauss-Jordan method for computing the inverse matrix with constant entries. We then introduce and compute an inverse matrix system utilizing the extension of the Gauss-Jordan method (the Gauss-Jordan systems) to handle matrices with parametric entries.

To find the inverse of a $n \times n$ matrix A: adjoin the identity matrix I to the right side of A, thereby producing a matrix of the form $[A|I]$ and then apply the row operations to this matrix until the left side is reduced to I. If successful, these operations will convert the right side to A^{-1}, so that the final matrix will have the form $[I|A^{-1}]$.

In many scenarios, the elements of a matrix are not fixed constants but instead depend on parameters or variables. These matrices with parametric entries play a pivotal role in modeling systems subject to change or variability. The ability to compute the inverse of such matrices is essential for solving systems of linear equations and gaining insights into the behavior of parametric systems. So, applying the Gauss-Jordan method to compute the inverse of matrices with parametric entries is an appropriate idea. In this direction, we introduce the concept of an inverse matrix system for parametric matrices. Then, we present the IMS algorithm, which extends the Gauss-Jordan method, to compute such systems.

Definition 5 *Let A be a square parametric matrix over $\mathbb{K}[\mathbf{a}] = \mathbb{K}[a_1, \ldots, a_m]$ and $\Delta = det(A)$. A finite triples set $\{(N_i, W_i, A_i)\}_{i=1}^{\ell} \bigcup \{(N'_j, W'_j, A'_j)\}_{j=\ell+1}^{\ell'}$ is an inverse matrix system of A if for each specialization $\sigma : \mathbb{K}[\mathbf{a}] \to \overline{\mathbb{K}}$ there is an $i \in \{1, \ldots, \ell\}$ or one $j \in \{\ell+1, \ldots, \ell'\}$ such that*

- *$\sigma(A_i)$ is inverse matrix of $\sigma(A)$ (or A'_j = Singular)*
- *$\sigma(p) = 0$; $\forall p \in N_i \subset \mathbb{K}[\mathbf{a}]$ (or $\forall p \in N'_j \subset \mathbb{K}[\mathbf{a}]$)*
- *$\sigma(q) \neq 0$; $\forall q \in W_i \subset \mathbb{K}[\mathbf{a}]$ (or $\forall q \in W'_j \subset \mathbb{K}[\mathbf{a}]$)*
- *$\bigcup_{i=1}^{\ell} \mathbb{V}(N_i) \setminus \mathbb{V}(\prod_{w \in W_i} w) = \overline{\mathbb{K}}^m \setminus \mathbb{V}(\Delta)$*

- $\bigcup_{j=\ell+1}^{\ell'} \mathbb{V}(N'_j) \setminus \mathbb{V}(\prod_{w\in W'_j} w) = \mathbb{V}(\Delta)$

For computing the inverse of a parametric matrix $A_{n\times n}$, we let Sys be a variable which is initialized to an empty set, and finally, it is the output inverse matrix system. Similar to matrices with constant arrays, at the beginning we can adjoin the identity matrix $I_{n\times n}$ to the right side of A, forming the matrix $[A|I]$ and then apply the GES algorithm to this matrix for computing a Gaussian elimination system of $[A|I]$. There are two kinds of triples in this step: those in which the rank of the produced matrices is less than n when the state is "Singular", and those whose matrices have rank n, which are the candidates of the "Invertible" state. The goal is to compute inverses, so the Gauss-Jordan systems are computed only when the rank equals n. Therefore, for the first kind, without any additional computation, the triple $(N_i, W_i,$ "Singular") is added to Variable Sys and the GJS algorithm idea only is applied to the matrices of the second kind of triples to compute a Gauss-Jordan system of $[A|I]$. At the end of this procedure, for the second triples kind, the right side of the produced matrices $n \times 2n$ is A^{-1}. Briefly, each recorded triple in Sys is of the form $(N, W,$ Singular or $A^{-1})$ where (N, W) is a conditions pair and A^{-1} is the inverse matrix of A according to (N, W) in non-singular cases.

Algorithm 4 IMS (Inverse Matrix System)

Require: $A_{n\times n}$; a parametric matrix over $\mathbb{K}[\mathbf{a}]$
Ensure: An inverse matrix system for A

```
M_{n×2n} := [A_{n×n}|I_{n×n}]
{(N_i, W_i, M_i)}_{i=1}^{ℓ} := A Gaussian Elimination System
        for M using GES algorithm
Sys:= { }
for i from 1 to ℓ do
  if rank(M_i) < n then
    Sys :=Sys ⋃ {(N_i, W_i, "Singular")}
  else
    B_i := the reduced row echelon form of M_i
    R_i := The right sub-matrix of B_i with order n
    Sys :=Sys ⋃ {(N_i, W_i, R_i)}
  end if
end for
Return(Sys)
```

Theorem 6 IMS *algorithm terminates after a finite number of steps and is correct.*

Proof: The termination of the algorithm is ensured by the finiteness of the GES algorithm. The correctness of the algorithm is guaranteed by the correctness of the LDS algorithm and the authenticity of the Gauss-Jordan method for the computation of inverse matrix with constant arrays. Indeed, by the structure of the algorithm, we explore applying the Gauss-Jordan method to compute the inverse of matrices with parametric entries. More precisely, it discusses parameter space using the LDS algorithm and for all consistent possible states decides the invertibility or singularity of matrices according to the parametric constraints. If the left sub-matrix M is equal to the identity matrix I then the triple (N_i, W_i, R_i) is added into the global variable Sys where R_i is the right sub-matrix M. Otherwise, the triple $(N_i, W_i,$ "Singular"$)$ is added into the global variable Sys which means that A is singular with respect to the corresponding conditions pair (N_i, W_i). □

We illustrate the behavior of this algorithm with a simple example.

Example 8 *Let us consider the parametric matrix from example 3 in the following.*

$$A = \begin{bmatrix} r-1 & 1 & t+1 \\ 2 & 2 & -s \\ r+1 & 3 & -1-s \end{bmatrix}$$

In the initial stage of the IMS *algorithm, a Gaussian elimination system of A is computed (see example 3.3). The rank of the corresponding matrices of the last five triples of this system is less than 3. As a result, without any additional computation, the triple* $\{(N_i, W_i,$ "Singular"$)\}_{i=1}^{5}$ *is added to Variable* Sys. *Then, the* GJS *algorithm idea is applied only to the first two triples for completing the Gauss-Jordan system for $[A|I]$. Finally, using our* `Maple` *implementation of the* IMS *algorithm an inverse matrix system of A is obtained as follows:*

Parametric Constraints	*Inverse Matrix System (*IMS*)*
$N_1 = [\,]$, $W_1 = [2+t, r-2, r-1]$	$\begin{bmatrix} -1/2\,\frac{s-2}{(t+2)(r-2)} & -1/2\,\frac{s+3\,t+4}{(t+2)(r-2)} & 1/2\,\frac{s+2\,t+2}{(t+2)(r-2)} \\ 1/2\,\frac{rs-s-2}{(t+2)(r-2)} & 1/2\,\frac{rs+rt+2\,r-s+t}{(t+2)(r-2)} & -1/2\,\frac{rs-s+2\,t+2}{(t+2)(r-2)} \\ (t+2)^{-1} & (t+2)^{-1} & -(t+2)^{-1} \end{bmatrix}$
$N_2 = [r-1]$, $W_2 = [2+t]$	$\begin{bmatrix} 1/2\,\frac{s-2}{t+2} & 1/2\,\frac{s+3\,t+4}{t+2} & -1/2\,\frac{s+2\,t+2}{t+2} \\ (t+2)^{-1} & -\frac{t+1}{t+2} & \frac{t+1}{t+2} \\ (t+2)^{-1} & (t+2)^{-1} & -(t+2)^{-1} \end{bmatrix}$
$N_3 = [2+t]$, $W_3 = [r-2, r-1]$	Singular
$N_4 = [r-2]$, $W_4 = [s+2t+2]$	Singular
$N_5 = [s+2t+2, r-2]$, $W_5 = [2+t]$	Singular
$N_6 = [2+t, r-1]$, $W_6 = [\,]$	Singular
$N_7 = [2+t, s-2, r-2]$, $W_7 = [\,]$	Singular

The computation of the inverse of parametric matrices (inverse matrix systems) can be used to solve systems of linear equations involving parametric coefficients. This opens up new possibilities for analyzing and predicting the behavior of systems with varying parameters. There are many practical examples from engineering, physics, and economics where matrices with parametric entries and their inverses play a critical role. These examples showcase the versatility and applicability of the Gauss-Jordan method in parametric modeling relevance in addressing real-world challenges.

4 Algorithm Performance

In this section, we provide the performance of computing the inverse of parametric matrices using the Gauss-Jordan method, including the used memory and the timing duration of the computations via some examples. For this purpose, we have implemented all the algorithms described in this paper in `Maple 18`. The following parametric matrices are chosen from the ring $\mathbb{K}[a, b, c, d, m, n]$, and our aim is to compute an inverse matrix system of each matrix.

The results are shown in the tables below, where the timings were executed on a personal computer with `Ryzen 6800, 8 GB RAM,` and `64 bits` under the `Windows 10` operating system. The third and fourth columns in these tables indicate the CPU time (in seconds) and the amount of required memory (in gigabytes) of the procedure. The last column shows the number of triples of the computed IMS algorithm. To illustrate the efficiency of our algorithm, we compare the performance of two methods, i.e. Gröbner systems and the IMS algorithm. However, this comparison is somewhat unfair because the IMS algorithm not only reports invertibility but also computes the inverse matrix in that case. In this direction, suppose that the following $F_1, \ldots, F_7$ are the parametric linear polynomial systems corresponding to the above matrices in the parametric polynomial ring $\mathbb{K}[a, b, c, d, m, n][x_1, x_2, x_3, x_4, x_5, x_6, x_7, x_8]$.

- $F_1 = [(a-1)x_1 + x_2 + (c+2)x_3, (b+1)x_2 - x_3, 3x_1 + (a-b)x_2 - 2x_3]$
- $F_2 = [(a-1)x_1 + x_2 + (a+1)x_3 - x_4, 2x_1 + (b+1)x_3 - x_4, x_1a + (c+b)x_2 + (a-b)x_3 - 2x_4, (a-1)x_1 + (b-c)x_2 + (a-c)x_3]$
- $F_3 = [(2-m)x_1 + nx_2 + (c-1)x_3 - x_4, -bx_4 - x_2 + x_3, mx_1 + cx_2 + (1-b)x_3 + 2x_4, -x_1 - cx_2 + (a-1)x_3]$
- $F_4 = [nx_2 + x_3 - x_5, bx_1 + (a-1)x_2 + x_4 + 2x_5, -2x_1 + (m-1)x_2 + x_3 + 2x_5, (n-1)x_2 + 3x_3 + bx_4, -x_2 + (1+d)x_3 + 2x_4]$
- $F_5 = [(m-1)x_1 + x_3 + 2x_5, nx_2 + x_3 + x_4 - x_6, -2x_1 + (a-1)x_3 - x_5, dx_4 - x_2 + x_5 - x_6, -x_3 + (b-1)x_5, cx_6 + 3x_1 + x_2 - x_5]$
- $F_6 = [x_2 + (m-1)x_3 + (m+1)x_5, (m^2-1)x_1 + 2x_3 - x_4 + x_6 + mx_7, x_2 + (n-1)x_3 + (n+1)x_4, -x_2 + (m+2)x_3 - x_6, -x_1 + (-m+1)x_2 - x_4 + x_5, -x_2 + 2x_3 + (m-2)x_7, (n-2)x_3 + x_4 + (m-n)x_5 + (m^2-4)x_6]$
- $F_7 = [(a-1)x_1 + x_3 + 2x_4 + (a+1)x_5 + (a^3+a)x_7, x_2 + (a-2)x_3 + (a+3)x_5 + x_7, (a^3-1)x_1 + 2x_3 - x_4 + x_5a + x_6 + x_8, x_2 + (a-1)x_3 + (1+3a)x_4 + (a^4-1)x_8, (a^4-a^2)x_1 - x_2 + 2x_3 + 2ax_4 - x_6, -x_1 + (a+1)x_2 - x_4 + x_5, -x_2 + 2x_3 + x_5a + (a-1)x_7, -2x_3 + x_4 + (a^3+1)x_5 + (a^2-1)x_6 + (a^5-a^2)x_7]$

Matrix	Time (Sec)	Used Memory (GB)	Branch
$\begin{bmatrix} a-1 & 1 & c+2 \\ 0 & b+1 & -1 \\ 3 & a-b & -2 \end{bmatrix}_{3\times 3}$	0.23	0.02	6
$\begin{bmatrix} a-1 & 1 & a+1 & -1 \\ 2 & 0 & b+1 & -1 \\ a & c+b & a-b & -2 \\ a-1 & b-c & a-c & 0 \end{bmatrix}_{4\times 4}$	0.98	0.14	15
$\begin{bmatrix} 2-m & n & c-1 & -1 \\ 0 & -1 & 1 & -b \\ m & c & 1-b & 2 \\ -1 & -c & a-1 & 0 \end{bmatrix}_{4\times 4}$	2.01	0.34	27
$\begin{bmatrix} 0 & n & 1 & 0 & -1 \\ b & a-1 & 0 & 1 & 2 \\ -2 & m-1 & 1 & 0 & 2 \\ 0 & n-1 & 3 & b & 0 \\ 0 & -1 & 1+d & 2 & 0 \end{bmatrix}_{5\times 5}$	4.67	0.79	45
$\begin{bmatrix} m-1 & 0 & 1 & 0 & 1 & 0 \\ 0 & n & 1 & 1 & 0 & -1 \\ -1 & 0 & a-1 & 0 & -1 & 0 \\ 0 & -1 & 0 & d & 1 & -1 \\ 0 & 0 & -1 & 0 & b-1 & 0 \\ 1 & 1 & 0 & 0 & -1 & c \end{bmatrix}_{6\times 6}$	6.78	1.02	66
$\begin{bmatrix} 0 & 1 & m-1 & 0 & m+1 & 0 & 0 \\ m^2-1 & 0 & 2 & -1 & 0 & 1 & m \\ 0 & 1 & n-1 & 1+n & 0 & 0 & 0 \\ 0 & -1 & m+2 & 0 & 0 & -1 & 0 \\ -1 & 1-m & 0 & -1 & 1 & 0 & 0 \\ 0 & -1 & 2 & 0 & 0 & 0 & m-2 \\ 0 & 0 & n-2 & 1 & m-n & m^2-4 & 0 \end{bmatrix}_{7\times 7}$	6.28	0.97	28
$\begin{bmatrix} a-1 & 0 & 1 & 2 & a+1 & 0 & a^3+a & 0 \\ 0 & 1 & a-2 & 0 & a+3 & 0 & 1 & 0 \\ a^3-1 & 0 & 2 & -1 & a & 1 & 0 & 1 \\ 0 & 1 & a-1 & 1+3a & 0 & 0 & 0 & a^4-1 \\ a^4-a^2 & -1 & 2 & 2a & 0 & -1 & 0 & 0 \\ -1 & a+1 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & -1 & 2 & 0 & a & 0 & a-1 & 0 \\ 0 & 0 & -2 & 1 & a^3+1 & a^2-1 & a^5-a^2 & 0 \end{bmatrix}_{8\times 8}$	3.12	0.45	8

The efficient PGBMain algorithm can now be used to compute Gröbner systems. However, a drawback of this algorithm is that the output Gröbner system may contain several branches, resulting in the corresponding Gröbner basis being $\{1\}$. By consolidating these branches into a single branch, the consistency check, as well as the used memory and time of computations, can be reduced, significantly improving the algorithm's performance. This simple modification was included in the PGBMain algorithm in [6] and we use it to compute Gröbner systems of the seven lists mentioned above.

Example	Time (sec.)	Used Memory (GB)
F_1	0.28	0.02
F_2	1.12	0.17
F_3	3.64	0.63
F_4	3.93	0.56
F_5	6.56	1.28
F_6	>300	—
F_7	29.65	5.26

Acknowledgment

This research was in part supported by a grant from IPM (No.1402130044). I sincerely appreciate all the valuable comments and suggestions of the reviewers, which helped me to improve the quality of the manuscript.

References

[1] Bruno Buchberger. Bruno Buchberger's PhD thesis 1965: An algorithm for finding the basis elements of the residue class ring of a zero dimensional polynomial ideal. *Journal of Symbolic Computation*, 41(3-4):475–511, 2006. doi:10.1016/J.JSC.2005.09.007.

[2] David A. Cox, John Little, and Donal O'Shea. *Ideals, Varieties, and Algorithms - An Introduction to Computational Algebraic Geometry and Commutative Algebra*. Undergraduate Texts in Mathematics. Springer, 4th edition, 2015. doi:10.1007/978-3-319-16721-3.

[3] Mahdi Dehghani Darmian, Amir Hashemi, and Antonio Montes. Erratum to "A new algorithm for discussing Gröbner bases with

parameters" Journal of Symbolic Computation 33(1-2) (2002) 183-208]. *Journal of Symbolic Computation*, 46(10):1187–1188, 2011. doi:10.1016/J.JSC.2011.05.002.

[4] Mahdi Dehghani Darmian and Amir Hashemi. Parametric FGLM algorithm. *Journal of Symbolic Computation*, 82:38–56, 2017. doi:10.1016/j.jsc.2016.12.006.

[5] Mahdi Dehghani Darmian and Amir Hashemi. Generalization of the F_4 algorithm to parametric polynomial ideals. In *7th International Conference on Combinatorics, Cryptography, Computer Science and Computation, I4C 2022*, pages 1014–1021. 2022. URL: https://i4c.iust.ac.ir/UPL/Paper2022/accpapers/i4c2022-1028.pdf.

[6] Amir Hashemi, Mahdi Dehghani Darmian, and Marzieh Barkhordar. Gröbner systems conversion. *Mathematics in Computer Science*, 11(1):61–77, 2017. doi:10.1007/S11786-017-0295-3.

[7] Amir Hashemi, Benyamin M.-Alizadeh, and Mahdi Dehghani Darmian. Computing comprehensive Gröbner systems: A comparison of two methods. *The Computer Science Journal of Moldova*, 25(3):278–302, 2017. URL: http://www.math.md/publications/csjm/issues/v25-n3/12511/.

[8] Deepak Kapur, Yao Sun, and Dingkang Wang. A new algorithm for computing comprehensive Gröbner systems. In Wolfram Koepf, editor, *23rd International Symposium on Symbolic and Algebraic Computation, ISSAC 2010*, pages 29–36. ACM, 2010. doi:10.1145/1837934.1837946.

[9] Antonio Montes. A new algorithm for discussing Gröbner bases with parameters. *Journal of Symbolic Computation*, 33(2):183–208, 2002. doi:10.1006/JSCO.2001.0504.

[10] Antonio Montes and Michael Wibmer. Gröbner bases for polynomial systems with parameters. *Journal of Symbolic Computation*, 45(12):1391–1425, 2010. doi:10.1016/J.JSC.2010.06.017.

[11] Katsusuke Nabeshima. A speed-up of the algorithm for computing comprehensive Gröbner systems. In Dongming Wang, editor, *20th Symbolic and Algebraic Computation, International Symposium, ISSAC 2007*, pages 299–306. ACM, 2007. doi:10.1145/1277548.1277589.

[12] Akira Suzuki and Yosuke Sato. A simple algorithm to compute comprehensive Gröbner bases using Gröbner bases. In Barry M. Trager, editor, *19th International Symposium on Symbolic and Algebraic Computation, ISSAC 2006*, pages 326–331. ACM, 2006. doi:10.1145/1145768.1145821.

[13] Volker Weispfenning. Comprehensive Gröbner bases. *Journal of Symbolic Computation*, 14(1):1–30, 1992. doi:10.1016/0747-7171(92)90023-W.