Fast Algorithms for Monomial Janet(-like) Bases

Amir Hashemi[a], Rodrigo Iglesias[c], Matthias Orth[b,d], Eduardo Sáenz-de-Cabezón[c], Werner M. Seiler[b]

[a] *Department of Mathematical Sciences, Isfahan University of Technology, Isfahan 84156-83111, Iran*
School of Mathematics, Institute for Research in Fundamental Sciences (IPM), Tehran, 19395-5746, Iran
[b] *Institut für Mathematik, Universität Kassel, Heinrich-Plett-Str. 40, 34132 Kassel, Germany*
[c] *Universidad de La Rioja, Logroño, Spain*
[d] *KU Leuven, Department of Mathematics, Celestijnenlaan 200B box 2400, BE-3001 Leuven*

Abstract

In this paper, we develop and compare three different approaches for computing Janet bases for monomial ideals. Each of these methods is straightforwardly extended to compute Janet-like bases as well. The first approach is a recursive method based on the structural properties of Janet bases, originally introduced by Janet in 1920. Additionally, we investigate the connection between Janet(-like) bases and staggered linear bases, which were introduced by Gebauer and Möller in 1986, leading to novel algorithms for the Janet(-like) completion process. As the third method, we present an iterative data structure approach for computing minimal Janet(-like) bases, utilizing the concept of Janet trees introduced by Gerdt et al. in 2001. Finally, we demonstrate that the minimal Janet basis is inherently encoded within the minimal Janet-like basis, enabling a more efficient computation of the minimal Janet basis. We analyze the arithmetic complexity of these methods and provide experimental benchmarks that illustrate their effectiveness in practical applications.

Keywords: Polynomial ideals, Gröbner bases, involutive bases, Janet bases, Janet-like bases, completion algorithms, staggered linear bases
2010 MSC: 13P10, 13D02, 68W30

1. Introduction

Gröbner bases are a fundamental concept in computational commutative algebra and algebraic geometry, and their efficient determination has been an important topic for a long time. *Involutive bases* are a special kind of Gröbner bases with additional combinatorial properties. The basic ideas underlying them stem from Janet's works on general systems of partial differential equations (Janet, 1920, 1929). The first rigorous

Email addresses: `Amir.Hashemi@iut.ac.ir` (Amir Hashemi), `Rodrigo.Iglesias@unirioja.es` (Rodrigo Iglesias), `morth@mathematik.uni-kassel.de` (Matthias Orth), `eduardo.saenz-de-cabezon@unirioja.es` (Eduardo Sáenz-de-Cabezón), `seiler@mathematik.uni-kassel.de` (Werner M. Seiler)

definition of involutive bases concerned what is nowadays called a Pommaret basis and was given by Zharkov and Blinkov (1996); the definition of arbitrary involutive bases and a general algorithm for their construction is due to Gerdt and Blinkov (1998). A more efficient algorithm allowing for the construction of minimal involutive bases was presented in (Gerdt and Blinkov, 1998). For a comprehensive study and for applications of the theory of involutive bases to commutative algebra and to partial differential equations, we refer to (Seiler, 2010).

Using ideas of Janet, Gerdt (2005) introduced an efficient algorithm for constructing involutive bases via a completion process in which the products of the current basis elements by non-multiplicative variables are reduced with respect to the basis. This process is guaranteed to terminate in finitely many steps for any division satisfying certain technical assumptions. Furthermore, Gerdt and his colleagues have demonstrated that this approach can also serve as an effective tool for the computation of Gröbner bases; see (Blinkov et al., 2003) for an efficient implementation and analysis of this approach. In particular, Gerdt et al. (2001) presented an effective approach to Janet completion that utilizes Janet trees as data structures. However, for a given polynomial ideal, the Janet basis may be much larger than the corresponding reduced Gröbner basis (toric ideals are a prototypical example). To address this issue, Gerdt and Blinkov (2005) introduced the notion of *Janet-like bases*; a generalization of Janet bases in which the completion process takes into account non-multiplicative powers instead of non-multiplicative variables in the completion process. As a result, any method for computing Janet bases can be extended into an algorithm for computing Janet-like bases. Furthermore, every Janet basis includes a Janet-like basis for the ideal it generates.

Another key aspect of this paper is the concept of staggered linear bases. This type of basis is a linear basis of a polynomial ideal which includes a Gröbner basis (Gebauer and Möller, 1986). The first correct approach for computing staggered linear bases using intermediate syzygies was given in (Möller et al., 1992). A simple and efficient algorithm to compute these bases was given recently in (Hashemi and Möller, 2023).

In this work, we develop, analyze, and compare three types of algorithms for computing Janet and Janet-like bases for monomial ideals. First, we employ recursive structures for Janet bases, as outlined in (Hashemi et al., 2023). Next, we propose another iterative algorithm based on the observation that Janet bases constitute a specialized class of staggered linear bases. Finally, inspired by the Janet tree structure introduced in (Gerdt et al., 2001) and its variant presented in (Hashemi et al., 2025), we develop a new iterative algorithm for computing Janet bases. We also discuss how these algorithms can be extended to construct Janet-like bases. We establish the arithmetic complexities of our algorithms and present experimental results comparing their performance. Additionally, we provide a comparison with the `CoCoALib` implementation of Janet bases by Albert et al. (2015).

The structure of the paper is as follows. In the next section, we give basic notations and definitions that are used throughout the paper. In Sections 3 and 4 we present improved versions of the recursive algorithms from (Hashemi et al., 2023) for the computation of monomial Janet and Janet-like bases, respectively. We continue in Section 5 with an analysis of the connection of the theories of staggered linear bases and Janet bases, leading to a novel iterative algorithm for the construction of Janet bases. We extend these results to Janet-like bases in Section 6. In Section 7, we introduce a variation of the existing iterative algorithms for computing minimal monomial Janet bases and extend this approach to the computation of monomial Janet-like bases. In Section 8, we investigate the relation between Janet-like and Janet bases, yielding a fast method for obtaining a minimal Janet basis from a minimal Janet-like basis, a procedure for which we coin the name *filling in the gaps*. We conclude in Section 9 by presenting benchmarks for the algorithms presented in this work.

2. Preliminaries

In this section, we review some basic definitions and notations from the theories of Gröbner bases and involutive bases that will be used in the rest of the article. Throughout, we work in the polynomial ring $\mathcal{P} = \mathcal{K}[X] = \mathcal{K}[x_1, \ldots, x_n]$ over a field $\mathcal{K}$. We consider the polynomials $f_1, \ldots, f_k \in \mathcal{P}$ and the ideal $\mathcal{I} = \langle f_1, \ldots, f_k \rangle$ generated by them. We denote the total degree of and the degree with respect to a variable x_i of a polynomial $f \in \mathcal{P}$ by $\deg(f)$ and $\deg_i(f)$, respectively. We write $\mathcal{T} = \{x_1^{\alpha_1} \cdots x_n^{\alpha_n} \mid \alpha_i \geq 0,\ 1 \leq i \leq n\}$ for the monoid of all terms in $\mathcal{P}$. A term ordering on $\mathcal{T}$ is denoted by $\prec$ and throughout we shall assume that $x_1 \prec \cdots \prec x_n$. The leading term of a given non-zero polynomial $f \in \mathcal{P}$ with respect to $\prec$ is denoted by $\operatorname{lt}(f)$. If $F \subset \mathcal{P}$ is a finite set of non-zero polynomials, we denote by $\operatorname{lt}(F)$ the set $\{\operatorname{lt}(f) \mid f \in F\}$. A finite set $G \subset \mathcal{I}$ is called a *Gröbner basis* for $\mathcal{I}$ with respect to $\prec$, if its leading ideal satisfies $\operatorname{lt}(\mathcal{I}) = \langle \operatorname{lt}(f) \mid f \in \mathcal{I} \rangle = \langle \operatorname{lt}(G) \rangle$. We refer e. g. to (Cox et al., 2015; Adams and Loustaunau, 1994; Mora, 2005, 2016) for more details on Gröbner bases.

Next, we recall some relevant concepts for involutive divisions and bases, see (Gerdt, 2005; Seiler, 2010) for more details.

Definition 2.1. An *involutive division* $\mathcal{L}$ on $\mathcal{T} \subset \mathcal{P}$ associates to any finite set $U \subset \mathcal{T}$ of terms and any term $u \in U$ a set of *$\mathcal{L}$-non-multipliers* $\bar{\mathcal{L}}(u, U)$ given by the terms contained in a prime monomial ideal. The variables generating this prime ideal are called the *non-multiplicative variables* $\mathrm{NM}_{\mathcal{L}}(u, U) \subseteq X$ of $u \in U$. The set of *$\mathcal{L}$-multipliers* $\mathcal{L}(u, U)$ is given by the order ideal $\mathcal{T} \setminus \bar{\mathcal{L}}(u, U)$; it has as Dickson basis the set of *multiplicative variables* $\mathrm{M}_{\mathcal{L}}(u, U) = X \setminus \mathrm{NM}_{\mathcal{L}}(u, U)$. For any term $u \in U$, its *involutive cone* is defined as $\mathcal{C}_{\mathcal{L}}(u, U) = u \cdot \mathcal{L}(u, U)$. For an involutive division, the involutive cones must satisfy the following conditions:

(i) For two terms $v \neq u \in U$ with $\mathcal{C}_{\mathcal{L}}(u, U) \cap \mathcal{C}_{\mathcal{L}}(v, U) \neq \emptyset$, we have $u \in \mathcal{C}_{\mathcal{L}}(v, U)$ or $v \in \mathcal{C}_{\mathcal{L}}(u, U)$.

(ii) If a term $v \in U$ lies in an involutive cone $\mathcal{C}_{\mathcal{L}}(u, U)$, then $\mathcal{L}(v, U) \subset \mathcal{L}(u, U)$.

(iii) For any term u in a subset $V \subset U$, we have $\mathcal{L}(u, U) \subseteq \mathcal{L}(u, V)$.

We write $u \mid_{\mathcal{L}} w$ for a term $u \in U$ and an arbitrary term $w \in \mathcal{T}$, if $w \in \mathcal{C}_{\mathcal{L}}(u, U)$. In this case, u is called an *$\mathcal{L}$-involutive divisor* of w and w an *$\mathcal{L}$-involutive multiple* of u.

The first two conditions ensure that involutive cones can intersect only trivially. The third condition is often called the *filter axiom*. Obviously, it suffices for defining an involutive division to say what are the (non-)multiplicative variables for each term u in a finite set U. Note that involutive divisibility $u \mid_{\mathcal{L}} w$ implies ordinary divisibility, but not vice versa.

Definition 2.2. For a finite set of terms $U \subset \mathcal{T}$ and an involutive division $\mathcal{L}$ on $\mathcal{T}$, the *involutive span* of U is the union $\mathcal{C}_{\mathcal{L}}(U) = \bigcup_{u \in U} \mathcal{C}_{\mathcal{L}}(u, U)$. The set U is *involutively complete* or a *weak involutive basis*, if $\mathcal{C}_{\mathcal{L}}(U) = U \cdot \mathcal{T}$. For a *(strong) involutive basis* the union is disjoint, i. e. every term in $\mathcal{C}_{\mathcal{L}}(U)$ has a unique involutive divisor. An involutive division is *Nœtherian*, if every monomial ideal in $\mathcal{P}$ has an involutive basis.

Example 2.3. One of the most important involutive divisions is the *Janet division* introduced by Janet (1929, pages 16-17). Let $U \subset \mathcal{P}$ be a finite set of terms. For each sequence $d_1, \ldots, d_n$ of non-negative integers and for each index $1 \leq i \leq n$, we introduce the corresponding *Janet class* as the subset

$$U_{[d_i,\ldots,d_n]} = \{u \in U \mid \deg_j(u) = d_j,\ i \leq j \leq n\} \subset U\ . \tag{2.1}$$

The variable x_n is *Janet multiplicative* (or shorter J-multiplicative) for the term $u \in U$, if $\deg_n(u) = \max\{\deg_n(v) \mid v \in U\}$. For $i < n$ the variable x_i is Janet multiplicative for $u \in U_{[d_{i+1},\ldots,d_n]}$, if $\deg_i(u) = \max\{\deg_i(v) \mid v \in U_{[d_{i+1},\ldots,d_n]}\}$. The Janet division is Nœtherian.

We now turn to the Janet-like division introduced in (Gerdt and Blinkov, 2005).

Definition 2.4. Let $U \subset \mathcal{T}$ be a finite set of terms. For any term $u \in U$ and any index $1 \leq i \leq n$, we set

$$h_i(u, U) = \max\{\deg_i(v) \mid u, v \in U_{[d_{i+1},\ldots,d_n]}\} - \deg_i(u)\ .$$

If $h_i(u, U) > 0$, the power $x_i^{k_i}$ with

$$k_i = \min\{\deg_i(v) - \deg_i(u) \mid v, u \in U_{[d_{i+1},\ldots,d_n]}, \deg_i(v) > \deg_i(u)\}$$

is called a *non-multiplicative* power of u for the *Janet-like division*. The set of all non-multiplicative powers of $u \in U$ is denoted by $\mathrm{NMP}(u, U)$. The elements of the set

$$\mathrm{NM}(u, U) = \{v \in \mathcal{T} \mid \exists w \in \mathrm{NMP}(u, U), w \mid v\}$$

are called the *JL-non-multipliers* for $u \in U$. The terms outside of it are the *JL-multipliers* for u. An element $u \in U$ will be called a *Janet-like divisor* of $w \in \mathcal{T}$, if $w = u \cdot v$ with v a JL-multiplier for u.

A finite set $U \subset \mathcal{T}$ is called *Janet-like basis* of the monomial ideal $\langle U \rangle$, if every term $t \in \langle U \rangle \cap \mathcal{T}$ has a Janet-like divisor in U. A finite set of polynomials $F \subset \mathcal{P} \setminus \{0\}$ is a *Janet-like basis* of $\mathcal{I} = \langle F \rangle$, if we have $\mathrm{lt}(f) \neq \mathrm{lt}(g)$ for all $f \neq g \in F$ and $\mathrm{lt}(F)$ forms a Janet-like basis for $\mathrm{lt}(\mathcal{I})$.

Although the Janet-like division is not an involutive division, it preserves all algorithmic properties of the Janet division and allows for the construction of Janet-like bases and in turn Gröbner bases. Indeed, the main algorithmic idea for the construction of Janet-like bases is similar to that of Janet bases; instead of multiplying by non-multiplicative variables one now multiplies by non-multiplicative powers. Moreover, Janet-like bases can also be represented by trees (Hashemi et al., 2022) and bar codes (Ceria, 2022). Since the rest of the paper focuses on checking and computing minimal Janet and Janet-like bases, we provide the definition below.

Definition 2.5. An $\mathcal{L}$-involutive (or a Janet-like) basis $U \subset \mathcal{P}$ is called *minimal*, if no proper subset of U is an $\mathcal{L}$-involutive (or a Janet-like) basis of the ideal $\langle U \rangle$.

3. A recursive Janet completion algorithm

In this section, based on the results from (Hashemi et al., 2023), we present effective recursive methods for testing whether a given set of terms forms a Janet basis, as well as for the Janet completion process. Janet (1920, page 86) introduced the following recursive criterion for determining whether a set of terms constitutes a Janet basis, which stems from an extensive discussion on the properties of the Janet division. For more details, we refer also to (Ceria, 2022, Theorem 3.11).

Theorem 3.1. *Let $U = \{t_1, \dots, t_m\} \subset \mathcal{T}$ be a finite set of terms. We define $t_i' = t_i|_{x_n=1}$ for all i and $U' = \{t_1', \dots, t_m'\} \subset \mathcal{K}[x_1, \dots, x_{n-1}]$. If $\alpha = \max\{\deg_n(t_1), \dots, \deg_n(t_m)\}$, then we introduce for each degree $\lambda \leq \alpha$ the sets $I_\lambda = \{i \mid \deg_n(t_i) = \lambda\}$ and $U_\lambda' = \{t_i' \mid i \in I_\lambda\}$. Then, U is a Janet basis if and only if the following two conditions are satisfied:*
- *(i) For each $\lambda \leq \alpha$ the set U_λ' is a Janet basis in $\mathcal{K}[x_1, \dots, x_{n-1}]$.*
- *(ii) Each term $t_i' \in U_\lambda'$ with $\lambda < \alpha$ lies in the Janet span of $U_{\lambda+1}'$.*

A slight modification in (Hashemi et al., 2023, Theorem 3.4) (see the next theorem) improves this result, enabling verification of ordinary term memberships instead of condition (*ii*).

Theorem 3.2. *In the situation of Theorem 3.1, let $\beta = \min\{\deg_n(t_1), \dots, \deg_n(t_m)\}$. Then, U is a Janet basis if and only if the following conditions are satisfied:*
- *(i) For each $\lambda \leq \alpha$, U_λ' is a Janet basis in $\mathcal{K}[x_1, \dots, x_{n-1}]$.*
- *(ii) For each $\beta \leq \lambda < \alpha$, we have $U_\lambda' \subset \langle U_{\lambda+1}' \rangle$.*

Furthermore, Hashemi et al. (2023, Theorem 3.10) introduced the following test for minimal Janet bases.

Theorem 3.3. *With the notations of Theorem 3.1, let U be a Janet basis for the ideal it generates. Then, U is minimal if and only if the following conditions are satisfied:*
- *(i) For each $\lambda \leq \alpha$, U_λ' is a minimal Janet basis.*
- *(ii) We have $\langle U_{\alpha-1}' \rangle \neq \langle U_\alpha' \rangle$.*

Based on these results, Hashemi et al. (2023, Algorithm 3) introduced an algorithm to minimize a Janet basis; however, no algorithm has been described for computing the minimal Janet basis of a monomial ideal. In this context, we present Algorithm 1; an adapted version of (Hashemi et al., 2023, Algorithm 3) that computes the minimal Janet basis for a given monomial ideal represented by its minimal generating set. We use $\mathrm{Gen}(A)$ to denote the unique minimal generating set consisting of terms for the monomial ideal A. Let U be a set of terms; note that $\mathrm{Gen}(\langle U \rangle) \subset U$. The process of removing irrelevant elements from U to obtain the minimal generating set of the ideal it generates is known as *minimization* or *inter-reduction*.

Theorem 3.4. *Algorithm 1 terminates in finitely many steps and returns the minimal Janet basis for the ideal generated by its input set. Its arithmetic complexity*[1]

[1] In this work, by arithmetic complexity we mean the total number of all involved elementary operations such as comparison, addition and multiplication over the base field. Moreover, we assume that the cost of a single operation is one.

Algorithm 1: RecMinimalJanetCompletion

Data: A polynomial ring $\mathcal{P} = \mathcal{K}[x_1, \ldots, x_n]$ and a finite and inter-reduced set $U \subset \mathcal{P}$ of terms

Result: The minimal Janet basis of $\langle U \rangle$

1 **begin**

2 $\quad U \longleftarrow \{t_1, \ldots, t_m\}$

3 $\quad$ **if** $n = 1$ ***or*** $m = 1$ **then**

4 $\quad\quad$ **return** U

5 $\quad \alpha \longleftarrow \max\{\deg_n(t_1), \ldots, \deg_n(t_m)\}$

6 $\quad \beta \longleftarrow \min\{\deg_n(t_1), \ldots, \deg_n(t_m)\}$

7 $\quad V \longleftarrow \emptyset$ **and** $V'_{\beta-1} \longleftarrow \emptyset$

8 $\quad$ **for** $i = \beta, \ldots, \alpha$ **do**

9 $\quad\quad V'_i \longleftarrow \{t \in \mathcal{K}[x_1, \ldots, x_{n-1}] \mid t \cdot x_n^i \in U\}$

10 $\quad\quad V'_i \longleftarrow \mathrm{Gen}\left(\langle V'_i \cup V'_{i-1} \rangle\right)$

11 $\quad\quad V'_i \longleftarrow$ `RecMinimalJanetCompletion`$(\mathcal{K}[x_1, \ldots, x_{n-1}], V'_i)$

12 $\quad\quad V \longleftarrow V \cup \{t x_n^i \mid t \in V'_i\}$

13 $\quad$ **return** (V)

is $O\left(m^2 n^2 d\right)$ where m denotes the size of the input set U, n is the number of variables, and d represents the maximum of the differences between the maximal and minimal degrees of the elements of U with respect to each variable.

Proof. The termination of the algorithm is straightforward because it calls itself n times and each call consists of a finite number of iterations. This guarantees the finite termination of the algorithm.

To prove the correctness, assume that V is the output of the algorithm and V'_i for each i is as defined in Theorem 3.1. From the structure of the algorithm (line 10), we conclude that $\langle V'_{i-1} \rangle \subset \langle V'_i \rangle$ for each i. Furthermore, by line 11, V'_i is a Janet basis for each i. Therefore, from Theorem 3.2, it follows that V is a Janet basis.

Now, to prove that V is minimal, we proceed by induction and apply Theorem 3.3. We claim that at each iteration (on the number of variables) the Janet basis constructed by the algorithm is minimal. We note that the input set U is inter-reduced. If $n = 1$ then the set U is the minimal Janet basis for the ideal it generates. Now, assume for simplicity that we are working with n variables, and the induction hypothesis holds true for $n - 1$ variables. We show that the set constructed at the end of the **for**-loop forms the minimal Janet basis for the ideal it generates. Here we follow the notations used in the algorithm. We must show that $A := V \cup \{t x_n^\alpha \mid t \in V'_\alpha\}$ constructed in line 12 is a minimal Janet basis. Note that the set V'_i for each i (as a set in $n - 1$ variables) constructed in line 11, is the minimal Janet basis for the ideal it generates. Now, it suffices to show that $\langle V'_{\alpha-1} \rangle \neq \langle V'_\alpha \rangle$ where these sets are constructed in line 11. Since U is inter-reduced, the set V'_α produced in line 9 is non-empty, and this proves the desired inequality. These arguments show that the set A is the minimal Janet basis for $\langle U \rangle$.

Finally, we establish the complexity bound. It is evident that sorting the sequence

$t_1, \ldots, t_m$ with respect to the lexicographical ordering given by $x_1 \prec \cdots \prec x_n$ requires $O(mn\log(m))$ operations. Furthermore, once the sequence is ordered from least to greatest, computing the minimal generating set of the ideal generated by this sequence necessitates $O(m^2n)$ field operations. Assume we are working in the ring $\mathcal{K}[x_1, \ldots, x_\ell]$. At this stage, define the set

$$A := \{t_i|_{x_{\ell+1}=\cdots=x_n=1} \mid i = 1, \ldots, m\}.$$

We then partition A into subsets $A_\beta, \ldots, A_\alpha$, where A_i consists of the terms t such that $\deg_\ell(t) = i$. For simplicity, we may assume that $\alpha - \beta = d$ and that each A_i contains m/d elements. For each i, we compute the minimal generating set of the ideal $\langle A_\beta, \ldots, A_i\rangle$. Given that U is already ordered, these sets are also ordered. Consequently, finding the minimal generating set for this ideal requires $O((im/d)^2n)$ operations. Since i ranges from 1 to d, the total number of operations needed in this step is $O(m^2nd)$. We invoke the algorithm for $\ell = 1, \ldots, n$, leading to an overall complexity of $O(m^2n^2d)$. □

Example 3.5. We illustrate the steps of Algorithm 1 for the inter-reduced set $U = \{x_2^2x_3,\ x_1^2x_3^3\} \subset \mathcal{K}[x_1, x_2, x_3]$. Keeping the notations used in the algorithm, at the first recursion level in $\mathcal{K}[x_1, x_2]$ one obtains

1. $V_1' = \{x_2^2\}$. Its minimal Janet basis is $\{x_2^2\}$ and $V = \{x_2^2x_3\}$.
2. $V_2' = \{x_2^2\}$. Its minimal Janet basis is $\{x_2^2\}$, and we have $V = \{x_2^2x_3, x_2^2x_3^2\}$.
3. $V_3' = \{x_1^2, x_2^2\}$. Its minimal Janet basis is $\{x_1^2, x_1^2x_2, x_2^2\}$.

The final output is the minimal Janet basis $V = \{x_2^2x_3, x_2^2x_3^2, x_1^2x_3^3, x_1^2x_2x_3^3, x_2^2x_3^3\}$.

4. A recursive Janet-like completion algorithm

Janet's criterion (Theorem 3.1) has been generalized in (Hashemi et al., 2023) to Janet-like bases. Based on the results established in that paper, we propose an algorithm analogous to Algorithm 1 for constructing Janet-like bases. First, we introduce some notations. If $U = \{t_1, \ldots, t_m\}$ is a set of terms, then there exists a sequence of natural numbers $\lambda_0, \ldots, \lambda_\ell$ with ℓ depending on U such that each λ_i is the x_n-degree of some term $t_j \in U$ and such that conversely for each $t_j \in U$ there is a λ_i which is the x_n-degree of t_j.

Theorem 4.1 ((Hashemi et al., 2023, Theorem 3.14)). *Let $U = \{t_1, \ldots, t_m\} \subset \mathcal{P}$ be a set of terms and let $\lambda_0 < \lambda_1 < \cdots < \lambda_\ell$ be natural numbers encoding the x_n-degrees appearing in U. For each index $0 \le i \le \ell$, let $U_{\lambda_i} \subseteq U$ be the subset of terms of U having x_n-degree λ_i and set $U'_{\lambda_i} = \{t/x_n^{\lambda_i} \mid t \in U_{\lambda_i}\}$. Then U is a Janet-like basis of the ideal it generates if and only if the following two conditions are satisfied:*

(i) Every set U'_{λ_i} is a Janet-like basis of the monomial ideal $\langle U'_{\lambda_i}\rangle \subseteq \mathcal{K}[x_1, \ldots, x_{n-1}]$.
(ii) For each $0 \le i < \ell$, the inclusion $U'_{\lambda_i} \subset \langle U'_{\lambda_{i+1}}\rangle$ holds.

Theorem 4.2 ((Hashemi et al., 2023, Theorem 3.17)). *Keeping the notations of Theorem 4.1, let U be a Janet-like basis for the ideal it generates. Then, U is minimal if and only if the following conditions are satisfied:*

(i) For each $i \le \ell$, U'_{λ_i} is a minimal Janet-like basis.
(ii) For each $i < \ell$, we have $\langle U'_{\lambda_i}\rangle \neq \langle U'_{\lambda_{i+1}}\rangle$.

Here, similar to Algorithm 1, we present an adapted version based on these results to compute minimal Janet-like bases: Algorithm 2.

Algorithm 2: RecMinimalJanetLikeCompletion

Data: A polynomial ring $\mathcal{P} = \mathcal{K}[x_1, \ldots, x_n]$ and a finite $U \subset \mathcal{P}$ of inter-reduced terms
Result: The minimal Janet-like basis of the ideal $\langle U \rangle$

begin
  $U \longleftarrow \{t_1, \ldots, t_m\}$
  if $n = 1$ ***or*** $m = 1$ **then**
    return U
  $(\lambda_0, \lambda_1, \ldots, \lambda_\ell) \longleftarrow$ the sequence of x_n-degrees of the terms in U ordered such that $\lambda_0 < \lambda_1 < \cdots < \lambda_\ell$
  $V \longleftarrow \emptyset$ and $V'_{\lambda_{-1}} \longleftarrow \emptyset$
  for $i = 0, \ldots, \ell$ **do**
    $V'_{\lambda_i} \longleftarrow \{t \in \mathcal{K}[x_1, \ldots, x_{n-1}] \mid t \cdot x_n^{\lambda_i} \in U\}$
    $V'_{\lambda_i} \longleftarrow \mathrm{Gen}\left(\langle V'_{\lambda_i} \cup V'_{\lambda_{i-1}} \rangle\right)$
    $V'_{\lambda_i} \longleftarrow$ `RecMinimalJanetLikeCompletion`$(\mathcal{K}[x_1, \ldots, x_{n-1}], V'_{\lambda_i})$
    $V \longleftarrow V \cup \{t x_n^{\lambda_i} \mid t \in V'_{\lambda_i}\}$
  return (V)

Theorem 4.3. *Algorithm 2 terminates in finitely many steps and returns the minimal Janet-like basis for the ideal generated by its input set. Its arithmetic complexity is* $O\left(m^2 n^2 d\right)$ *where m denotes the size of the input set* $U = \{t_1, \ldots, t_m\}$*, n is the number of variables, and d is the maximum of* $\{\deg_i(t_1), \ldots, \deg_i(t_m) \mid i = 1, \ldots, n\}$.

Proof. The proofs of the claims proceed similarly to the proof of Theorem 3.4 and are therefore omitted. □

Example 4.4. We illustrate the steps of Algorithm 2 for the inter-reduced set $U = \{x_2^2 x_3,\ x_1^2 x_3^3\} \subset \mathcal{K}[x_1, x_2, x_3]$. Keeping the notations used in the algorithm, at the first recursion level in $\mathcal{K}[x_1, x_2]$ one obtains:

1. $V'_1 = \{x_2^2\}$. Its minimal Janet-like basis is $\{x_2^2\}$ and $V = \{x_2^2 x_3\}$.
2. $V'_3 = \{x_1^2, x_2^2\}$. Its minimal Janet-like basis is $\{x_1^2, x_2^2\}$.

The final output is the minimal Janet-like basis $V = \{x_2^2 x_3, x_1^2 x_3^3, x_2^2 x_3^3\}$.

5. Staggered linear bases and Janet bases

In this section, we aim to explore the relationship between staggered linear bases and Janet bases. Understanding this relationship enables us to develop a new algorithm for testing whether a given set of terms forms a Janet basis. Furthermore, we will describe an additional algorithm for Janet completion based on this insight. To begin, let us revisit the definition of staggered linear bases. For a polynomial $f \in \mathcal{P}$ and an arbitrary set $\mathcal{T}' \subseteq \mathcal{T}$ of terms, we define $\mathcal{T}' \cdot f = \{t \cdot f \mid t \in \mathcal{T}'\}$. Given an ideal $\mathcal{I} \subset \mathcal{P}$, we can consider it as a $\mathcal{K}$-linear subspace of $\mathcal{P}$. an staggered linear basis is a basis for this vector space. More precisely, we have:

Definition 5.1. Let $\mathcal{I} = \langle f_1, \dots, f_s \rangle \subset \mathcal{P}$ be a polynomial ideal and $\mathbf{B} = \bigcup_{i=1}^{s} \mathbb{A}_i \cdot f_i$ for sets $\mathbb{A}_1, \dots, \mathbb{A}_s \subseteq \mathcal{T}$ of terms. Then, $\mathbf{B}$ is called a *staggered linear basis* for $\mathcal{I}$, if the following conditions hold:

(1) for each $t_j \in \mathcal{T} \setminus \mathbb{A}_j$ there exists $t_i \in \mathbb{A}_i$ with $i < j$ such that $t_i \cdot \mathrm{lt}(f_i) = t_j \cdot \mathrm{lt}(f_j)$,

(2) for each $t_j \in \mathcal{T} \setminus \mathbb{A}_j$ and each $t \in \mathcal{T}$ it holds $t \cdot t_j \in \mathcal{T} \setminus \mathbb{A}_j$,

(3) for each $t_i \in \mathbb{A}_i, t_j \in \mathbb{A}_j$ with $i \neq j$ we have $t_i \cdot \mathrm{lt}(f_i) \neq t_j\, \mathrm{lt}(f_j)$.

Using the above defined notations, the set $\mathbb{A}_i$ associated with each polynomial f_i is referred to as the set of *allowed terms* for f_i, whereas the complement of $\mathbb{A}_i$; i.e. $\mathcal{T} \setminus \mathbb{A}_i$, is known as the set of *forbidden terms* for f_i. By applying the second condition in this definition and Dickson's lemma, we can conclude that there exist finitely many terms $t_1, \dots, t_\ell \in \mathcal{T} \setminus \mathbb{A}_i$ such that $\langle t_1, \dots, t_\ell \rangle = \mathcal{T} \setminus \mathbb{A}_i$. For convenience, we denote $\{t_1, \dots, t_\ell\}$ by $\mathbb{F}_i$. Consequently, we can express $\mathbb{A}_i$ as $\mathbb{A}_i = \mathcal{T} \setminus \langle \mathbb{F}_i \rangle$. In the remainder of the paper, instead of denoting an staggered linear basis for the ideal $\mathcal{I}$ as $\mathbb{A}_1 \cdot f_1 \cup \cdots \cup \mathbb{A}_s \cdot f_s$, where each $\mathbb{A}_i \cdot f_i$ is possibly infinite, we adopt the notation $\{(f_1, \mathbb{F}_1), \dots, (f_s, \mathbb{F}_s)\}$, where $\mathbb{F}_i \subset \mathcal{T}$ for each i is a finite set. Moreover, we assume that $\mathbb{F}_i$ for each i, is the minimal generating set of the ideal $\langle \mathbb{F}_i \rangle$. For each i, the set $\mathbb{A}_i \cdot f_i$ is called a *cone*, and f_i is said to be the *vertex* of the cone. For simplicity, we use the abbreviation SLB to refer to "staggered linear basis". Furthermore, instead of the polynomials $f_1, \dots, f_s$, we consider the terms $t_1, \dots, t_s$. For more details on SLB's, see (Hashemi and Möller, 2023). As a direct consequence of the definition of SLB's, we can derive the following useful lemma.

Lemma 5.2. *Let $\mathcal{J} = \langle t_1, \dots, t_s \rangle$. Then, $\mathbf{B} := \{(t_1, \mathbb{F}_1), \dots, (t_s, \mathbb{F}_s)\}$ is an SLB for $\mathcal{J}$ if and only if $\mathbb{F}_i$ is the minimal generating set for $\langle \mathrm{lcm}(t_1, t_i)/t_i, \dots, \mathrm{lcm}(t_{i-1}, t_i)/t_i \rangle$ for each i.*

Proof. Assume that $\mathbf{B}$ is an SLB. From Definition 5.1, we have $\mathbb{F}_1 = \emptyset$. Let $m \in \langle \mathrm{lcm}(t_1, t_i)/t_i, \dots, \mathrm{lcm}(t_{i-1}, t_i)/t_i \rangle$. If $m \in \langle \mathbb{F}_i \rangle$, we are done. Otherwise, we have $m \cdot t_i = u \cdot t_\ell$ for some $\ell < i$ and some term u. It follows that $u \in \langle \mathbb{F}_\ell \rangle$, which contradicts the first item in Definition 5.1. Now, suppose that $m \in \langle \mathbb{F}_i \rangle$. Then, there exists $\ell < i$ and a term u such that $m \cdot t_i = u \cdot t_\ell$. This implies that $m \in \langle \mathrm{lcm}(t_1, t_i)/t_i, \dots, \mathrm{lcm}(t_{i-1}, t_i)/t_i \rangle$.

The converse is an easy application of the basic algorithm described in (Hashemi and Möller, 2023) for computing SLB's which concludes the proof. □

Remark 5.3. It is easy to see that $\langle \mathrm{lcm}(t_1, t_i)/t_i, \dots, \mathrm{lcm}(t_{i-1}, t_i)/t_i \rangle = \langle t_1, \dots, t_{i-1} \rangle : t_i$.

Let us continue with a simple example of an SLB. We consider the ideal $\mathcal{I} = \langle x_1^3 x_3^3, x_1 x_2^8 \rangle$ in the ring $\mathcal{K}[x_1, x_2, x_3]$ and the lexicographical ordering on this ring given by $x_1 \prec x_2 \prec x_3$. The generating set of $\mathcal{I}$ is arranged according to this ordering, from greatest to lowest. Then, using Lemma 5.2, one observes that the set $\mathbf{B} = \{(x_1^3 x_3^3, \{\}), (x_1 x_2^8, \{x_1^2 x_3^3\})\}$ is an SLB for $\mathcal{I}$. Now, we will try to transform this SLB into a new SLB such that each set of forbidden terms consists of variables[2]. For this purpose, we divide each forbidden term by the highest variable (with respect to $\prec$) appearing in it. So, we decompose $x_1^2x_3^3$ into $x_1^2x_3^2$ and x_3. Then, we add the new element $(x_1x_2^8x_3, \{x_1^2x_3^2\})$ and the new SLB becomes

$$\left\{(x_1^3x_3^3, \{\}), (x_1x_2^8x_3, \{x_1^2x_3^2\}), (x_1x_2^8, \{x_1^2x_3^3, x_3\})\right\}.$$

Now, we shall update the middle element. Thus, we get

$$\left\{(x_1^3x_3^3, \{\}), (x_1x_2^8x_3^2, \{x_1^2x_3\}), (x_1x_2^8x_3, \{x_1^2x_3^2, x_3\}), (x_1x_2^8, \{x_1^2x_3^3, x_3\})\right\}.$$

The critical step is here when we would like to update the second element. If we multiply it by x_3 then the order (with respect to $\prec$) of the elements is changed and we shall update sets of forbidden terms. So, the new SLB that we find is

$$\left\{(x_1x_2^8x_3^3, \{\}), (x_1^3x_3^3, \{x_2^8\}), (x_1x_2^8x_3^2, \{x_1^2x_3, x_3\}), (x_1x_2^8x_3, \{x_1^2x_3^2, x_3\}), (x_1x_2^8, \{x_1^2x_3^3, x_3\})\right\}.$$

If we repeat this process by multiplying the second element by the powers of x_2, then the set of vertices of the final SLB (such that each set of forbidden terms consists of only variables) forms the desired minimal Janet basis of $\mathcal{I}$. From this discussion, we conclude several observations as follows.

Lemma 5.4. *Let* $\mathbf{B} = \{(t_1, \mathbb{F}_1), \ldots, (t_s, \mathbb{F}_s)\}$ *be an SLB for the ideal* $\mathcal{J}$ *generated by the terms* $t_1, \ldots, t_s$*. Assume that there exist a term* m *and integers* i, j *such that* m *is a term in* $\mathbb{F}_j$ *of degree at least* 2 *and* $x_i \mid m$*. If in* $\mathbf{B}$ *we replace* $(t_j, \mathbb{F}_j)$ *by*

$$\left(x_i \cdot t_j, \mathrm{Gen}(\langle \mathbb{F}_j \rangle : x_i)\right) \cup \left(t_j, \mathrm{Gen}(\langle \mathbb{F}_j \cup \{x_i\}\rangle)\right)$$

then we obtain an SLB for $\mathcal{J}$*.*

Proof. We must show that $(t_j, \mathbb{F}_j) = (t_j, \mathrm{Gen}(\langle \mathbb{F}_j \cup \{x_i\}\rangle)) \cup (t_j \cdot x_i, \mathrm{Gen}(\langle \mathbb{F}_j \rangle : x_i))$, where $\mathrm{Gen}(A)$ returns the minimal generating set for the monomial ideal A. We analyze the elements of both sides. Let $m = t_j \cdot v$ be an element of $(t_j, \mathbb{F}_j)$ where $v \notin \langle \mathbb{F}_j \rangle$. We consider two cases:

1. If $x_i \nmid v$, then m clearly belongs to the first cone on the right-hand side.
2. If $x_i \mid v$, we can express m as $t_j \cdot x_i \cdot u$ for some term u. Since $v \notin \langle \mathbb{F}_j \rangle$, it follows that $u \notin \langle \mathbb{F}_j \rangle : x_i$, which means m is in the second cone on the right-hand side.

Conversely, since $\mathbb{F}_j$ is a subset of both $\mathbb{F}_j \cup \{x_i\}$ and $\langle \mathbb{F}_j \rangle : x_i$, any element on the right-hand side must also belong to the left-hand side. This completes the proof. □

By repeating the process defined in this lemma, we can easily see that after finitely many steps we move towards an SLB such that the set of forbidden terms of each cone is generated by a subset of variables. This leads to the following definition.

[2] In the subsequent examples, to clarify the main idea, we may not perform the minimization process on the set of forbidden terms.

Definition 5.5. Let $\mathbf{B} = \{(t_1, \mathbb{F}_1), \dots, (t_s, \mathbb{F}_s)\}$ be an SLB for the ideal $\mathcal{J}$ generated by terms $t_1, \dots, t_s$. $\mathbf{B}$ is called *linear* if for each i, $\mathbb{F}_i$ contains only variables.

A natural question that may arise here is the following: If, in obtaining a linear SLB by applying Lemma 5.4, we consider x_i, the highest variable that divides m, at each step, does the obtained linear SLB constitute a Janet basis as we saw in the previous example? The next example demonstrates that this is not true in general:

Example 5.6. Consider the ideal $\mathcal{I} = \langle x_2x_4, x_2x_3, x_1x_3\rangle \subset \mathbb{Q}[x_1, x_2, x_3, x_4]$. Then $\mathbf{B} = \{(x_2x_4, \{\}), (x_2x_3, \{x_4\}), (x_1x_3, \{x_2\})\}$ is a linear SLB for $\mathcal{I}$. However, the set of vertices of $\mathbf{B}$ is not a Janet basis, and the correct Janet basis is obtained by adding the new elements $\{x_2x_3x_4, x_1x_3x_4\}$ to the generating set of $\mathcal{I}$.

We should note, however, that when we apply a specific minimization process to linearize the $\mathbb{F}_i$'s, we can successfully construct a Janet basis. Specifically, in our example, the set of forbidden terms for the last element of $\mathbf{B}$ is $\{x_2x_4, x_2\}$ and thus, during the customary minimization, x_2 eliminates x_2x_4. However, if we restrict the minimization process so that only certain variables dividing other terms can eliminate them, using the following criterion, we can properly guide the construction of the Janet basis: We say that a variable x_i can eliminate a term m if $x_i \mid m$ and additionally x_i is the greatest variable appearing in m. In the remainder of this section, we refer to this type of minimization as *Janet minimization*. Note that we do not employ any other form of minimization in this process. For example, a term of degree 2 like x_1x_2 cannot eliminate another term like $x_1^2x_2$. In this example, only x_2 can remove $x_1^2x_2$. Now, let us use this minimization to linearize an SLB.

Example 5.7. Consider the ideal $\mathcal{I} = \langle x_2x_4, x_2x_3, x_1x_3\rangle \subset \mathbb{Q}[x_1, x_2, x_3, x_4]$. The generating set of the ideal is organized in descending order according to the lexicographical ordering defined on this ring, where $x_1 \prec \cdots \prec x_4$. An SLB for this ideal is given by $\mathbf{B} = \{(x_2x_4, \{\}), (x_2x_3, \{x_4\}), (x_1x_3, \{x_2x_4, x_2\})\}$. By performing the ordinary minimal basis for the last element, we observe that it yields a linear SLB, which does not give rise to a Janet basis. However, through the process of Janet minimization, we conclude that $\mathbf{B}$ is not linear. The application of new splittings results in

$$\{(x_2x_3x_4, \{\}), (x_1x_3x_4, \{x_2\}), (x_2x_4, \{x_3\}), (x_2x_3, \{x_4\}), (x_1x_3, \{x_2\})\}$$

Notably, the set of vertices from this SLB corresponds to a Janet basis for the ideal $\mathcal{I}$.

In the following theorem, we will outline the conditions under which a linear SLB constitutes a Janet basis, under the assumption that each $\mathbb{F}_i$ has been minimized through the Janet minimization process.

Theorem 5.8. *Let $H = \{t_1, \dots, t_s\} \subset \mathcal{T}$ and $\mathbf{B} = \{(t_1, \mathbb{F}_1), \dots, (t_s, \mathbb{F}_s)\}$ be an SLB for the ideal $\mathcal{J}$ generated by H. Furthermore, assume that $t_s \prec \cdots \prec t_1$ where $\prec$ stands for the lexicographical ordering with $x_1 \prec \cdots \prec x_n$.*

(1) *If $\{t_1, \dots, t_s\}$ is a Janet basis then $\mathbf{B}$ is linear and for each i, $\mathbb{F}_i$ is the set of Janet non-multiplicative variables for t_i.*

(2) *If $\mathbf{B}$ is linear by performing Janet minimization process then H is a Janet basis for $\mathcal{J}$.*

Proof. (1) First, we show that for every i, and for every non-multiplicative variable $x_j \in \mathrm{NM}_J(t_i, H)$ we have $x_j \in \langle t_1, \ldots, t_{i-1} \rangle : t_i$. Since H is a Janet basis, for every non-multiplicative prolongation $x_j \cdot t_i$ there exists a unique Janet divisor $t_\ell \in H$. By definition of Janet division, we conclude that $\deg_m(t_i) = \deg_m(t_\ell)$ for all $m = j+1, \ldots, n$ and $\deg_j(t_i) = \deg_j(t_\ell) - 1$. It follows that $t_i \prec_{lex} t_\ell$ and consequently $\ell < i$. Therefore, $x_j \in \langle t_1, \ldots, t_{i-1} \rangle : t_i$. By Lemma 5.2, we know that $\mathbb{F}_i$ is the generating set of this ideal, and in turn $x_j \in \mathbb{F}_i$.

Now, we show that for each i, and every variable $x_j \in \mathbb{F}_i$, we have $x_j \in \mathrm{NM}_J(t_i, H)$. Again, applying Lemma 5.2 implies that $x_j \in \langle t_1, \ldots, t_{i-1} \rangle : t_i$ and consequently $x_j \cdot t_i \in \langle t_1, \ldots, t_{i-1} \rangle$. Then, there exist $\ell < i$ and a term u such that $x_j \cdot t_i = u \cdot t_\ell$. Assume that ℓ is the lowest integer satisfying this property. From the ordering on the set H, we have $t_i \prec_{lex} t_\ell$. It follows that u contains only the variables $x_1, \ldots, x_j$. Moreover, we deduce that $\deg_m(t_i) = \deg_m(t_\ell)$ for all $m = j+1, \ldots, n$ and $\deg_j(t_i) = \deg_j(t_\ell) - 1$. This shows that $x_j \in \mathrm{NM}_J(t_i, H)$.

To complete the proof of (1), it remains to show that for each i, $\mathbb{F}_i$ contains only variables. Assume that $u \in \mathbb{F}_i$ and x_ℓ, x_j both divide u with $\ell < j$. From the above discussion, it is seen that x_j lies in $\mathrm{NM}_J(t_i, H)$. Since H is a Janet basis, $x_j \cdot t_i$ has a unique Janet divisor; specifically, there exist $\ell < i$ and a term u such that $x_j \cdot t_i = u \cdot t_\ell$ and u contains only Janet multiplicative variables of t_ℓ. This shows that $x_j \in \langle t_1, \ldots, t_{i-1} \rangle : t_i$, and therefore, it belongs to $\mathbb{F}_i$ which contradicts the minimality of $\mathbb{F}_i$. All these arguments prove (1).

(2) We must demonstrate that for every i, and for every non-multiplicative variable $x_j \in \mathrm{NM}_J(t_i, H)$, the product $x_j \cdot t_i$ has a Janet divisor, i.e. there exist an index $\ell < i$ and a term $u \in \mathcal{K}[\mathrm{M}_J(t_\ell, H)]$ such that $x_j \cdot t_i = u \cdot t_\ell$. We proceed by induction on i. Since t_1 is the greatest element with respect to $\prec_{lex}$, then every variable is Janet multiplicative for t_1. Now, suppose that the claim holds true for $t_1, \ldots, t_{i-1}$. From the fact that $x_j \in \mathrm{NM}_J(t_i, H)$, it follows that there exists an index ℓ such that $t_i, t_\ell \in U_{[d_{j+1}, \ldots, d_n]}$ where for each k, d_k represents $\deg_{x_k}(t_i)$ and $\deg_{x_j}(t_i) < \deg_{x_j}(t_\ell)$. Thus, $t_i \prec_{lex} t_\ell$ and $\ell < i$. Assume that ℓ is the largest integer satisfying these properties. This implies that there are terms u, v such that $u \cdot x_j \cdot t_i = v \cdot t_\ell$ and therefore $u \cdot x_j \in \langle t_1, \ldots, t_{i-1} \rangle : t_i$. Note that x_j has the highest index compared to all the variables dividing u. Since **B** is linear by performing Janet minimization process then x_j must appear in $\mathbb{F}_i$. Hence $x_j \in \langle t_1, \ldots, t_{i-1} \rangle : t_i$, and there exist integer $m < i$ and a term w such that $x_j \cdot t_i = w \cdot t_m$. If w includes only Janet multiplicative variables of t_m we are done. Otherwise, it contains a Janet non-multiplicative variable x_k. By applying the induction hypothesis the claim is proved. □

Applying this theorem, we describe the following effective Janet basis test.

Theorem 5.9. *Algorithm 3 terminates in finitely many steps and is correct. Furthermore, its arithmetic complexity is $O(ns^2)$ where again n is the number of variables and s the size of the input set.*

Proof. The termination of the algorithm is straightforward, and its correctness follows from Theorem 5.8. We now address its complexity. At the beginning of the algorithm, we sort $t_1, \ldots, t_s$, which requires $O(ns \log(s))$ operations. This cost can be disregarded in the subsequent analysis.

Algorithm 3: JanetTest

```
Data: A finite set of terms {t_1, ..., t_s}
Result: Is {t_1, ..., t_s} a Janet basis?
begin
    (t_1, ..., t_s) ⟵ The sorted form of t_1, ..., t_s with respect to ≺_lex
    F_i ⟵ The minimal elements of {lcm(t_i, t_j)/t_i | j = 1, ..., i − 1} for any i
      following Janet minimization process
    if all F_i's are linear then
        return true
    else
        return false
```

Next, to construct the sets $\mathbb{F}_i$ for each i, we will perform two operations. First, we compute $\{\mathrm{lcm}(t_i, t_j)/t_i \mid j = 1, \ldots, i-1\}$ for each i, which can be accomplished in $O(n \cdot i)$ operations. Additionally, sorting this set and performing the Janet minimization process will require the same complexity. Therefore, the total number of arithmetic operations required for all i is $O(ns^2)$, which concludes the proof. □

Remark 5.10. For the efficiency of this algorithm and to execute the Janet minimization process, we first compute the set $A_i := \{\mathrm{lcm}(t_i, t_j)/t_i \mid j = 1, \ldots, i-1\}$ for each $i = 1, \ldots, s$. Next, we identify the highest variable x_j appearing in A_i. If x_j is present in A_i, we remove all elements in A_i where x_j is their highest variable and continue examining the subsequent variable. Finally, if A_i satisfies the required conditions, we proceed to the next index, $i + 1$. If, during the process, we encounter an obstruction, the algorithm returns false; otherwise, it concludes successfully and returns true.

Remark 5.11. Assume that $d \geq 2$ is the average of the differences between the maximal and minimal degrees of the t_i's with respect to each of the variables. In (Hashemi et al., 2023, Theorem 3.6) the complexity $O(\max\{s^2 + ns, n^2(s-d)^2/4 + ns^2\})$ is provided for the Janet basis test. As observed, Algorithm 3 exhibits a lower complexity.

Based on Theorem 5.8, we are able to describe a new algorithm for the Janet completion as well.

Theorem 5.12. *Algorithm 4 terminates in finitely many steps and outputs a minimal Janet basis for the ideal $\langle U \rangle$. Moreover, the arithmetic complexity of this algorithm is $O(nl^2 \log(l))$ where again n is the number of variables and l the size of the (output) minimal Janet basis of this ideal.*

Proof. At the beginning of the algorithm, $\mathbb{F}_1, \ldots, \mathbb{F}_k$ contains only finitely many terms. Furthermore, any newly added term is distinct from the existing ones and divides $\mathrm{lcm}(t_1, \ldots, t_k)$. These arguments establish the finite termination of the algorithm.

Next, we prove the correctness of the algorithm. Since, at the end of the algorithm, $\mathbf{B} := \{(t_1, \mathbb{F}_1), \ldots, (t_l, \mathbb{F}_l)\}$ satisfies the conditions of item (2) in Theorem 5.8, we conclude that $A := \{t_1, \ldots, t_l\}$ is a Janet basis for the ideal it generates. Moreover, by the structure of the algorithm, this ideal is equal to $\langle U \rangle$. Additionally, $\mathbb{F}_i$ represents the

Algorithm 4: MinimalJanetCompletion

Data: A polynomial ring $\mathcal{P} = \mathcal{K}[x_1, \ldots, x_n]$ and a finite set $U \subset \mathcal{P}$ of inter-reduced terms

Result: A minimal Janet basis for $\langle U \rangle$ along with the set of non-multiplicative variables for each element in the basis

begin
- $(t_1, \ldots, t_k) \longleftarrow$ The sorted form of U with respect to $\prec_{lex}$
- $i \longleftarrow k$
- **while** $i > 0$ **do**
  - $\mathbb{F}_i \longleftarrow$ The minimal elements of $\{\text{lcm}(t_i, t_j)/t_i \mid j = 1, \ldots, i-1\}$ following Janet minimization process
  - **while** $\{m \in \mathbb{F}_i \mid \deg(m) > 1\} \neq \emptyset$ **do**
    - Select the highest variable $x_j$with respect to $\prec$ in $\mathbb{F}_i$ such that $x_j \notin \mathbb{F}_i$
    - $k \longleftarrow k + 1$ and $i \longleftarrow k$
    - $t_k \longleftarrow x_j \cdot t_i$
    - $\mathbb{F}_i \longleftarrow \mathbb{F}_i \setminus \{m \in \mathbb{F}_i \text{ s.t. } x_j \mid m\} \cup \{x_j\}$
  - $(t_1, \ldots, t_k) \longleftarrow$ The sorted form of $t_1, \ldots, t_k$ with respect to $\prec_{lex}$
  - $i \longleftarrow i - 1$
- **return** $\{(t_1, \mathbb{F}_1), \ldots, (t_k, \mathbb{F}_k)\}$

set of all Janet non-multiplicative variables for t_i. We now prove that A is the minimal Janet basis. Assume, for the sake of contradiction, that t_i is the highest element with respect to $\prec_{lex}$ in A and that it is redundant. This situation cannot arise at the beginning of the algorithm, as the input set U is inter-reduced. Further, let us assume that t_ℓ is the unique element in A that is a Janet divisor of t_i, considering t_ℓ in $A \setminus \{t_i\}$. Given the specific ordering utilized in A, we must have $i < \ell$. Consequently, there exists a term u, which involves only Janet multiplicative variables of t_ℓ, as an element of $A \setminus \{t_i\}$ such that $t_i = u \cdot t_\ell$. It follows that $u \in \langle t_1, \ldots, t_{\ell-1} \rangle : t_\ell$. Let x_j be the largest variable dividing u. Due to the specialized form of minimization we employ, x_j must belong to $\mathbb{F}_\ell$, which implies that $x_j \cdot t_\ell$ appears in A. Based on the assumption regarding t_ℓ, we deduce that $t_i = x_j \cdot t_\ell$. From the structure of the algorithm (specifically, the condition of the inner **while**-loop), t_i cannot be produced by t_ℓ. Thus, there exists another element t_k with $k > \ell$ such that $t_i = x_r \cdot t_k$. Furthermore, t_k creates t_i and is generated before t_i. Next, two cases arise. If t_ℓ exists when t_i is created, then $x_r \in \mathbb{F}_k$ and t_i would not be produced. Otherwise, t_ℓ would not be created. These arguments demonstrate that H is minimal.

Finally, we will address the complexity analysis of the algorithm. Similar to the proof of Theorem 5.9, for each k, sorting $t_1, \ldots, t_k$ and computing $\mathbb{F}_k$ using the Janet minimization process requires $O(nk \log(k) + nk) = O(nk \log(k))$ operations. Since k

varies from s to l, we get

$$\sum_{k=s}^{l} nk\log(k) \leq n\log(l)\sum_{k=s}^{l} k$$

and this inequality gives the claimed complexity bound. □

Remark 5.13. Note that the complexity bound provided in this theorem includes the size of the output basis, which is different from the bound stated in Theorem 3.4.

6. Staggered linear bases and Janet-like bases

In this section, we aim to explore the relationship between SLB's and Janet-like bases. To this end, we directly extend the results from the previous section to Janet-like bases. We will first introduce a new algorithm to determine whether a given SLB of a monomial ideal constitutes a Janet-like basis. Subsequently, we will present a novel method for constructing the Janet-like completion from a given SLB.

Let us start with a simple example. Let $\mathcal{I} = \langle x_1^3x_3^3, x_1x_2^8\rangle \subset \mathcal{K}[x_1, x_2, x_3]$. Consider the lexicographical ordering with $x_1 \prec x_2 \prec x_3$ and order the generating set of $\mathcal{I}$ using this ordering from greatest to lowest. Then, using Lemma 5.2, one observes that the set $\mathbf{B} = \{(x_1^3x_3^3, \{\}), (x_1x_2^8, \{x_1^2x_3^3\})\}$ is an SLB for $\mathcal{I}$. Now, we will try to transform this SLB into a new SLB such that each set of forbidden terms consists of pure power of variables. For this purpose, we divide each forbidden term by the greatest power of the highest variable (with respect to $\prec$) appearing in it. So, we decompose $x_1^2x_3^3$ into x_1^2 and x_3^3. Then, we multiply $x_1x_2^8$ by x_3^3 and add the result to $\mathbf{B}$, resulting in the new ordered SLB $\{(x_1x_2^8x_3^3, \{\}), (x_1^3x_3^3, \{x_2^8\}), (x_1x_2^8, \{x_1^2x_3^3, x_3^3\})\}$. Note that, similar to the construction of Janet bases from SLB's, this sorting of elements according to the lexicographical ordering is essential for the construction of Janet-like bases. From this observation, we can extend Lemma 5.4 to the following lemma.

Lemma 6.1. *Let* $\mathbf{B} = \{(t_1, \mathbb{F}_1), \ldots, (t_s, \mathbb{F}_s)\}$ *be an SLB for the ideal* $\mathcal{J}$ *generated by the terms* $t_1, \ldots, t_s$*. Assume that there exists a mixed term*[3] m *and integers* i, j *such that* $m \in \mathbb{F}_j$ *and* $x_i \mid m$*. Let* a *be the largest integer such that* $x_i^a \mid m$*. If in* $\mathbf{B}$ *we replace* $(t_j, \mathbb{F}_j)$ *by*

$$\left(x_i^a \cdot t_j, \mathrm{Gen}(\langle \mathbb{F}_j\rangle : x_i^a)\right) \cup \left(t_j, \mathrm{Gen}(\langle \mathbb{F}_j \cup \{x_i^a\}\rangle)\right),$$

then we obtain an SLB for $\mathcal{J}$*.*

Proof. It follows similar lines to the proof of Lemma 5.4. □

Definition 6.2. Let $\mathbf{B} = \{(t_1, \mathbb{F}_1), \ldots, (t_s, \mathbb{F}_s)\}$ be an SLB for the ideal $\mathcal{J}$ generated by terms $t_1, \ldots, t_s$. $\mathbf{B}$ is called *irreducible* if for each i, the ideal generated by each $\mathbb{F}_i$ is irreducible.

[3] In this paper, we refer to a term as *mixed* if it is divisible by at least two different variables.

Let us fix some assumptions that we use in the rest of this section. Consider the lexicographic term ordering $\prec_{lex}$ with $x_1 \prec_{lex} \cdots \prec_{lex} x_n$. Let $t_1, \ldots, t_s$ be a sequence of terms such that $t_s \prec_{lex} \cdots \prec_{lex} t_1$. One can see easily, by Lemma 5.2, that the set $\mathbb{F}_j$ for each j of forbidden terms for t_j is computed as the minimal generating set of the ideal

$$\langle \mathrm{lcm}(t_i, t_j)/t_j \mid i = 1, \ldots, j-1 \rangle.$$

Now, the natural question that may arise is whether every Janet-like basis is an irreducible SLB?

Lemma 6.3. *Let* $\mathbf{B} = \{(t_1, \mathbb{F}_1), \ldots, (t_s, \mathbb{F}_s)\}$ *be an SLB for the ideal* $\mathcal{J}$ *generated by terms* $t_1, \ldots, t_s$ *where* $t_s \prec_{lex} \cdots \prec_{lex} t_1$*. If* $T = \{t_1, \ldots, t_s\}$ *forms a Janet-like basis, then* $\mathbf{B}$ *is irreducible.*

Proof. We proceed by arguing reductio ad absurdum. Assume thus that $\mathbf{B}$ is not irreducible. Let t_ℓ be the maximum element with respect to $\prec_{lex}$ such that $\langle \mathbb{F}_\ell \rangle$ is not an irreducible ideal. Assume that x_i is the highest variable appearing in $\mathbb{F}_\ell$ such that there is a mixed term in $\mathbb{F}_\ell$ containing x_i. Assume that $x^\alpha \in \mathbb{F}_\ell$ is the minimum term among all these terms. Thus, x^α is of the form $x_1^{\alpha_1} \cdots x_i^{\alpha_i}$ and $\alpha_i \neq 0$. These assumptions imply that $x_i^{\alpha_i}$ is a non-multiplicative power for t_ℓ. Since T is a Janet-like basis, we conclude that $x_i^{\alpha_i} \cdot t_\ell$ has a Janet-like divisor in T. It follows that there exists $t_m \in T$ and a term u such that $u \cdot t_m = x_i^{\alpha_i} \cdot t_\ell$. On the other hand, $t_\ell \prec_{lex} t_m$ and in turn, by definition, we have $x_i^{\alpha_i} \in \langle \mathbb{F}_\ell \rangle$; leading to a contradiction. □

The next question that may arise is whether any irreducible SLB with respect to lexicographic order forms a Janet-like basis? The example below shows that the answer to this question is generally negative, and in the subsequent part of this section, we will investigate how we can derive a Janet-like basis from a given irreducible SLB.

Example 6.4. An irreducible SLB does not give in general a Janet-like basis. In the polynomial ring $\mathcal{K}[x_1, x_2, x_3]$ with a lexicographic term ordering induced by $x_1 \prec x_2 \prec x_3$, consider the monomial ideal $\mathcal{I} = \langle G \rangle = \langle t_1 = x_3^4, t_2 = x_2 x_3^3, t_3 = x_2^3 x_3^2, t_4 = x_1 x_2 x_3, t_5 = x_1^3 x_3, t_6 = x_1^3 x_2^3 \rangle$. Its generators are sorted from lex-largest to smallest. Using Lemma 6.1, we obtain the SLB

$$\mathbf{B}_0 = \{(x_3^4, \emptyset), (x_2 x_3^3, \{x_3\}), (x_2^3 x_3^2, \{x_3\}), (x_1 x_2 x_3, \{x_2^2 x_3, x_3^2\}), (x_1^3 x_3, \{x_2, x_3^3\}), (x_1^3 x_2^3, \{x_3\})\}.$$

In total, only one mixed term appears in these sets; $x_2^2 x_3 \in \mathbb{F}_4$. Thus, we pick the lex-largest pure variable power that divides it, x_3, and add the term $u := x_3 t_4$ as a generator. We can sort the enlarged generating set $G_1 := G \cup \{u\}$ using the lex-order again. Thus, u is inserted between t_3 and t_4 and we get

$$\mathbf{B}_1 = \{(x_3^4, \emptyset), (x_2 x_3^3, \{x_3\}), (x_2^3 x_3^2, \{x_3\}), (x_1 x_2 x_3^2, \{x_2^2, x_3\}),$$
$$(x_1 x_2 x_3, \{x_3\}), (x_1^3 x_3, \{x_2, x_3^3\}), (x_1^3 x_2^3, \{x_3\})\}.$$

One sees that no mixed terms appear in the $\mathbb{F}_i$'s and in turn $\mathbf{B}_1$ is an irreducible SLB, however, G_1 is *not yet* the Janet-like basis of $\mathcal{I}$. In the following, we will show how we can apply a method similar to Janet minimization process introduced in Section 5 to derive a Janet-like basis from this SLB. In this direction, for each i we write a complete description

$$H_i := \{\mathrm{lcm}(t_i, t_j)/t_i \mid j = 1, \dots, i-1\}$$

rather than just its minimal generating set $\mathbb{F}_i$, and this entails

$$\mathbf{B}_2 = \{(x_3^4, \emptyset), (x_2x_3^3, \{x_3\}), (x_2^3x_3^2, \{x_3^2, x_3\}), (x_1x_2x_3^2, \{x_3^2, x_3, x_2^2\}), (x_1x_2x_3, \{x_3^3, x_3^2, x_2^2x_3, x_3\}),$$
$$(x_1^3x_3, \{x_3^3, x_2x_3^2, x_2^3x_3, x_2x_3, x_2\}), (x_1^3x_2^3, \{x_3^4, x_3^3, x_3^2, x_3^2, x_3, x_3\})\}.$$

We say that a pure power x_i^a can eliminate a term m if $x_i^a \mid m$ and additionally x_i is the greatest variable appearing in m. We refer to this kind of minimization as *Janet-like minimization*. By performing this minimization on $\mathbf{B}_2$, we obtain:

$$\mathbf{B}_3 = \{(x_3^4, \emptyset), (x_2x_3^3, \{x_3\}), (x_2^3x_3^2, \{x_3\}), (x_1x_2x_3^2, \{x_3, x_2^2\}), (x_1x_2x_3, \{x_3\}),$$
$$(x_1^3x_3, \{x_3^3, x_2x_3^2, x_2^3x_3, x_2x_3, x_2\}), (x_1^3x_2^3, \{x_3\})\}.$$

Now, we look for the lowest element in this set whose set of forbidden terms contains a mixed product of variables. We find $x_2x_3 \in H_6$. By applying Lemma 6.1, we now split this SLB by adding a new element, $(x_1^3x_3) \cdot x_3$. This yields the new SLB

$$\mathbf{B}_4 = \{(x_3^4, \emptyset), (x_2x_3^3, \{x_3\}), (x_2^3x_3^2, \{x_3\}), (x_1x_2x_3^2, \{x_3, x_2^2\}), (x_1^3x_3^2, \{x_3^2, x_2x_3, x_2\}),$$
$$(x_1x_2x_3, \{x_3\}), (x_1^3x_3, \{x_3, x_2\}), (x_1^3x_2^3, \{x_3\})\}.$$

Again, we see that $x_2x_3 \in H_5$. The final refining gives an irreducible SLB following the Janet-like minimization

$$\mathbf{B}_5 = \{(x_3^4, \emptyset), (x_2x_3^3, \{x_3\}), (x_1^3x_3^3, \{x_3, x_2\}), (x_2^3x_3^2, \{x_3\}), (x_1x_2x_3^2, \{x_3, x_2^2\}),$$
$$(x_1^3x_3^2, \{x_3, x_2\}), (x_1x_2x_3, \{x_3\}), (x_1^3x_3, \{x_3, x_2\}), (x_1^3x_2^3, \{x_3\})\}$$

and the set of vertices now provides the correct minimal Janet-like basis.

Based on this observation, we can establish an analogue of Theorem 5.8 for Janet-like bases. Since the proof is similar, we will omit it.

Theorem 6.5. *Let $H = \{t_1, \dots, t_s\} \subset \mathcal{T}$ and $\mathbf{B} = \{(t_1, \mathbb{F}_1), \dots, (t_s, \mathbb{F}_s)\}$ be an SLB for the ideal $\mathcal{J}$ generated by H. Furthermore, assume that $t_s \prec \cdots \prec t_1$ where $\prec$ denotes the lexicographical ordering with $x_1 \prec \cdots \prec x_n$.*

(1) *If $\{t_1, \dots, t_s\}$ is a Janet-like basis then $\mathbf{B}$ is irreducible and for each i, $\mathbb{F}_i$ is the set of Janet non-multiplicative powers for t_i.*

(2) *If $\mathbf{B}$ is irreducible under the Janet-like minimization process, then H is a Janet-like basis for $\mathcal{J}$.*

Utilizing Lemma 6.3 and this theorem, we can develop algorithms analogous to Algorithms 3 and 4 for the Janet-like basis test and minimal Janet-like completion. In the following sections, we will refer to these algorithms as Algorithms 5 and 6, respectively. It is important to note that the arithmetic complexity of these algorithms is similar to that of Algorithms 3 and 4.

7. Iterative Janet completion algorithms

There exists a generic iterative algorithm for computing monomial completions for any involutive division (provided they exist) using non-multiplicative prolongation (Gerdt and Blinkov, 1998, Theorem 4.14). In (Seiler, 2010, Section 4.2), it was noted that this algorithm returns a minimal involutive basis if the input is the minimal basis of the studied monomial ideal in the usual sense. Specialisations for the monomial completion of Janet bases have been presented by Gerdt et al. (2001, MonomialJanetBasis algorithm) and by Hashemi et al. (2025, Algorithm 1). The former algorithm is based on so-called Janet trees as the underlying data structure which allows for efficiently performing many relevant tasks like determining multiplicative variables or finding involutive divisors. The description of the algorithm is rather technical, as the authors work immediately with binary trees, obscuring the underlying mathematical ideas which can be more clearly seen in (Seiler, 2010, Addendum, Section 3.1). In this section, we propose a new, more efficient variant of these algorithms (along with a counterpart for Janet-like bases) where the explicit use of trees is replaced by a good ordering of the basis and where thus considerable overhead for managing the data structure is avoided.

Contrary to (Hashemi et al., 2025), our approach uses the lexicographical ordering of the terms. The use of this ordering is quite natural, as e.g. in a Janet tree the leaves contain the generators sorted lexicographically. The ordering is crucial for our algorithm, as it enables us to find the sets of non-multiplicative variables for any set of terms more efficiently based on the following result, which can be understood as capturing a key property of Janet trees without actually referring to trees. Below, we consider the lexicographical term ordering given by $x_1 \prec \cdots \prec x_n$.

Lemma 7.1. *Let $U = \{t_1, \ldots, t_k\} \subset \mathcal{P}$ be a lex-ordered set of terms, and let ℓ be the largest subindex such that $\deg_\ell(t_{i+1}) > \deg_\ell(t_i)$ for some $1 \le i < k$. Then,*

$$\mathrm{NM}_{J,U}(t_i) = \{x_\ell\} \cup \{x_j \mid j > \ell \text{ and } x_j \in \mathrm{NM}_{J,U}(t_{i+1})\}.$$

Proof. If we assume that the set of terms $U = \{t_1, \ldots, t_k\} \subset \mathcal{P}$ is ordered with respect to lex then it is clear that $\deg_j(t_{i+1}) = \deg_j(t_i)$ for all $j > \ell$. Hence, by definition, every non-multiplicative variable of t_{i+1} with subindex larger than ℓ is also non-multiplicative for t_i. Furthermore, these equalities combined with the fact that $\deg_\ell(t_{i+1}) > \deg_\ell(t_i)$ directly imply that x_ℓ is also non-multiplicative for t_i as well. Now note that there cannot exist any term $t \in U$ such that $\deg_k(t) > \deg_k(t_i)$ and $\deg_j(t) = \deg_j(t_i)$ for all $j > k$ for an index $k < \ell$; because that would imply that $t_i \prec t \prec t_{i+1}$, which contradicts our assumption. □

Theorem 7.2. *Algorithm 7 terminates in finitely many steps and outputs a minimal Janet basis for the ideal $\langle U\rangle$. Moreover, the arithmetic complexity of this algorithm is $O(nl^2 \log(l))$ where n is the number of variables and l the size of the (output) minimal Janet basis of $\langle U\rangle$.*

Proof. This algorithm follows an iterative approach similar to Algorithm 4. The steps taken by both algorithms are essentially the same. However, Algorithm 4 computes the

Algorithm 7: IterMinimalJanetCompletion

Data: A polynomial ring $\mathcal{P} = \mathcal{K}[x_1, \ldots, x_n]$ and a finite set $U \subset \mathcal{P}$ of inter-reduced terms

Result: The minimal Janet basis of $\langle U \rangle$ together with the non-multiplicative variables of each element of the basis

```
begin
    (t_1, ..., t_k) ⟵ The sorted form (from the smallest to the greatest) of U
        with respect to ≺_lex
    for i = k − 1, ..., 1 do
        ℓ ⟵ max{ j | deg_j(t_{i+1}) > deg_j(t_i) }
        NM_i ⟵ (NM_{i+1} ∩ {x_{ℓ+1}, ..., x_n}) ∪ {x_ℓ}
    NM_k ⟵ {}
    H ⟵ {(t_1, NM_1), ..., (t_k, NM_k)}
    for i = 1, ..., k do
        for x_j ∈ NM_i from the smallest to the largest variable do
            flag ⟵ false
            aux ⟵ t_i · x_j
            a ⟵ i + 1
            while flag = false or a ≤ k do
                if aux ≺ t_a then
                    exit while-loop
                if deg_j(t_a) = deg_j(t_i) + 1, deg_ℓ(t_a) ≤ deg_ℓ(t_i) ∀ℓ < j then
                    flag ⟵ true
                a ⟵ a + 1
            if flag = false then
                // aux is inserted in the a-th place to maintain the lex order
                H ⟵ H ∪ {(aux, {})}
                Adjust non-multiplicative variables NM_{i+1}, ..., NM_a
                k ⟵ k + 1
    return H
```

colon ideals to obtain the sets of non-multiplicative variables on each iteration together with enough information to determine which non-multiplicative prolongations need to be added to complete the basis. Whereas, Algorithm 7 directly computes the non-multiplicative variables once in the first lines and then it only "updates" them as needed as the algorithm progresses. Thus, the finite termination of the algorithm follows by the same argument.

Next, we prove the correctness of the algorithm. Note that by Lemma 7.1 all sets of non-multiplicative variables are correctly computed in the first **for**-loop. Next, as usual, our algorithm must check whether every non-multiplicative prolongation of the form $t_i \cdot x_j$ possesses a Janet divisor already in the set of terms, and whenever it does not, it must include it for completion.

Recall that the lex term ordering $\prec$ is J-ordering (Iglesias and Sáenz-de-Cabezón, 2025, Proposition 2.1); i.e. if the set of terms does contain a Janet divisor t_k of $t_i \cdot x_j$, then $t_i \prec t_k$. Hence, efficiently, Algorithm 7 starts the search from the term t_{i+1} and proceeds in increasing order (see line 16). Moreover, when the algorithm encounters a term t_a that is lexicographically larger than $t_i \cdot x_j$, it escapes the search, with the $flag$ marked as false, indicating that this prolongation needs to be added for completion. This is correct, as no term that is lexicographically larger than any other term can be its Janet divisor, let alone its divisor. Additionally, storing the position of the immediately next element in lexicographical order in the variable a allows the new element to be inserted directly at that position, rather than being pushed to the end of the set and then re-sorted.

Besides, note that because x_j is non-multiplicative for t_i and the Janet divisor $t_k \succ t_i$, then $\deg_j(t_k) = \deg_j(t_i) + 1 = \deg_j(t_i \cdot x_j)$ must hold. Hence, from Theorem 5.8 we know that if in addition for every $\ell < j$, $\deg_\ell(t_k) \leq \deg_\ell(t_i)$, the prolongation then does not need to be added, as $t_i : t_k = x_j$ implies the existence of a Janet divisor already among the terms. Thus, when that is the case, Algorithm 7 marks the $flag$ as true and proceeds to the next non-multiplicative prolongation.

Really, all that remains to be proven regarding the correctness is that when new elements of the form $t_i \cdot x_j$ are added to the set $\mathcal{H}$ in the algorithm, it only affects the non-multiplicative variables of the terms that are positioned exactly between t_i and $t_i \cdot x_j$. This is not only computationally beneficial from the point of view that the adjustment of non-multiplicative variables becomes faster, but it also allows the algorithm to completely insert all the required elements in one go, without having to go back to consider variables that were initially multiplicative, as other iterative algorithms like (Hashemi et al., 2025, Algorithm 1) do. The fact that the terms lex-larger than $t_i \cdot x_j$ are not affected is a direct result of Lemma 7.1. Furthermore, applying Lemma 7.1 once again, if we can show that the set of non-multiplicative variables of t_i remains the same after the addition of any of its non-multiplicative prolongation, then we can claim that all terms lex-smaller than t_i are not affected either. This is clearly the case since x_j was already non-multiplicative and by the definition of the assignment of Janet non-multiplicative variables the rest of them remain the same.

The minimality of the output $\mathcal{H}$ follows from the same argument used in Theorem 5.12. A non-minimal Janet basis contains at least a term t that, when removed from $\mathcal{H}$, there exists a term $t' \in \mathcal{H}$ that divides it involutively. However, due to the specific J-ordering employed and the fact that the minimal generators of $\langle U \rangle$ are computed at the beginning, such a situation cannot occur.

The proof of the complexity bound also follows a similar approach to that of Theorem 5.12. For each newly computed element aux, determining its position and checking its Janet divisor with respect to the current set $\mathcal{H}$ requires $O(n|\mathcal{H}|\log(|\mathcal{H}|))$ operations. Since the updated set $\mathcal{H}$ (after adding $\{aux\}$) remains sorted, adjusting the non-multiplicative variables for each element requires $O(n)$ operations by Lemma 7.1. This adjustment is needed for elements ranging from the i-th to the a-th position (see line 22), and thus the entire operation is performed in $O(n|\mathcal{H}|)$. Finally, since the size of $\mathcal{H}$ varies from k to l, the total complexity follows the claimed bound. □

The lex term ordering is, naturally, also well suited for the Janet-like division. Consequently, Lemma 7.1 can be generalized to enable an efficient computation of all sets of Janet-like non-multiplicative powers. The extension is done in a straightforward manner from the previously used arguments:

Lemma 7.3. *Let $U = \{t_1, \dots, t_k\} \subset \mathcal{P}$ be a lex-ordered set of terms, and let $a_\ell = \deg_\ell(t_{i+1}) - \deg_\ell(t_i)$, where ℓ is the largest subindex such that $\deg_\ell(t_{i+1}) > \deg_\ell(t_i)$ for some $1 \le i < k$. Then,*

$$\mathrm{NMP}_{JL,U}(t_i) = \{x_\ell^{a_\ell}\} \cup \{x_j^{a_j} \mid j > \ell \text{ and } x_j^{a_j} \in \mathrm{NMP}_{JL,U}(t_{i+1})\}.$$

Utilizing now Lemma 7.3 together with the Janet-like version of the arguments used in Theorem 7.2, we can straightforwardly develop an algorithm analogous to Algorithm 7 for the minimal Janet-like completion. To avoid repetitive content, we omit the detailed description of it. In the following sections, we will refer to this particular algorithm as Algorithm 8.

8. Computing Janet bases from Janet-like bases

Recall that, although the Janet-like division is not an involutive division, it preserves all the algorithmic merits of the Janet division (Gerdt and Blinkov, 2005). In fact, this is another indicator of the deep connection between the two divisions. Moreover, using some of the results from the last two sections based on SLB's, we can assert the following:

Proposition 8.1. *Let $\mathcal{J} \subset \mathcal{K}[x_1, \dots, x_n]$ be some monomial ideal, then the minimal Janet-like basis of $\mathcal{J}$ is a subset of the minimal Janet basis of $\mathcal{J}$.*

Proof. The idea here is to show that every element in the minimal Janet-like basis is also an element of the minimal Janet basis. Note that all minimal generators of $\mathcal{J}$ must be in both the minimal Janet-like basis and the minimal Janet basis. Thus, let us assume there exists some element of the minimal Janet-like basis $h_\ell \notin \mathrm{Gen}(\mathcal{J})$. Recall that the minimal Janet-like basis can be obtained using Algorithm 6. Thus, let m be the first mixed term encountered via Algorithm 6 in $\mathbb{F}_j$ and let $h_\ell = h_j \cdot x_i^a$ such that a is the highest integer for which $x_i^a \mid m$ and no $x_t \mid m$ for any $t > i$. Therefore, there exists another element h_k with $k < j$ such that $m = \mathrm{lcm}(h_k, h_j)/h_j$. Besides, there does not exist any other term $m' \in \mathbb{F}_j$ and any integer $b < a$ such that $x_i^b \mid m'$ and x_i is the highest variable appearing in m'. Let us now have a look at the elements that are included during the Janet completion process via Algorithm 4. For this case, m being mixed and $a > 1$ imply that $\deg(m) \ge 2$; hence, it needs to be decomposed. Note that there may be other terms before m in $\mathbb{F}_j$ that can have also degree 2 or more; and thus, that now need to be decomposed before m yielding the addition of new elements to the Janet basis. These terms are pure powers of variables with subindex different from i. However, the insertion of these new elements does not compromise the survivability of the term m, owing to the Janet minimization process employed in this algorithm. When it comes to m, by Algorithm 4 a new element $h_{j'} = h_j \cdot x_i$ needs to be added. Note that $k < j$; hence, $\deg_i(h_k) = \deg_i(h_j) + a$ and $\deg_t(h_k) = \deg_t(h_j)$

$\forall t > i$. Thus, it is clear that $h_k >_{lex} h_{j'} >_{lex} h_j$. This implies that $\mathbb{F}_{j'}$ contains a term m/x_i of degree two or more with $x_i \mid m/x_i$ since $a > 1$, but recall that now there does not exist any other term $m' \in \mathbb{F}_{j'}$ and any integer $b < a-1$ such that $x_i^b \mid m'$ and x_i is the highest variable appearing in m'. Therefore, a new element $h_{j''} = h_{j'} \cdot x_i$ needs to be added. After repeating this step a times we eventually reach the element $h_{j^a} = h_{j^{a-1}} \cdot x_i = \cdots = h_j \cdot x_i^a = h_\ell$, which is also added during the Janet basis monomial completion. This same process can be applied to all the subsequent elements completing the Janet-like basis leading to the conclusion that every element in the minimal Janet-like basis is, in fact, also an element of the minimal Janet basis. □

Note that the minimal set of monomial generators of a given ideal is a subset of the minimal Janet-like basis, which, as we have just seen, is itself a subset of the minimal Janet basis. Furthermore, one can quickly deduce from the description of Algorithm 4, combined with Lemma 5.2, that if any subset of the minimal Janet basis containing all the minimal generators of a given ideal is used as input for Algorithm 4, then even if the minimization of the inputted set is omitted (first line of the Algorithm), the output will still be the minimal Janet basis of that ideal. This process of completing a Janet-like basis to a Janet basis is what we coined as *filling in the gaps* process or *FTG* for short. Moreover, during this process, for any given element h_j of the basis, we call the *children of* h_j the elements that are added during the while loop that decomposes the non-linear surviving terms in $\mathbb{F}_j$.

Lemma 8.2. *Let $\mathcal{J}$ be a monomial ideal, $\mathcal{H}$ the minimal Janet-like basis of $\mathcal{J}$ and $\mathcal{H}'$ the minimal Janet basis of $\mathcal{J}$. We have that* $\mathrm{NM}_J(h_j, \mathcal{H}') = \{x_i \mid x_i^a \in \mathrm{NMP}(h_j, \mathcal{H})\}$ *for any generator $h_j \in \mathcal{H}'$.*

Proof. This is a direct result of the decomposition of non-linear terms in Algorithm 4 during the filling in the gaps process. Following Lemma 5.4 and the order used in Algorithm 4, it is straightforward that the children added to complete $\mathcal{H}$ into $\mathcal{H}'$ affect the pure powers of variables, turning them into just variables for every set $\mathbb{F}_i$. All that is left to show is that, if done in this particular order, these sets do not change as future children of other elements are added. In other words, new terms cannot appear in the $\mathbb{F}_i$'s that have already been linearized, even if new elements are placed in front. Let us start with the minimal Janet-like basis $\mathcal{H}$, following Algorithm 4, after the first round of decomposition of non-linear terms all children of the smallest lexicographical element are added. Let $\{h_1, \ldots, h_k\}$ be the lexicographically ordered SLB composed of the elements in $\mathcal{H}$ plus the children of the last element. Clearly, $\mathbb{F}_k$ now comprises only variables (following, of course, the Janet minimization process). We must then show that $\mathbb{F}_k = \mathrm{NM}_J(h_k, \mathcal{H}')$ no matter what the rest of the $\mathbb{F}_i$'s look like. To get a contradiction, let us assume there exists $x_t \in \mathrm{NM}_J(h_k, \mathcal{H}')$ such that $x_t \notin \mathbb{F}_k$. The only way this can still be true is if some child, h', with $\deg_t(h') > \deg_t(h_k)$ and $\deg_i(h') \leq \deg_i(h_k)$ for all $i > t$ where $x_t \notin \mathbb{F}_k$, is added during the Janet completion. Note that this possible child h' can only affect the terms in $\mathbb{F}_k$ if $h' >_{lex} h_k$; thus, $\deg_i(h') = \deg_i(h_k)$ for all $i > t$. We assumed that $x_t \notin \mathbb{F}_k$, which means that $\deg_t(h_k) = \max\{\deg_t(h_s) \mid h_s \in \{h_1, \ldots, h_k\}$ and $\deg_i(h_s) = \deg_i(h_k)$ for all $i > t\}$. Therefore, this leaves us with two possible cases:

- Case I: There exists $h_\ell \in \{h_1, \ldots, h_k\}$ with $\deg_t(h_\ell) > \deg_t(h_k)$ and $\deg_i(h_\ell) = \deg_i(h_k)$ for every $i > t$ except at one index $r > t$ where $\deg_r(h_\ell) = \deg_r(h_k) - 1$ in which case a possible child $h' = h_\ell \cdot x_r$ will destroy our claim. However, h_k was the lexicographically smallest element, so this case is not possible.

- Case II: There exists $h_\ell \in \{h_1, \ldots, h_k\}$ with $\deg_t(h_\ell) > \deg_t(h_k)$ and $\deg_i(h_\ell) \leq \deg_i(h_k)$ for all $i > t$ for which one of his children $h' = h_\ell \cdot x_t$. Again, this cannot happen because either $h_\ell \prec h_k$ or else there exists $m \in \mathbb{F}_\ell$ such that x_t is the variable with largest subindex that divides m but then this would imply $x_t \in \mathbb{F}_k$, and hence, our contradiction. Note that all the subsequent $\mathbb{F}_i$'s, i.e. all sets with index $i < k$, are not affected by the element h_k nor by any other lexicographically smaller element. Thus, we can now consider the subset $\{h_1, \ldots, h_{k-1}\}$. We then add all children of h_{k-1} and repeat the same process, obtaining the same outcome.

□

Lemma 8.3. *Let $\mathcal{H}$ be the minimal Janet basis of $\mathcal{J}$ obtained through the filling in the gaps process. All children of h_j in $\mathcal{H}$ inherit the multiplicative and non-multiplicative variables from h_j.*

Proof. Let us start with the minimal Janet-like basis of $\mathcal{J}$. When viewed as an staggered linear basis, we observe that $\mathbb{F}_j = \{x_i^{a_i} \mid i \in \{1, \ldots, n\}\}$ for every element h_j in the basis (Theorem 6.5). We also know that if there exists $a_k > 1$ such that $x_k^{a_k} \in \mathbb{F}_j$, then there exists a child of h_j of the form $h_j \cdot x_k$ in $\mathcal{H}$. Let us denote this element as $h_{j'}$; then it is clear that if we add this child to the minimal Janet basis, then all $x_i^{a_i} \in \mathbb{F}_j$ with $i > k$ will also belong to $\mathbb{F}_{j'}$; and hence, by Lemma 8.2, all such x_i's will be non-multiplicative for both h_j and $h_{j'}$ in $\mathcal{H}$. Thus, it just remains to show that $\mathrm{NM}_J(h_{j'}, \mathcal{H}) \cap \{x_1, \ldots, x_k\} = \{x_i \mid x_i^{a_i} \in \mathbb{F}_j \text{ and } i \leq k\}$. Note that $a_k > 1$, thus $x_k^{a_k - 1} \in \mathbb{F}_{j'}$. Also, from the minimal Janet-like basis when viewed as an SLB, we observe that for any $x_i^{a_i} \in \mathbb{F}_j$ with $i < k$, there exists h_ℓ such that $\deg_i(h_\ell) = \deg_i(h_j) + a_i$ and $\deg_t(h_\ell) = \deg_t(h_j)$ for every $t > i$. Thus, $h_{j'} \succ_{lex} h_\ell \succ_{lex} h_j$, and hence, h_ℓ will have a child $h_{\ell'} = h_\ell \cdot x_k$ in front of $h_{j'}$, i.e. $\ell' < j'$, which implies that each of these x_i's will also be non-multiplicative for $h_{j'}$ in $\mathcal{H}$ as desired. All that is left to check is that no child can possess a non-multiplicative variable that is multiplicative for the parent (with respect to the minimal Janet basis $\mathcal{H}$). This can be easily shown by reductio ad absurdum, so, let us assume such a variable x_i exists for an element $h_{j'}$ that is a child of h_j, i.e. $h_{j'} = h_j \cdot x_q$ for some $x_q \in \mathrm{NM}_J(h_j, \mathcal{H})$. Then, because $x_i \in \mathrm{NM}_J(h_{j'}, \mathcal{H})$, by Theorem 5.8, it implies that there exists an element $h_k \in \mathcal{H}$ such that $\deg_i(h_k) = \deg_i(h_{j'}) + 1$ and $\deg_t(h_k) = \deg_t(h_{j'}) = \deg_t(h_j)$ for every $t > i$. Note that if $i \geq q$ then that would automatically imply that $x_i \in \mathrm{NM}_J(h_j, \mathcal{H})$, thus, a contradiction to our assumption. Otherwise, assume $i < q$: then again by Theorem 5.8, there exists some $h_k \in \mathcal{H}$ such that:

$$\deg_\ell(h_{j'}) \begin{cases} = \deg_\ell(h_k) - 1 & = \deg_\ell(h_j) \qquad , \text{when } \ell = i \\ = \deg_\ell(h_k) & = \deg_\ell(h_j) \qquad , \text{when } i < \ell < q \\ = \deg_\ell(h_k) & = \deg_\ell(h_j) + 1 \quad , \text{when } \ell = q \\ = \deg_\ell(h_k) & = \deg_\ell(h_j) \qquad , \text{when } \ell > q. \end{cases}$$

Note that this implies that $h_k \succ h_j$; and hence, x_q is the variable with largest subindex that divides $m = \mathrm{lcm}(h_k, h_j)/h_j$, while $x_q^2 \nmid m$. Thus, we can conclude that in this scenario, either $m = x_q$, which implies that $\mathcal{H}$ is not the minimal Janet basis of $\mathcal{J}$ since $h_{j'}$ would become redundant, or m is mixed. In the latter case, the decomposition of m should have happened during the Janet-like completion; hence, $h_{j'}$ is not a child of any element but it is instead an element of the Janet-like basis, which provides the last contradiction that completes the proof. □

By combining these last two lemmata, we obtain as a direct result an explicit method to describe the entire minimal Janet basis in terms of the elements of the minimal Janet-like basis and the non-multiplicative powers of each element:

Theorem 8.4. *Let $\mathcal{H} = \{h_1, \dots, h_s\}$ be the minimal Janet-like basis of a monomial ideal $\mathcal{J} \subset \mathcal{P}$. For each $h_i \in \mathcal{H}$, let $\mathcal{T}_i$ be the finite set of terms in the quotient ring $\mathcal{K}[x_j \mid x_j^{a_j} \in \mathrm{NMP}(h_i, \mathcal{H})]/\langle \mathrm{NMP}(h_i, \mathcal{H})\rangle$. Then, we have that*

$$\mathcal{H}' = \{h_i \cdot t \mid h_i \in \mathcal{H} \text{ and } t \in \mathcal{T}_i\}$$

is the minimal Janet basis of $\mathcal{J}$, and furthermore,

$$\mid \mathcal{H}' \mid = \sum_{i=1}^{s} \prod_{x_j^{a_j} \in \mathrm{NMP}(h_i, \mathcal{H})} a_j$$

It also is worth mentioning that it is already known from (Hashemi et al., 2023, Proposition 6.10) how to obtain a Janet basis from a Janet-like basis, and even how to read off the whole Janet complementary decomposition from the minimal Janet-like basis (Hashemi et al., 2022, Proposition 24). Here, we present an alternative approach to this process. Furthermore, since Lemmata 8.2 and 8.3 automatically provide the non-multiplicative variables for all elements of the minimal Janet basis, a significant portion of the computational effort is saved. In conclusion, by combining these lemmata with the last theorem, we are basically claiming that all the information about the minimal Janet basis is encoded inside the minimal Janet-like basis. Thus, we provide a notably more efficient method to compute the minimal Janet basis, as obtaining the Janet-like basis is generally far less computationally expensive. This efficiency will be demonstrated in the following section.

9. Implementation and performance comparison of algorithms

Our final section is devoted to the implementation of the algorithms presented throughout the paper. We first compare the efficiency of the proposed algorithms for computing minimal Janet bases. Second, we compare the efficiency of algorithms that compute the minimal Janet-like bases. And lastly, we compare the efficiency of directly computing the minimal Janet basis versus computing the Janet-like basis using the fastest algorithm and then completing it to the Janet basis using the FTG process, as described in the immediately preceding section.

For our experiments, we consider various random monomial ideals. There are many different strategies to produce randomness when dealing with monomial ideals, see for instance the approach in (De Loera et al., 2019). Our simple approach is as follows: Let

n, g and d be three positive integers. We build g monomials in n variables by randomly choosing an exponent between 0 and d for each of the variables and multiplying these powers of variables. Since there might be divisibility relations among the monomials produced in this way, we keep on adding generators until we reach a number of g mutually non-divisible monomials. We then consider the ideal minimally generated by these g monomials. In order to avoid extreme cases and unwanted variations in the execution times, for each choice of n, g and d we generate ten monomial ideals, and run each algorithm five times on each of these ideals, taking then the average time of these five runs to get a more accurate measure of the computing time. Technically, n represents the number of variables, d the maximum degree, and g the number of generators. However, in the tables, only the parameter n is shown since, for simplicity, we always set $n = d = g$ for all our examples. We do so because then the variation in the sizes of the ideals and their corresponding Janet bases allows us to observe how the computation times vary. For each value of $n = d = g$ we compute and write in the tables the average value of the computing time in the ten ideals generated for that value and taken from the average of the five runs for each ideal. In Tables 1, 2 and 3 column `Size JB` and `Size JLB` indicate the average size of the corresponding minimal Janet and Janet-Like bases respectively.

Our algorithms are implemented in the `C++` library `CoCoALib` ((Abbott and Bigatti)). All experiments were run on an `Apple M3 Pro` processor with 6 Performance Cores (up to 4.06 GHz) and 6 Efficiency Cores (2.8 GHz) and 18 GB RAM, running under `MacOsX` operating system. All times in the tables are given in seconds, and the best time for each case is written in boldface. In all tables, `OOT` stands for *Out Of Time*, indicating that the execution of the algorithms was stopped after two hours, and `OOM` stands for *Out Of Memory*, meaning the system ran out of RAM during the execution of the algorithms.

9.1. Computation of Janet bases

For the direct computation of minimal Janet bases, we introduced a recursive Algorithm 1. This algorithm presents a more efficient approach to recursive minimal completion compared to existing algorithms, such as Hashemi et al. (2023, Algorithm 3), and will therefore serve as the representative for recursive algorithms. The computing times for this algorithm are given in the `J Recursive` column of Table 1. On the other hand, Algorithm 4 computes minimal Janet bases using a different approach, primarily based on SLB's. The times for this algorithm are given in the `J SLB` column of Table 1. Lastly, we also introduced Algorithm 7, an innovative iterative approach which uses the lexicographic order of terms to its advantage to achieve minimal completion more efficiently than some existing algorithms, such as (Hashemi et al., 2025, Algorithm 1). In our testing, Algorithm 7 proved to be significantly more efficient than the other iterative algorithms; thus, it is selected to be the representative for iterative algorithms.

The actual implementation of these algorithms varies slightly from the schematic descriptions, as we incorporated some optimizations beyond what was outlined in the previous sections. For the implementation of the recursive Algorithm 1, the differences are minimal, limited to minor details such as strategic reordering of monomials to avoid potentially redundant computations and to improve computational efficiency. On the other hand, more significant changes were implemented in the SLB Algorithm 4. Specifically, instead of computing all colon operations with lexicographically larger elements for each basis element and then applying the proper Janet minimization process (as described in Section 5), we optimize the computation by leveraging the properties of the Janet minimization process as well as the fact that the basis is kept lexicographically ordered throughout the whole completion process. Hence, in this improved approach, colon operations are computed one by one, and those operations yielding terms that would have been eliminated during the minimization process are now skipped. At the same time, we track the position within the process to ensure that the new elements that will complete the basis are inserted in the correct place, avoiding the need for any reordering of the elements. Additionally, we introduced a break point that halts the search for significant colon operations once the last variable appears in the $\mathbb{F}_i$ sets, saving considerable computational time by avoiding irrelevant calculations. For the actual implementation of the iterative Algorithm 7 some minor details are to be noted. The most relevant one is that the adjustment of the sets of non-multiplicative variables after a prolongation $t_i \cdot x_j$ is added is actually applied to an even smaller set of terms, the ones in positions between the prior prolongation and the newly added one. Furthermore, this adjustment is done in an optimized manner, as only the variables with subindex lower than j can actually be affected.

n	`Size JB`	`J General`	`J Recursive`	`J SLB`	`J Iterative`
6	193.8	0.01159	0.00036	0.00037	**0.00012**
7	1180.5	0.09350	0.00156	0.00270	**0.00068**
8	5465.2	1.05535	0.00779	0.01831	**0.00410**
9	38854.8	29.28535	0.06880	0.21304	**0.04236**
10	140924.6	167.92882	0.26584	0.94272	**0.17657**
11	1175672.3	`OOT`	2.67588	11.37893	**1.96186**
12	5273657.1	`OOT`	12.76148	65.01852	**11.14658**
13	21563782.0	`OOT`	58.37471	333.61277	**56.80410**

Table 1: Comparison between `C++` implementations of Janet basis computation

The table includes another column, `J General` that shows the times obtained from the `CoCoALib` implementation of the Janet-basis algorithm described in (Albert et al., 2015). This comparison is not entirely fair, as the algorithm in question is designed for general polynomial ideals, whereas ours are specialized for monomial ideals. However, the algorithm allows for the selection of one of four different strategies. After some testing on the four strategies, one of them proved to be significantly more efficient than the others for the monomial ideals we generate. Therefore, we include this algorithm with that specific strategy in the comparison as a benchmark representing the previous state-of-the-art implementations of Janet bases algorithms. The results in Table 1 demonstrate how much one can gain by utilizing a specialized algorithm for the monomial case. For small values of n, the iterative algorithm clearly performs best However, with increasing n the recursive algorithm catches up and is finally at almost the same level as the iterative one. The SLB algorithm is significantly slower.

9.2. Computation of Janet-like bases

For the computation of minimal Janet-like bases, we will also consider one recursive algorithm, one algorithm based on SLB's and one iterative algorithm (Algorithms 2, 6 and 8). As before, we use optimization on the actual implementation, which then differs slightly from the description given in previous sections. These optimizations are nearly identical to those described in the preceding subsection, but they are now adapted to the Janet-like counterpart.

As the following table shows, both the recursive Algorithm 2 and the iterative Algorithm 8 seem to be our most efficient choice again. The computing times for this algorithm are reported in the `J-L Recursive` column and `J-L Iterative` column of Table 2, respectively. For random examples with "smaller" bases, the iterative algorithm again outperforms the recursive one. But as for Janet bases, the recursive algorithm catches up as n increases. Since here it is possible to proceed to larger values of n, at the end the recursive algorithm even overtakes and becomes the fastest one. The SLB algorithm is again not competitive compared to these two.

n	`Size JLB`	`J-L Recursive`	`J-L SLB`	`J-L Iterative`
6	106.6	0.00012	0.00008	**0.00003**
7	361.2	0.00029	0.00030	**0.00010**
8	1112.8	0.00095	0.00138	**0.00040**
9	3636.6	0.00377	0.00677	**0.00180**
10	9993.2	0.01137	0.02550	**0.00612**
11	31962.4	0.04245	0.11229	**0.02438**
12	97640.4	0.13468	0.43299	**0.09495**
13	266318.6	0.40172	1.46883	**0.31463**
14	1575265.8	2.48607	11.54700	**2.29572**
15	4625245.1	8.26236	41.58812	**8.10513**
16	12822586.0	22.61042	124.85532	**20.73170**
17	53812460.0	**102.29907**	774.50050	106.93616
18	166906166.6	**396.76583**	2567.41633	414.66800

Table 2: Comparison between `C++` implementations of Janet-like basis computation

9.3. Direct Janet basis computation vs. Janet-like computation plus FTG completion

The proposed algorithms allow us to compute the minimal Janet basis of a monomial ideal in two ways. One method is through direct computation, either using Algorithm 1, Algorithm 4 or Algorithm 7, and the other method involves first computing a Janet-like basis (for which we use the implementation of Algorithm 2 or Algorithm 8), which is usually a much faster computation, and then completing this Janet-like basis into the full Janet basis via the FTG process, as described in Section 8. Table 3 shows the time comparison between these methodologies using our algorithms. Column `Janet` shows the time taken by the best Janet basis algorithm, column `Janet-like` shows the time of the best Janet-like basis algorithm, column `FTG` shows the time of the FTG completion from the Janet-like basis to the Janet basis (added up to the time taken to compute the Janet-like basis). Columns `size JB` and `size JLB` show the size of the corresponding Janet and Janet-like bases, respectively.

n	`size JB`	`size JLB`	`Janet`	`Janet-like`	`FTG`
6	193.8	106.6	0.00012	0.00003	0.00006
7	1180.5	361.2	0.00068	0.00010	0.00025
8	5465.2	1112.8	0.00410	0.00040	0.00110
9	38854.8	3636.6	0.04236	0.00180	0.00772
10	140924.6	9993.2	0.17657	0.00612	0.02953
11	1175672.3	31962.4	1.96186	0.02438	0.31014
12	5273657.1	97640.4	11.14658	0.09495	1.32594
13	21563782.0	266318.6	56.80410	0.31463	8.53521
14	`OOM`	1575265.8	`OOM`	2.29572	`OOM`
15	`OOM`	4625245.1	`OOM`	8.10513	`OOM`

Table 3: Comparison between direct Janet and Janet-like strategies for Janet basis computation

The results in Table 3 indicate that the most efficient way to compute Janet bases is to actually compute a Janet-like basis first and then complete it to a Janet basis using the FTG process. Comparing the times taken by this approach with those of the current `CoCoALib` implementation of Janet bases (shown in the `J General` column of Table 1), we observe an extremely large performance gap. However, it is worth pointing out that for almost any purpose this filling is not necessary, as the Janet-like basis provides the same information as the Janet basis.

9.4. Dependency of the computing time on number of variables and size of the basis

The data on the tables in the previous sections demonstrate that the size of the bases dominates the computation times, in the sense that on both Janet and Janet-like bases computations, there is a linear relation between the size of the computed object and the time required to compute it, while the dependency is exponential with respect to the number of variables. This can be seen in Figure 1 where the results for all the examples that we ran are shown. The graphics in the top row show (using a log scale on the vertical axis) the exponential dependency of the computing time on the number of variables. The bottom row of the figure shows (on a double logarithmic scale) the relation between the size of the Janet and Janet-like basis and its computing time. Here we see a linear relation. These results are compatible with the general complexity theory of Gröbner bases (Mayr, 1997), which can be even double-exponential in the worst case.

Finally, the graph in Figure 2 (using a double logarithmic scale) suggests a linear relation between the size of the Janet and the Janet-like bases in our examples. This ratio could be dependent on the type of examples used for our computation, but it poses an interesting question on how to estimate the ratio between the size of Janet and Janet-like bases in general. This is considered as a future line of research and exceeds the scope of the present paper.

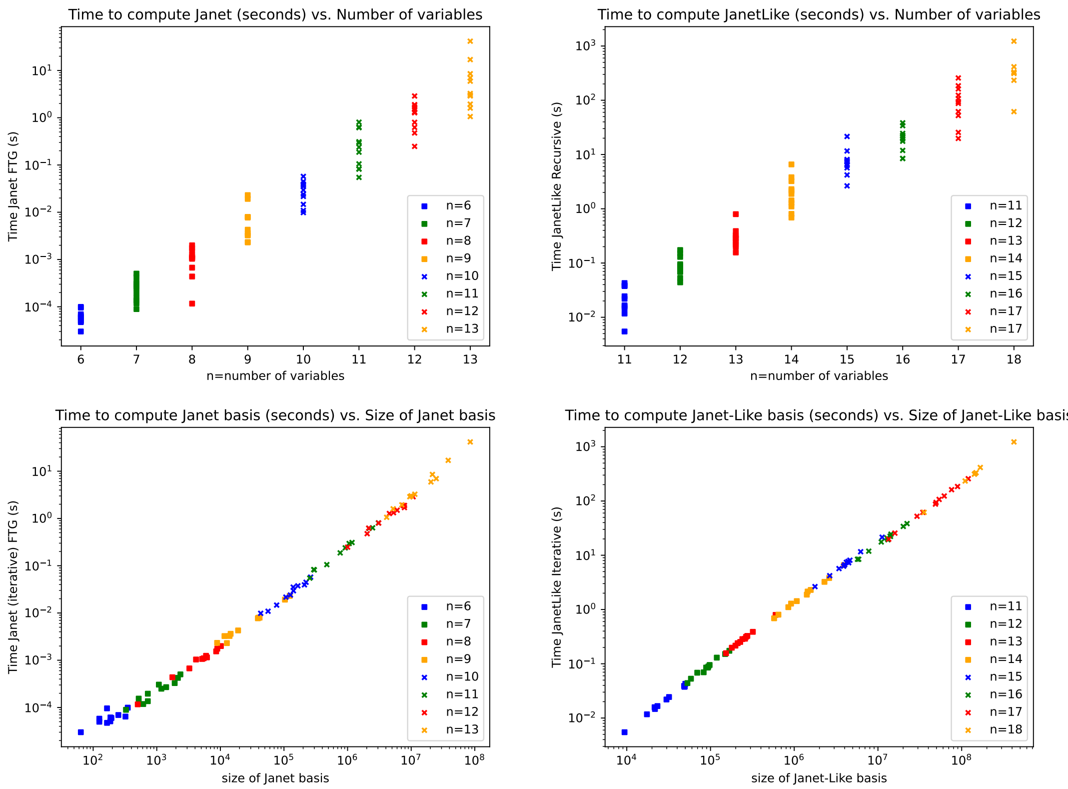

Figure 1: Time versus number of variables and time versus size when computing Janet and Janet-like bases in the examples of Table 3

Acknowledgements

The research of the first author was in part supported by a grant from IPM (No. 1404130412).

References

Abbott, J., Bigatti, A.M., . CoCoALib: a C++ library for doing Computations in Commutative Algebra. Available at `http://cocoa.dima.unige.it/cocoalib`.

Adams, W.W., Loustaunau, P., 1994. An introduction to Gröbner bases. volume 3. Providence, RI: American Mathematical Society.

Albert, M., Fetzer, M., Seiler, W.M., 2015. Janet bases and resolutions in CoCoALib, in: Gerdt, V.P., Koepf, W., Seiler, W.M., Vorozhtsov, E.V. (Eds.), Computer Algebra in Scientific Computing, Springer International Publishing, Cham. pp. 15–29.

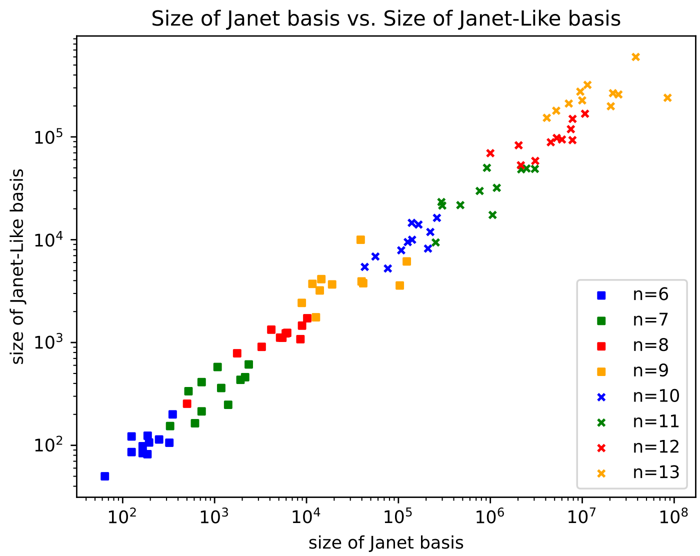

Figure 2: Size of Janet bases versus size of Janet-Like bases for the examples in Table 3

Blinkov, Y.A., Cid, C.F., Gerdt, V.P., Plesken, W., Robertz, D., 2003. The MAPLE package Janet: I. Polynomial systems., in: Computer algebra in scientific computing. 6th international workshop, CASC 2003, Passau, Germany, September 20–26, 2003. Proceedings. Berlin: Springer, pp. 31–40.

Ceria, M., 2022. Applications of bar code to involutive divisions and a "greedy" algorithm for complete sets. Math. Comput. Sci. 16, 17. Id/No 30.

Ceria, M., 2022. Bar code and Janet-like division. Atti Accad. Peloritana dei Pericolanti 100, A2.

Cox, D.A., Little, J., O'Shea, D., 2015. Ideals, varieties, and algorithms. An introduction to computational algebraic geometry and commutative algebra. 4th revised ed. Cham: Springer.

De Loera, J.A., Petrović, S., Silverstein, L., Stasi, D., Wilburne, D., 2019. Random monomial ideals. Journal of Algebra 519, 440–473.

Gebauer, R., Möller, H.M., 1986. Buchberger's algorithm and staggered linear bases, in: Proceedings of SYMSAC'86, ACM, New York, NY, USA. pp. 218–221.

Gerdt, V.P., 2005. Involutive algorithms for computing Gröbner bases., in: Computational commutative and non-commutative algebraic geometry. Proceedings of the NATO Advanced Research Workshop, 2004. Amsterdam: IOS Press, pp. 199–225.

Gerdt, V.P., Blinkov, Y.A., 1998. Involutive bases of polynomial ideals. Math. Comput. Simul. 45, 519–541.

Gerdt, V.P., Blinkov, Y.A., 1998. Minimal involutive bases. Math. Comput. Simul. 45, 543–560.

Gerdt, V.P., Blinkov, Y.A., 2005. Janet-like monomial division., in: Computer algebra in scientific computing. 8th international workshop, CASC 2005, Kalamata, Greece, September 12–16, 2005. Proceedings. Berlin: Springer, pp. 174–183.

Gerdt, V.P., Blinkov, Y.A., Yanovich, D.A., 2001. Construction of Janet bases. I: Monomial bases, in: Computer algebra in scientific computing, CASC 2001. Proceedings of the 4th international workshop, Konstanz, Germany, September 22–26, 2001. Berlin: Springer, pp. 233–247.

Hashemi, A., Mirhashemi, M., Seiler, W.M., 2025. Machine learning parameter systems, noether normalisations and quasi-stable positions. Journal of Symbolic Computation 126, 102345.

Hashemi, A., Möller, H.M., 2023. A new algorithm for computing staggered linear bases. J. Symb. Comput. 117, 1–14.

Hashemi, A., Orth, M., Seiler, W.M., 2022. Complementary decompositions of monomial ideals and involutive bases. Appl. Algebra Eng. Commun. Comput. 33, 791–821.

Hashemi, A., Orth, M., Seiler, W.M., 2023. Recursive structures in involutive bases theory. J. Symb. Comput. 118, 32–68.

Iglesias, R., Sáenz-de-Cabezón, E., 2025. Involutively stable ideals and componentwise linearity. Preprint in preparation .

Janet, M., 1920. Sur les systèmes d'équations aux dérivées partielles. C. R. Acad. Sci., Paris 170, 1101–1103.

Janet, M., 1929. Leçons sur les Systèmes d'Équations aux Dérivées Partielles. Cahiers Scientifiques, Fascicule IV, Gauthier-Villars, Paris.

Mayr, E.W., 1997. Some complexity results for polynomial ideals. Journal of Complexity 13, 303–325.

Möller, H.M., Mora, T., Traverso, C., 1992. Gröbner bases computation using syzygies., in: International symposium on Symbolic and algebraic computation 92. ISSAC 92. Berkeley, CA, USA, July 27–29, 1992. Baltimore, MD: ACM Press, pp. 320–328.

Mora, T., 2005. Solving polynomial equation systems. II. Macaulay's paradigm and Gröbner technology. volume 99 of *Encycl. Math. Appl.* Cambridge: Cambridge University Press.

Mora, T., 2016. Solving polynomial equation systems. Vol. IV. Buchberger theory and beyond. volume 158. Cambridge: Cambridge University Press.

Seiler, W.M., 2010. Involution — The formal theory of differential equations and its applications in computer algebra. Berlin: Springer.

Zharkov, A., Blinkov, Y., 1996. Involution approach to investigating polynomial systems. Math. Comput. Simul. 42, 323–332.