.

The Closedness Subspace Method for Computing the Multiplicity Structure of a Polynomial System

ZHONGGANG ZENG

Abstract. The multiplicity structure of a polynomial system at an isolated zero is identified with the dual space consisting of differential functionals vanishing on the entire polynomial ideal. Algorithms have been developed for computing dual spaces as the kernels of Macaulay matrices. These previous algorithms face a formidable obstacle in handling Macaulay matrices whose dimensions grow rapidly when the problem size and the order of the differential functionals increase. This paper presents a new algorithm based on the closedness subspace strategy that substantially reduces the matrix sizes in computing the dual spaces, enabling the computation of multiplicity structures for large problems. Comparisons of timings and memory requirements demonstrate a substantial improvement in computational efficiency.

1. Introduction

This paper presents what we call the closedness subspace method for computing the multiplicity structure of a polynomial system

$$f_1(x_1, \cdots, x_s) = \cdots = f_t(x_1, \cdots, x_s) = 0 \tag{1.1}$$

at an isolated zero $\hat{\mathbf{x}} = (\hat{x}_1, \cdots, \hat{x}_s)$ based on the Macaulay-Gröbner duality formulation of the multiplicity.

Formulation, analysis and calculation of the multiplicity for the polynomial system (1.1) at a zero are fundamental problems in algebraic geometry [**6, 10, 13, 15**]. There are rich structural invariants at $\hat{\mathbf{x}}$ besides the multiplicity, such as the dual space, depth, breadth [**5**], and regularity [**4**]. Those invariants not only dictate the difficulty in solving the system (1.1) for multiple zeros [**5, 8**], but also determine the nature of the ideal $\langle f_1, \ldots, f_t \rangle$ itself [**11**].

Computational algorithms have been in development for identifying the multiplicity and various invariants in symbolic, numerical and hybrid computations such as [**7, 12, 14, 17**] and more recently in [**4, 5, 18**]. The most efficient approach appears to be identifying the dual spaces through the kernels of the Macaulay matrices [**5, 10**]. As mentioned in [**15**, p. 334], the first implementation of a dual space computing algorithm is carried out in [**17**] but the code is no longer available. The algorithm in [**5**] is implemented with both symbolic and numerical options and included in the software package `ApaTools` [**19, 20**]. A multiplicity algorithm is also to be imbedded in software packages such as `Bertini` [**1, 2, 3**]. However, difficulty may rise to a prohibitive level when the multiplicity is high and the number of variables increases. For example, consider the KSS system [**5, 7**]

$$f_j(x_1, \cdots, x_s) \;=\; (x_1 + \cdots + x_s) + x_j^2 - 2x_j - n + 1, \text{ for } j = 1, \ldots, n \tag{1.2}$$

2000 *Mathematics Subject Classification.* 65H10,68W25,68W30,14Q99,74G35,13P10.
This research was supported in part by NSF under Grant DMS-412003 and DMS-0715127.

which we shall adapt as one of the benchmark problem sequences for multiplicity computation. The largest Macaulay matrix required at the zero $\hat{\mathbf{x}} = (1, \ldots, 1)$ is $12,012 \times 3,432$ for $n = 7$ to obtain the multiplicity 64, grows to $102,960 \times 24,310$ for $n = 8$ with multiplicity 163, and reaches $218,790 \times 48,620$ for $n = 9$ and multiplicity 256, far exceeding the memory capacity of today's desktop computers. Even if the problem is within the memory capacity, the matrix size becomes the main bottleneck that slows down the overall computation.

In this paper we develop a *closedness subspace method* with an objective to improve the efficiency substantially in both storage space and execution time for computing the multiplicity structure. The main strategy is to take advantage of the closedness condition as formulated in [**11, 15**]. (c.f. Lemma 3.1 in §3). This closedness condition leads to a sequence of low-dimension closedness subspaces that contain the dual subspaces, and yields much smaller sizes of matrices from which the dual space is to be computed. For instance, the largest matrix is of column dimension 1304 for computing the multiplicity of the KSS system (1.2) for $n = 8$, and 2295 for $n = 9$, a small fraction of the Macaulay matrix sizes.

The experimental results show a remarkable improvement in computing efficiency. Applying a sequence of specifically designed matrix computation schemes, the implementation of this new method is consistently faster than its predecessor in `ApaTools` [**19, 20**] on benchmark systems requiring at least one second to solve. On larger problems in sequences of systems such as the KSS in (1.2), the ratio of improvement in computing time increases from 7-fold for $n = 4$ to 15-fold for $n = 5$, and to 177-fold for $n = 6$. More importantly, the new method continues to produce multiplicity structures when the system size increases to $n = 7$, 8, and 9, long after the previous method runs out of memory. On large systems of low multiplicities, the new method can be over 3500 times faster (cf. Table 4).

We shall outline the basic theory of the Macaulay-Gröbner formulation and the previous method for computing the multiplicity structure using the Macaulay matrices (§2). The proposed closedness subspace method will be presented in detail (§3–§6) along with the computing strategy described in §5 to take advantage of the block matrix structures. The computational experiments, results, and comparisons will be given in §7.

2. Preliminaries

The n dimensional complex vector space is denoted by $\mathbb{C}^n$, and the ring of polynomials in indeterminates $x_1, \ldots, x_s$ with coefficients in $\mathbb{C}$ is denoted by $\mathbb{C}[x_1, \ldots, x_s]$. A polynomial is denoted by a lower case letter, say p or f_j, etc. Let $\mathbb{N}^s$ denote the set of s dimensional index arrays, where each index is a nonnegative integer. Corresponding to an index array $\mathbf{j} = [j_1, \cdots, j_s] \in \mathbb{N}^s$, the (algebraic) monomials $\mathbf{x}^{\mathbf{j}} = x_1^{j_1} \cdots x_s^{j_s}$ and $(\mathbf{x} - \mathbf{y})^{\mathbf{j}} = (x_1 - y_1)^{j_1} \cdots (x_s - y_s)^{j_s}$, whereas the differentiation monomial

$$\partial_{\mathbf{j}} \equiv \partial_{j_1 \cdots j_s} \equiv \frac{1}{j_1! \cdots j_s!} \frac{\partial^{j_1 + \cdots + j_s}}{\partial x_1^{j_1} \cdots \partial x_s^{j_s}}, \tag{2.1}$$

with the order $|\mathbf{j}| \equiv j_1 + \cdots + j_s$.

Consider a system of t polynomials $\{f_1, \cdots, f_t\}$ in s variables with an isolated zero $\hat{\mathbf{x}}$ where $t \geq s$. Polynomials $f_1, \cdots, f_t \in \mathbb{C}[x_1, \ldots, x_s]$ generate an ideal $\mathcal{I} = \langle f_1, \cdots, f_t \rangle$. We define a (differential) monomial functional

$$\partial_{\mathbf{j}}[\hat{\mathbf{x}}] : \mathbb{C}[x_1, \ldots, x_s] \longrightarrow \mathbb{C}, \quad \text{where } \partial_{\mathbf{j}}[\hat{\mathbf{x}}](p) = (\partial_{\mathbf{j}} p)(\hat{\mathbf{x}}) \text{ for } p \in \mathbb{C}[\mathbf{x}]$$

at $\hat{\mathbf{x}} \in \mathbb{C}^n$. Generally, a (*differential*) *functional* at $\hat{\mathbf{x}} \in \mathbb{C}^s$ is a linear combination of those $\partial_{\mathbf{j}}[\hat{\mathbf{x}}]$'s. All functionals at the zero $\hat{\mathbf{x}}$ that vanish on $\mathcal{I}$ form a vector space $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ called the *dual space* of $\mathcal{I}$ at $\hat{\mathbf{x}}$

$$\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I}) \equiv \left\{ c = \textstyle\sum_{\mathbf{j} \in \mathbb{N}^s} c_{\mathbf{j}} \partial_{\mathbf{j}}[\hat{\mathbf{x}}] \;\middle|\; c(f) = 0, \text{ for all } f \in \mathcal{I} \right\} \tag{2.2}$$

where $c_{\mathbf{j}} \in \mathbb{C}$ for all $\mathbf{j} \in \mathbb{N}^s$. We may write $\partial_{\mathbf{j}}$ instead of $\partial_{\mathbf{j}}[\hat{\mathbf{x}}]$ for simplicity if the zero $\hat{\mathbf{x}}$ is clear from the context.

DEFINITION 2.1. [**15**] The multiplicity of an ideal $\mathcal{I} \subset \mathbb{C}[\mathbf{x}]$ at an isolated zero $\hat{\mathbf{x}}$ is m if the dual space $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ is of dimension m, while $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ itself defines the multiplicity structure of $\mathcal{I}$ at $\hat{\mathbf{x}}$.

This definition generalizes the multiplicity from the univariate case, where $\hat{x}$ is an m-fold root of $p \in \mathbb{C}[x]$ if $p(\hat{x}) = p'(\hat{x}) = \cdots = p^{(m-1)}(\hat{x}) = 0 \neq p^{(m)}(\hat{x})$. Namely, $\mathcal{D}_{\hat{x}}(\mathcal{I})$ is spanned by $\partial_0[\hat{x}], \partial_1[\hat{x}], \ldots, \partial_{m\text{-}1}[\hat{x}]$. This formulation of multiplicity, also referred to as the "arithmetical multiplicity" [**12**], is a classical approach that can be traced back to works of Lasker, Macaulay and Gröbner in early 1900s.

The dual space $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ consists of functionals which vanish on the *entire* ideal $\mathcal{I} = \langle f_1, \ldots, f_t \rangle$. In other words, let c be a differential functional at $\hat{\mathbf{x}}$. Then $c \in \mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ if and only if

$$c(p \cdot f_i) \;=\; 0 \quad \text{for all } \; p \in \mathbb{C}[x_1, \ldots, x_s] \; \text{ and } \; 1 \leq i \leq t. \tag{2.3}$$

The criterion (2.3) is called the *closedness condition* [**15**].

For $\alpha \in \mathbb{N}$, the subspace $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ consists of functionals in $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ with differential orders bounded by α.

The multiplicity structure of the ideal $\mathcal{I} = \langle f_1, \cdots, f_t \rangle$ at an isolated zero $\hat{\mathbf{x}}$ can be computed as the bases for the dual subspaces $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ for $\alpha = 0, 1, \cdots$ until reaching the smallest α, say $\alpha = \sigma$, such that

$$\mathit{dim}\Big(\mathcal{D}^{\sigma}_{\hat{\mathbf{x}}}(\mathcal{I})\Big) \;=\; \mathit{dim}\Big(\mathcal{D}^{\sigma+1}_{\hat{\mathbf{x}}}(\mathcal{I})\Big) \tag{2.4}$$

where $\mathit{dim}(\cdot)$ denotes the dimension of a vector space. When (2.4) occurs, all the subsequent dual subspaces $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathcal{D}^{\sigma}_{\hat{\mathbf{x}}}(\mathcal{I})$ for all $\alpha \geq \sigma$. Thus $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathcal{D}^{\sigma}_{\hat{\mathbf{x}}}(\mathcal{I})$. This σ is called the *depth* of the dual space $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ and represents the highest differential order in the dual space. The following lemma transforms the problem of finding the dual space to identifying matrix kernels.

LEMMA 2.1. [**5**] Let $\hat{\mathbf{x}}$ be an isolated zero of the ideal $\mathcal{I} = \langle f_1, \cdots, f_t \rangle \subset \mathbb{C}[x_1, \cdots, x_s]$. For $\alpha \in \mathbb{N}$, a differential functional $\sum_{|\mathbf{j}| \leq \alpha} c_{\mathbf{j}}\, \partial_{\mathbf{j}}[\hat{\mathbf{x}}]$ is in the dual subspace $\mathcal{D}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ if and only if the coefficient vector $\mathbf{c} = \big[c_{\mathbf{j}} \;:\; |\mathbf{j}| \leq \alpha\big] \in \mathbb{C}^{n_\alpha}$ satisfies the homogeneous system of equations

$$\textstyle\sum_{|\mathbf{j}| \leq \alpha} c_{\mathbf{j}}\, \partial_{\mathbf{j}}\big((\mathbf{x} - \hat{\mathbf{x}})^{\mathbf{k}} f_i \big)(\hat{\mathbf{x}}) \;=\; 0, \quad \text{for } \; \mathbf{k} \in \mathbb{N}^s, \;\; |\mathbf{k}| < \alpha, \;\; i \in \{1, 2, \cdots, s\}, \tag{2.5}$$

which can be written in the form of $S_\alpha\, \mathbf{c} = \mathbf{0}$ for $\mathbf{c} \in \mathbb{C}^{n_\alpha}$. Here S_α is called the α-th order Macaulay matrix of size $m_\alpha \times n_\alpha$ with

$$m_\alpha \;=\; \binom{\alpha - 1 + s}{\alpha - 1} t \quad \text{ and } \quad n_\alpha \;=\; \binom{\alpha + s}{\alpha} \;\text{ for } \alpha > 0 \tag{2.6}$$

and $S_0 \equiv \big[\, f_1(\hat{\mathbf{x}}), \cdots, f_t(\hat{\mathbf{x}}) \,\big]^{\top} = O_{t \times 1}$.

An algorithm is constructed accordingly for computing the multiplicity structure [**5**] along with the software module `MultiplicityStructure` in the Maple package `ApaTools` [**19, 20**]. The algorithm can be used in both symbolic and numerical computation. If the exact zero $\hat{\mathbf{x}}$ is available and exact arithmetic is applied in computing the matrix kernels, an exact basis for the dual space can be computed. Otherwise, numerical rank-revealing [**5, 9**] is employed in computing an approximate basis for the dual space and the exact multiplicity.

However, the sizes of Macaulay matrices given in (2.6) can easily exceed the storage capacity of a personal computer, or can become large enough to require excessive computing time. The objective of this paper is to construct alternative matrices, of much smaller sizes, that produce the same kernels. A sequence of Macaulay matrices $S_0,\ S_1,\ \ldots$ are shown in Figure 1 in § 6 in comparison with their counterparts of small sizes involved in the algorithm in this paper.

3. The closedness subspace and the closedness support

The main drawback of the Macaulay matrices S_α derived from equation (2.5) is its underlying domain of the vector space $\mathit{span}\big\{\ \partial_{\mathbf{j}}\ \big|\ \mathbf{j}\in\mathbb{N}^s,\quad |\mathbf{j}|\le\alpha\big\}$ as a linear transformation. This vector space has a dimension n_α given in (2.5) that can be prohibitively too high. To reduce the sizes of the matrices used in the multiplicity computation, we need subspaces of low dimensions where the same dual space resides. The closedness subspaces to be described here serve that purpose. The following lemma, which is originally applied by Stetter and Thallinger in their multiplicity algorithm [**15, 17**], rephrases the closedness condition (2.3) to an equivalent form.

LEMMA 3.1. [**15**, Theorem 8.36] Let $\mathcal{I}=\langle f_1,\ldots,f_t\rangle\in\mathbb{C}[x_1,\ldots,x_s]$ be a polynomial ideal and let $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ be its dual space at an isolated zero $\hat{\mathbf{x}}$. Then $c\in\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ if and only if

$$c(f_1)\ =\ c(f_2)\ =\ \cdots\ =\ c(f_t)\ =\ 0,\quad\text{and} \tag{3.1}$$

$$\Phi_1(c),\ \Phi_2(c),\ \ldots,\ \Phi_s(c)\ \in\ \mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I}) \tag{3.2}$$

where $\Phi_1,\ldots,\Phi_s$ are the linear *anti-differentiation operators* defined by

$$\Phi_\sigma\big(\partial_{j_1\ldots j_s}[\hat{\mathbf{x}}]\big)\ =\ \begin{cases} 0 & \text{if } j_\sigma=0\\ \partial_{j'_1\ldots j'_s}[\hat{\mathbf{x}}] & \text{otherwise}\end{cases}$$

with $j'_\sigma=j_\sigma-1$ and $j'_i=j_i$ for $i\in\{1,\cdots,s\}\backslash\{\sigma\}$.

Using an example in [**5**], the ideal $\mathcal{I}=\langle x_1^3,\ x_1^2x_2+x_2^4\rangle$ has a zero $\hat{\mathbf{x}}=\mathbf{0}\equiv(0,0)$ of multiplicity 12 with the dual space

$$\mathcal{D}_{\mathbf{0}}(\mathcal{I})=\mathit{span}\{\ \overbrace{\partial_{00}}^{0},\ \overbrace{\partial_{10},\partial_{01}}^{1},\overbrace{\partial_{20},\partial_{11},\partial_{02}}^{2},\overbrace{\partial_{12},\partial_{03}}^{3},\ \overbrace{\partial_{13},\partial_{04}\text{-}\partial_{21}}^{4},\overbrace{\partial_{05}\text{-}\partial_{22}}^{5},\overbrace{\partial_{06}\text{-}\partial_{23}}^{6}\ \} \tag{3.3}$$

where the basis functionals are grouped according to the orders of the differentials. The functional $\partial_{06}-\partial_{23}$ belongs to the dual subspace $\mathcal{D}^6_{\mathbf{0}}(\mathcal{I})$, and by Lemma 3.1, $\Phi_1(\partial_{06}-\partial_{23})=0-\partial_{13}$ and $\Phi_2(\partial_{06}-\partial_{23})=\partial_{05}-\partial_{22}$ are both in $\mathcal{D}^5_{\mathbf{0}}(\mathcal{I})$, as shown in (3.3).

Lemma 3.1 implies that each dual subspace $\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ is a subspace of the α-th order *closedness subspace*

$$\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})\ =\ \big\{c=\textstyle\sum_{|\mathbf{j}|\le\alpha}c_{\mathbf{j}}\partial_{\mathbf{j}}[\hat{\mathbf{x}}]\ \big|\ \Phi_\sigma(c)\in\mathcal{D}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I}),\ \sigma=1,\ldots,s\big\} \tag{3.4}$$

which will be shown to have a much lower dimension than n_α in (2.6).

The significance of the closedness subspace is two-fold. First of all, the dimension of the closedness subspace is much lower than that of the monomial subspace $span\{\partial_{\mathbf{j}} \,|\, |\mathbf{j}| \le \alpha\}$, leading to fewer columns in the matrices for kernel computations. Moreover, the functional $c \in \mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ is only required to satisfy t constraints in (3.1) since the closedness condition (2.3) is already satisfied by design. Consequently, the homogeneous system of equations (3.1) corresponds to a matrix whose row dimension is t instead of m_α in (2.6).

PROPOSITION 3.1. Let $\mathcal{I} = \langle f_1, \ldots, f_t\rangle \in \mathbb{C}[x_1, \ldots, x_s]$ be a polynomial ideal and let $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ be the α-th order closedness subspace of $\mathcal{I}$ at $\hat{\mathbf{x}}$ defined in *(3.4)*. If $\{c_1, \ldots, c_m\}$ forms a basis for $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$, then

$$\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I}) \;=\; \big\{\textstyle\sum_{i=1}^m u_j c_j \,\big|\, \sum_{i=1}^m u_j c_j(f_i) = 0,\ i = 1, \ldots, t\big\} \tag{3.5}$$

$$m \;=\; dim\big(\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})\big) \;\le\; t + dim\big(\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})\big) \;\le\; t + dim\big(\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})\big) \tag{3.6}$$

Proof. The equation (3.5) follows directly from Lemma 3.1 and the definition of $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$. Consequently, $\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ is isomorphic to the kernel of the matrix

$$W_\alpha \;=\; \begin{bmatrix} c_1(f_1) & c_2(f_1) & \cdots & c_m(f_1) \\ c_1(f_2) & c_2(f_2) & \cdots & c_m(f_2) \\ \vdots & \vdots & \ddots & \vdots \\ c_1(f_t) & c_2(f_t) & \cdots & c_m(f_t) \end{bmatrix}, \tag{3.7}$$

leading to (3.6). □

For example, the largest matrix W_9 whose kernel determines the dual space of the KSS system (1.2) for $n = 9$ is of size 9×265, which is almost negligible in comparison with the 218790×48620 Macaulay matrix for the same problem.

For computing the closedness subspace, we also need a monomial basis

$$M^\alpha_{\hat{\mathbf{x}}}(\mathcal{I}) \;=\; \big\{\, \partial_{\mathbf{j}} \,\big|\, 0 < |\mathbf{j}| \le \alpha,\ \Phi_\sigma(\partial_{\mathbf{j}}) \in S^{\alpha-1}_{\hat{\mathbf{x}}}(\mathcal{I}) \ \text{ for } \sigma = 1, \ldots, s\big\} \cup \{\partial_{0\ldots0}\} \tag{3.8}$$

which we refer to as the α-th order *closedness support*, where the α-th order *dual support* is defined by

$$S^\alpha_{\hat{\mathbf{x}}}(\mathcal{I}) \;=\; \big\{\, \partial_{\mathbf{j}} \,\big|\, \exists c = \textstyle\sum_{|\mathbf{i}|\le\alpha} c_{\mathbf{i}} \partial_{\mathbf{i}}[\hat{\mathbf{x}}] \in \mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I}) \text{ such that } c_{\mathbf{j}} \ne 0\big\} \tag{3.9}$$

for $\alpha \ge 0$. The closedness support $M^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ spans an intermediate subspace between the closedness subspace $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ and the vector space spanned by all differential monomials of order α or less:

$$\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I}) \;\subset\; \mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I}) \;\subset\; span\big(M^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})\big) \;\subset\; span\{\partial_{\mathbf{j}} \,|\, |\mathbf{j}| \le \alpha\}.$$

From the definition of the dual support $S^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ in (3.9), the third order monomial $\partial_{21} \notin S^3_{\mathbf{0}}(\mathcal{I})$ for the dual space is in (3.3), but $\partial_{21} \in S^4_{\mathbf{0}}(\mathcal{I})$. Namely, an α-th order monomial may not be in the α-th order dual support, but can be in the dual supports of higher orders.

Using the dual space (3.3) as an example, the 5-th order dual support is

$$S^5_{\mathbf{0}}(\mathcal{I}) \;=\; \{\partial_{00},\ \partial_{10},\ \partial_{01},\ \partial_{20},\ \partial_{11},\ \partial_{02},\ \partial_{12},\ \partial_{03},\ \partial_{21},\ \partial_{04},\ \partial_{13},\ \partial_{22},\ \partial_{05}\}$$

for $\mathcal{I} = \langle x_1^3,\ x_1^2x_2 + x_2^4\rangle$. As a result, the following 6-th order differential monomials $\partial_{60},\ \partial_{51},\ \partial_{42},\ \partial_{33},\ \partial_{24},\ \partial_{15}$ are not in $M^6_{\mathbf{0}}(\mathcal{I})$ since $\Phi_1(\partial_{60}) = \partial_{50},\ \Phi_2(\partial_{33}) = \partial_{32}$,

etc. do not belong to the dual support $S^5_{\mathbf{0}}(\mathcal{I})$, and the cardinality increases only by two from $M^5_{\mathbf{0}}(\mathcal{I})$ to $M^6_{\mathbf{0}}(\mathcal{I})$: $M^6_{\mathbf{0}}(\mathcal{I}) = M^5_{\mathbf{0}}(\mathcal{I}) \cup \{\partial_{06}, \partial_{23}\}$. Similarly,

$$\begin{aligned} M^0_{\mathbf{0}}(\mathcal{I}) &= \{\partial_{0\ldots0}\}, \quad M^1_{\mathbf{0}}(\mathcal{I}) = M^0_{\mathbf{0}}(\mathcal{I}) \cup \{\partial_{10}, \partial_{01}\}, \\ M^2_{\mathbf{0}}(\mathcal{I}) &= M^1_{\mathbf{0}}(\mathcal{I}) \cup \{\partial_{20}, \partial_{11}, \partial_{02}\}, \quad M^3_{\mathbf{0}}(\mathcal{I}) = M^2_{\mathbf{0}}(\mathcal{I}) \cup \{\partial_{30}, \partial_{21}, \partial_{12}, \partial_{03}\}, \\ M^4_{\mathbf{0}}(\mathcal{I}) &= M^3_{\mathbf{0}}(\mathcal{I}) \cup \{\partial_{13}, \partial_{04}\} \text{ and } M^5_{\mathbf{0}}(\mathcal{I}) = M^4_{\mathbf{0}}(\mathcal{I}) \cup \mathit{span}\{\partial_{14}, \partial_{22}, \partial_{05}\}. \end{aligned}$$

These monomial supports span subspaces in which we shall identify $\mathcal{C}^4_{\mathbf{0}}(\mathcal{I})$ and $\mathcal{C}^5_{\mathbf{0}}(\mathcal{I})$ with much smaller dimensions than the subspaces spanned by all monomials.

Finding the closedness subspaces requires matrix computations, as we shall discuss later in §4. The closedness supports are relatively easier to establish using the standard binary search. As a counterpart of the anti-differentiation operator Φ_σ, a linear differentiation operator Ψ_σ is defined as

$$\Psi_\sigma(\partial_{j_1\ldots j_s}) = \partial_{\hat{j}_1\ldots\hat{j}_s} \tag{3.10}$$

where $\hat{j}_i = j_i$ for $i \in \{1, \ldots, s\}\backslash\{\sigma\}$ and $\hat{j}_\sigma = j_\sigma + 1$. Starting from the known subspaces and the trivial support

$$\mathcal{C}^0_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathcal{D}^0_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathit{span}\{\partial_{0\ldots0}\}, \text{ and } S^0_{\hat{\mathbf{x}}}(\mathcal{I}) = \{\partial_{0\ldots0}\}, \tag{3.11}$$

it is straightforward to verify that $\mathcal{C}^1_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathit{span}\big(M^1_{\hat{\mathbf{x}}}(\mathcal{I})\big)$ with

$$M^1_{\hat{\mathbf{x}}}(\mathcal{I}) = \{\partial_{0\ldots0}, \partial_{10\ldots0}, \partial_{01\ldots0}, \ldots, \partial_{0\ldots01}\} = \big\{\, \partial_{\mathbf{j}} \,\big|\, |\mathbf{j}| \le 1\big\}$$

for any ideal $\mathcal{I}$ at any isolated zero $\hat{\mathbf{x}}$. We can then proceed to find the dual subspace $\mathcal{D}^1_{\hat{\mathbf{x}}}(\mathcal{I})$ from $\mathcal{C}^1_{\hat{\mathbf{x}}}(\mathcal{I})$ by the algorithm (6.1) in §6. Generally, after identifying the dual subspace $\mathcal{D}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$ of order $\alpha-1$ and its monomial support $S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$, the closedness support $M^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ of order up to α is to be established in two steps. The first step creates a temporary set T of possible monomials of order up to α by applying the differential operator Ψ_σ for $\sigma = 1, \ldots, s$ on all the $(\alpha-1)$-th order monomials in the support $S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$ of the dual subspace $\mathcal{D}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$. Then the second step checks each monomial $\partial_{\mathbf{j}} \in T$ to see if it is in the dual support $S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$, and excludes those monomials that fail the anti-differentiation test.

The following is the pseudo-code for setting up the closedness support $M^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ of order α from the dual support $S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$ of order $\alpha-1$.

(3.12)

```
Initialize the temporary set T = ∅
% step 1: create the set of possible monomials
for σ = 1, ..., s do
    Expand T = T ∪ Ψ_σ(S^{α-1}_x̂(I) \ S^{α-2}_x̂(I))
% step 2: exclude monomials not satisfying the closedness condition
for every ∂_j ∈ T do
    if Φ_σ(∂_j) ∉ S^{α-1}_x̂(I) for any σ ∈ {1, ..., s}, then
        Update T = T \ {∂_j}
    end if
Output M^α_x̂(I) = M^{α-1}_x̂(I) ∪ T
```

There are two for-do loops in the pseudo-code (3.12) corresponding to the two steps described above. Obtaining the closedness support is particularly effective in achieving high efficiency when the dual space possesses high sparsity in the sense of monomial support.

Using the dual space (3.3) as an example for $\alpha = 3$, we have $S^3_{\mathbf{0}}(\mathcal{I})\backslash S^2_{\mathbf{0}}(\mathcal{I}) = \{\partial_{12}, \partial_{03}\}$ and $\Psi_1\big(S^3_{\mathbf{0}}(\mathcal{I})\backslash S^2_{\mathbf{0}}(\mathcal{I})\big) = \{\partial_{22}, \partial_{13}\}$, $\Psi_2\big(S^3_{\mathbf{0}}(\mathcal{I})\backslash S^2_{\mathbf{0}}(\mathcal{I})\big) = \{\partial_{13}, \partial_{04}\}$.

Thus the first for-do loop results in $T = \{\partial_{22},\ \partial_{13},\ \partial_{04}\}$. After that, the second for-do loop check the three monomials one by one using the anti-differentiation operators Φ_1 and Φ_2, excluding ∂_{22} from T since $\Phi_2(\partial_{22}) = \partial_{21} \notin S^3_{\mathbf{0}}(\mathcal{I})$ and obtaining $M^3_{\hat{\mathbf{x}}}(\mathcal{I}) = M^2_{\hat{\mathbf{x}}}(\mathcal{I}) \cup \{\partial_{13},\ \partial_{04}\}$.

4. Computing the closedness subspace

Computation of the closedness subspace essentially comes down to matrix rank/kernel identification. Assume that $\mathcal{D}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$, $\mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$, $M^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$, and $S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$ are known at a certain stage with $\alpha > 0$, as they are given in (3.11) for $\alpha = 1$. Then the α-th order dual support $M^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ can also be assumed to be obtained by the procedure (3.12). We shall give a detailed description of the algorithm for computing the closedness subspace $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ as a subspace of $\mathit{span}\big(M^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})\big)$. Fundamental matrix computations will naturally arise from the closedness condition in Lemma 3.1.

Using a monomial ordering, we can write

$$M^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I}) = \big\{\partial_{\mathbf{j}_1}, \partial_{\mathbf{j}_2}, \ldots, \partial_{\mathbf{j}_m}\big\} \cup \{\partial_{0\ldots0}\} \quad \text{and} \quad \mathcal{D}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathit{span}\{c_1, c_2, \ldots, c_n\} \tag{4.1}$$

where each $c_i \in \mathit{span}\big(S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})\big)$. From $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I}) \in \mathit{span}\big(M^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})\big)$ and Lemma 3.1, the identification of $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ is equivalent to finding z_i's in $\mathbb{C}$ such that there exist $y_{\sigma k}$'s in $\mathbb{C}$ satisfying

$$\begin{aligned} \Phi_\sigma\big(z_1\partial_{\mathbf{j}_1} + z_2\partial_{\mathbf{j}_2} + \cdots + z_m\partial_{\mathbf{j}_m}\big) &= y_{\sigma 1}c_1 + y_{\sigma 2}c_2 + \cdots + y_{\sigma n}c_n, \\ &\text{for } \sigma = 1, \ldots, s. \end{aligned} \tag{4.2}$$

By the definitions of the closedness support $M^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ and the dual support $S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$, each $\Phi_\sigma(\partial_{\mathbf{j}_i})$ is either zero or a monomial in $S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$, and each c_i is a linear combination of monomials in $S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$. Equating the coefficients of each $\partial_{\mathbf{j}_i} \in S^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$ on both sides of (4.2) for $\sigma \in \{1, \ldots, s\}$ yields equations

$$z_i = d_{i\sigma 1}y_{\sigma 1} + d_{i\sigma 2}y_{\sigma 2} + \cdots + d_{i\sigma n}y_{\sigma n} \quad \text{if } \Phi_\sigma(\partial_{\mathbf{j}_i}) \neq 0 \tag{4.3}$$

$$0 = d_{i\sigma 1}y_{\sigma 1} + d_{i\sigma 2}y_{\sigma 2} + \cdots + d_{i\sigma n}y_{\sigma n} \quad \text{if } \Phi_\sigma(\partial_{\mathbf{j}_i}) = 0, \quad \text{for } i = 1, \ldots, m. \tag{4.4}$$

For each $i \in \{1, \ldots, m\}$, picking the equation (4.3) for the smallest σ leads to

$$\mathbf{z} = A\,\mathbf{y} \tag{4.5}$$

where $\mathbf{z} = [z_1, \cdots, z_m]^\top \in \mathbb{C}^m$, $\mathbf{y} = [y_{11}, \ldots, y_{1n}, y_{21}, \ldots, y_{2n}, \ldots, y_{sn}]^\top \in \mathbb{C}^{sn}$ and the matrix $A \in \mathbb{C}^{m\times sn}$. The remaining equations in (4.3) can then be rewritten in the form of

$$d_{i\sigma 1}y_{\sigma 1} + d_{i\sigma 2}y_{\sigma 2} + \cdots + d_{i\sigma n}y_{\sigma n} = d_{i\mu 1}y_{\mu 1} + d_{i\mu 2}y_{\mu 2} + \cdots + d_{i\mu n}y_{\sigma n} \tag{4.6}$$

by equating two expressions of z_i corresponding to different σ's. Combining (4.6) and (4.4), we have a homogeneous linear system of equations

$$B\,\mathbf{y} = \mathbf{0}. \tag{4.7}$$

The α-th order closedness subspace is thus

$$\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I}) = \big\{z_1\partial_{\mathbf{j}_1} + \cdots + z_m\partial_{\mathbf{j}_m} \;\big|\; \mathbf{z} = A\mathbf{y} \text{ and } B\mathbf{y} = \mathbf{0}\big\}.$$

Since the subspace $\mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$ of $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ is known at α, we need only the orthogonal complement $\mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})^{\perp}$ in $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$. Let $G = [g_{ik}] \in \mathbb{C}^{m\times\ell}$ be the matrix such that

$$\mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathit{span}\big\{g_{1k}\partial_{\mathbf{j}_1} + \cdots + g_{mk}\partial_{\mathbf{j}_m} \;\big|\; k = 1, \ldots, \ell\big\}. \tag{4.8}$$

Then the orthogonal complement $\mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})^{\perp}$ can be obtained by imposing an additional constraint

$$G^{\mathsf{H}}\mathbf{z} \;=\; G^{\mathsf{H}}A\mathbf{y} \;=\; \mathbf{0}. \tag{4.9}$$

Denote the kernels of B and $G^{\mathsf{H}}A$ as $\mathcal{K}(B)$ and $\mathcal{K}(G^{\mathsf{H}}A)$ respectively. Let N be the matrix whose columns span the joint kernel $\mathcal{K}(B)\cap\mathcal{K}(G^{\mathsf{H}}A)$, namely, the general solution to both (4.7) and (4.9) is $\mathbf{y}=N\mathbf{u}$. We have

$$\mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})^{\perp} \;=\; \big\{z_1\partial_{\mathbf{j}_1}+\cdots+z_m\partial_{\mathbf{j}_m} \;\big|\; \mathbf{z}=AN\mathbf{u}\big\} \tag{4.10}$$

and $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I}) \oplus \mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})^{\perp}$. We thereby completed a constructive proof of the following proposition.

PROPOSITION 4.1. Assume the dual subspace $\mathcal{D}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}$, the closedness subspace $\mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$, and the closedness support $M^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I}) = \{\partial_{\mathbf{j}_1},\ldots,\partial_{\mathbf{j}_m}\}\cup\{\partial_{0\ldots0}\}$ are available at the isolated zero $\hat{\mathbf{x}}$ to an ideal $\mathcal{I}\subset\mathbb{C}[x_1,\ldots,x_s]$ for a differential order $\alpha>0$. Then the α-th order closedness subspace $\mathcal{C}^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I}) = \mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I}) \oplus \mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})^{\perp}$ with

$$\mathcal{C}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})^{\perp} \;=\; \big\{z_1\partial_{\mathbf{j}_1}+\cdots+z_m\partial_{\mathbf{j}_m} \;\big|\; [z_1,\ldots,z_m]^{\top}=AN\mathbf{u},\ \mathbf{u}\in\mathbb{C}^{l}\big\}. \tag{4.11}$$

Here the matrix A is defined as in *(4.5)* and the columns of the matrix N span the joint kernel $\mathcal{K}(B)\cap\mathcal{K}(G^{\mathsf{H}}A)$ with B and G being given in *(4.7)* and *(4.9)* respectively. Furthermore, the column dimension of the matrices A and B are bounded by $s\cdot \mathit{dim}\big(\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})\big)$ for all α.

The following example illustrates this proposition.

EXAMPLE 4.1. Consider the ideal $\mathcal{I}=\langle x^2y^2-x^2+y^2, x^2-y^2\rangle$ at the zero $\mathbf{0}$. Suppose the following subspaces and monomial supports are available

$$\begin{aligned}\mathcal{D}^{1}_{\mathbf{0}}(\mathcal{I}) \;&=\; \mathcal{C}^{1}_{\mathbf{0}}(\mathcal{I}) \;=\; \mathit{span}\{\,\partial_{00},\ \partial_{10}+\partial_{01},\ \partial_{10}-\partial_{01}\,\}\\ M^{1}_{\mathbf{0}}(\mathcal{I}) \;&=\; \{\,\partial_{00},\ \partial_{10},\ \partial_{01}\,\},\quad S^{1}_{\mathbf{0}}(\mathcal{I}) \;=\; \{\,\partial_{00},\ \partial_{10},\ \partial_{01}\,\}.\end{aligned}$$

The procedure (3.12) identifies the second order closedness monomial support

$$M^{2}_{\mathbf{0}}(\mathcal{I}) \;=\; \{\,\partial_{10},\ \partial_{01},\ \partial_{20},\ \partial_{11},\ \partial_{02}\,\}\cup\{\partial_{00}\}.$$

To calculate $\mathcal{C}^{2}_{\mathbf{0}}(\mathcal{I})$, we first generate the equations (4.2) in this case as

$$\begin{aligned}&\Phi_1(z_1\partial_{10}+z_2\partial_{01}+z_3\partial_{20}+z_4\partial_{11}+z_5\partial_{02}) \;=\\ &\qquad z_1\partial_{00}+z_3\partial_{10}+z_4\partial_{01} \;=\; y_{11}\partial_{00}+y_{12}(\partial_{10}+\partial_{01})+y_{13}(\partial_{10}-\partial_{01})\\ &\Phi_2(z_1\partial_{10}+z_2\partial_{01}+z_3\partial_{20}+z_4\partial_{11}+z_5\partial_{02}) \;=\\ &\qquad z_2\partial_{00}+z_4\partial_{10}+z_5\partial_{01} \;=\; y_{21}\partial_{00}+y_{22}(\partial_{10}+\partial_{01})+y_{23}(\partial_{10}-\partial_{01})\end{aligned}$$

leading to the equation (4.5)

$$\mathbf{z} \;=\; \begin{bmatrix} z_1\\ z_2\\ z_3\\ z_4\\ z_5\end{bmatrix} \;=\; \begin{bmatrix} 1 & & & & & \\ & & & 1 & & \\ & 1 & 1 & & & \\ & 1 & -1 & & & \\ & & & & 1 & -1\end{bmatrix}\begin{bmatrix} y_{11}\\ y_{12}\\ y_{13}\\ y_{21}\\ y_{22}\\ y_{23}\end{bmatrix} \;=\; A\,\mathbf{y}$$

and (4.7) from equating $z_4=y_{12}-y_{13}$ and $z_4=y_{22}+y_{23}$ and

$$B\mathbf{y} \;=\; \begin{bmatrix} 0 & 1 & -1 & 0 & -1 & -1\end{bmatrix}\mathbf{y} \;=\; 0.$$

Using the existing $\mathcal{C}^{1}_{\mathbf{0}}(\mathcal{I})$, the matrix G in (4.9) is

$$G \;=\; \begin{bmatrix} 1 & 1 & 0 & 0 & 0\\ 1 & -1 & 0 & 0 & 0\end{bmatrix}^{\mathsf{H}},$$

leading to the equation $G^{\mathsf{H}} A\,\mathbf{y} = \mathbf{0}$. The joint kernel $\mathcal{K}(B) \cap \mathcal{K}(G^{\mathsf{H}}A)$ can then be computed as the column space of the matrix N and thus $\mathbf{z} = AN\mathbf{u}$ where

$$N \;=\; \begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix} \quad \text{and} \quad AN \;=\; \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 2 \\ 1 & 1 & 0 \\ -1 & 1 & 0 \end{bmatrix}$$

implying that $\mathcal{C}^1_{\mathbf{0}}(\mathcal{I})^{\perp} = \mathit{span}\{\partial_{20}+\partial_{11}-\partial_{02},\ \partial_{20}+\partial_{11}+\partial_{02},\ 2\partial_{20}\}$ is the orthogonal complement of $\mathcal{C}^1_{\mathbf{0}}(\mathcal{I})$ in $\mathcal{C}^2_{\mathbf{0}}(\mathcal{I}) = \mathcal{C}^1_{\mathbf{0}}(\mathcal{I}) \oplus \mathcal{C}^1_{\mathbf{0}}(\mathcal{I})^{\perp}$. □

5. Exploiting the block structure of matrices

The main cost of computing the closedness subspaces occurs at manipulating the matrices A and B in (4.5) and (4.7) respectively. Both matrices can be highly sparse if the dual space is spanned by sparse functionals. At minimum, one can exploit the block structures to reduce both storage and computing time.

Assume again the closedness support $M^{\alpha}_{\hat{\mathbf{x}}}(\mathcal{I})$ and the dual subspace $\mathcal{D}^{\alpha\text{-}1}_{\hat{\mathbf{x}}}(\mathcal{I})$ are written as in (4.1), while $\mathbf{z} = [z_1, \ldots, z_m]^{\top}$ and $\mathbf{y} = [\mathbf{y}_1^{\top}, \ \ldots, \ \mathbf{y}_s^{\top}]^{\top}$ with $\mathbf{y}_\sigma = [y_{\sigma 1}, \ldots, y_{\sigma n}]^{\top}$ for $\sigma = 1, \ldots, s$ as in (4.2). It is clear from (4.3) that there exists a permutation matrix P such that $P\mathbf{z} = [z_{i_1}, \ldots, z_{i_m}]^{\top}$ and

$$P\mathbf{z} \;=\; (P\,A)\,\mathbf{y} \;=\; \begin{bmatrix} A_1 & & & \\ & A_2 & & \\ & & \ddots & \\ & & & A_s \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \vdots \\ \mathbf{y}_s \end{bmatrix} \tag{5.1}$$

where $A_1, \ldots, A_s$ have the identical column dimension n. Consequently, the matrix A in (4.5) can be stored as an $m \times n$ matrix containing the nonzero blocks A_1, …, A_s only, and the matrix multiplication $G^{\mathsf{H}}A$ and AN in (4.9) and (4.10) respectively can be carried out by block matrix multiplication. The resulting storage and the operation count is a fraction of those in straightforward computation without exploiting the block structure.

Computing the kernel $\mathcal{K}(B)$ of the matrix B in (4.7) requires its triangularization by either elementary row operations or the QR decomposition. To save storage and computing time, the construction and triangularization of B can be accomplished interactively blockwise. The first block of B is generated by the equations (4.4) for $\sigma = 1$ in the form of $B^{(1)}_1 \mathbf{y}_1 = \mathbf{0}$. The block $B^{(1)}_1$ of B can then be triangularized in $B_1(1) \;=\; G_{11} T^{(1)}_{11}$ by either QR or pivoted LU decomposition, where $T^{(1)}_{11}$ is an upper-triangular matrix.

For $\sigma = 2$, the equation (4.2) leads to the equations (4.4) in the form of $B^{(2)}_2 \mathbf{y}_2 = \mathbf{0}$ and the equations (4.6) in the form of $B^{(2)}_1 \mathbf{y}_1 + C^{(2)}_1 \mathbf{y}_2 = \mathbf{0}$, resulting in the upper-left portion of B being generated and then triangularized as

$$\begin{bmatrix} B^{(1)}_1 & & \\ B^{(2)}_1 & C^{(2)}_1 & \\ & B^{(2)}_2 & \end{bmatrix} \;=\; \begin{bmatrix} G_{11} & & \\ & I & \\ & & I \end{bmatrix} \begin{bmatrix} T_{11} & & \\ B^{(2)}_1 & C^{(2)}_1 & \\ & B^{(2)}_2 & \end{bmatrix} \;\Longrightarrow\; \begin{bmatrix} T^{(2)}_{11} & T^{(2)}_{12} \\ & T^{(2)}_{22} \end{bmatrix}$$

using block matrix decompositions, where $T^{(2)}_{11}$ and $T^{(2)}_{22}$ are upper-triangular matrices. In general, after constructing matrix blocks of B at the integer $\sigma = \mu$

based on the equation (4.2) we obtain the triangularization

$$\begin{bmatrix} T_{11}^{(\mu)} & \cdots & T_{1\mu}^{(\mu)} \\ & \ddots & \vdots \\ & & T_{\mu\mu}^{(\mu)} \end{bmatrix}. \tag{5.2}$$

The equation (4.2) for $\sigma = \mu{+}1$ leads to equations (4.4) and (4.6) that can be written as $B_{\mu+1}^{(\mu+1)}\mathbf{y}_{\mu+1} = \mathbf{0}$ and $B_i^{(\mu+1)}\mathbf{y}_i + C_i^{(\mu+1)}\mathbf{y}_{\mu+1} = \mathbf{0}$ respectively in matrix form for $i = 1, \ldots, \mu$. Combining those equations with (5.2) yields further construction of matrix B and its triangularization

$$\left[\begin{array}{ccc|c} T_{11}^{(\mu)} & \cdots & T_{1\mu}^{(\mu)} & \\ & \ddots & \vdots & \\ & & T_{\mu\mu}^{(\mu)} & \\ \hline B_1^{(\mu+1)} & & & C_1^{(\mu+1)} \\ & \ddots & & \vdots \\ & & B_\mu^{(\mu+1)} & C_\mu^{(\mu+1)} \\ \hline & & & B_{\mu+1}^{(\mu+1)} \end{array}\right] \Longrightarrow \left[\begin{array}{ccc|c} T_{11}^{(\mu+1)} & \cdots & T_{1\mu}^{(\mu+1)} & T_{1,\mu+1}^{(\mu+1)} \\ & \ddots & \vdots & \vdots \\ & & T_{\mu\mu}^{(\mu+1)} & T_{\mu,\mu+1}^{(\mu+1)} \\ \hline & & & T_{\mu+1,\mu+1}^{(\mu+1)} \\ & & & \\ \hline & & & \end{array}\right]. \tag{5.3}$$

The triangularization "$\Longrightarrow$" above involves eliminating $B_1^{(\mu+1)}$, ..., $B_\mu^{(\mu+1)}$ using the upper-triangular blocks $T_{11}^{(\mu)}$, ..., $T_{\mu\mu}^{(\mu)}$ respectively, and triangularizing the resulting block originally occupied $C_1^{(\mu+1)}$, ..., $C_\mu^{(\mu+1)}, B_{\mu+1}^{(\mu+1)}$ into $T_{\mu+1,\mu+1}^{(\mu+1)}$.

Continuing the above construction/triangularization process from $\sigma = 1$ to $\sigma = s$, we reach the final triangularization $B = Q\,T$ where T is upper-triangular, Q is the accumulation of the transformations that is not explicitly needed. Notice that the kernel $\mathcal{K}(B)$ is identical to the kernel $\mathcal{K}(T)$. To compute the joint kernel $\mathcal{K}(B) \cap \mathcal{K}(G^{\mathsf{H}}A) = \mathcal{K}\left(\begin{bmatrix} G^{\mathsf{H}}A \\ T \end{bmatrix}\right)$, one needs to make the final triangularization of the trapezoidal shaped matrix $\begin{bmatrix} G^{\mathsf{H}}A \\ T \end{bmatrix}$ whose kernel can be efficiently computed by the rank-revealing method developed in [**9**].

The process of computing the closedness subspace $\mathcal{C}_{\hat{\mathbf{x}}}^{\alpha}(\mathcal{I})$ can now be summarized in the following pseudo-code.

$$\left\{\begin{array}{l} \texttt{initialize } A = O,\ T = O \\ \texttt{extract} \text{ rows of } A \text{ from (4.3) for } \sigma = 1 \\ \texttt{form } B_1^{(1)} \text{ from (4.4) and (4.6), triangularize it as } T_{11}^{(1)} \\ \texttt{for } \sigma = 2, 3, \ldots, s \texttt{ do} \\ \quad \texttt{extract} \text{ additional rows of } A \text{ from (4.3).} \\ \quad \texttt{form } B_1^{(\sigma)}, \ldots, B_\sigma^{(\sigma)} \text{ and } C_1^{(\sigma)}, \ldots, C_{\sigma-1}^{(\sigma)} \text{ from (4.4) and (4.6)} \\ \quad \texttt{carry out} \text{ the triangularization (5.3) for } \sigma = \mu{+}1 \\ \quad \texttt{update } T \\ \texttt{construct } G \text{ from (4.8)} \\ \texttt{compute } N \text{ whose columns span } \mathcal{K}(G^{\mathsf{H}}A) \cap \mathcal{K}(T) \\ \texttt{output } \mathcal{C}_{\hat{\mathbf{x}}}^{\alpha}(\mathcal{I}) = \mathcal{C}_{\hat{\mathbf{x}}}^{\alpha\text{-}1}(\mathcal{I}) \oplus \mathcal{C}_{\hat{\mathbf{x}}}^{\alpha\text{-}1}(\mathcal{I})^{\perp} \text{ with } \mathcal{C}_{\hat{\mathbf{x}}}^{\alpha\text{-}1}(\mathcal{I})^{\perp} \text{ in (4.11)} \end{array}\right. \tag{5.4}$$

6. Computing the dual space

At the stage α with the availability of the closedness subspace $\mathcal{C}_{\hat{\mathbf{x}}}^{\alpha}(\mathcal{I}) = \mathit{span}\{c_1, \ldots, c_m\}$ for the polynomial ideal $\mathcal{I} = \langle f_1, f_2, \ldots, f_t \rangle$, we can now describe the dual subspace $\mathcal{D}_{\hat{\mathbf{x}}}^{\alpha}(\mathcal{I})$ as (3.5) in Proposition 3.1 and compute it as the kernel

of the $t \times m$ matrix W_α in (3.7). The column dimensions of all such matrices W_α's are bounded by the multiplicity plus the number t of equations.

The initial $W_0 = O_{t\times 1}$ since $c_1 = \partial_{0\ldots0}$ and $f_1(\hat{\mathbf{x}}) = \cdots = f_t(\hat{\mathbf{x}}) = 0$. From an existing W_α, the new matrix $W_{\alpha+1}$ is recursively constructed by augmenting W_α with new columns generated by the basis functionals for $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})^\perp \cap \mathcal{C}^{\alpha+1}_{\hat{\mathbf{x}}}(\mathcal{I})$. The kernel $\mathcal{K}(W_\alpha)$ can be imbedded in $\mathcal{K}(W_{\alpha+1})$, $\mathcal{K}(W_{\alpha+2})$, ... by appending zeros the at the bottom of vectors in $\mathcal{K}(W_\alpha)$. The computation of the dual space $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ terminates in two ways. If the newly obtained closedness subspace $\mathcal{C}^{\alpha+1}_{\hat{\mathbf{x}}}(\mathcal{I})$ does not expand its dimension over $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$, namely $\mathcal{C}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})^\perp = \{0\}$, then there is no need to compute the dual subspace $\mathcal{D}^{\alpha+1}_{\hat{\mathbf{x}}}(\mathcal{I})$ since it does not expand over $\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$ either. Otherwise the process terminates when the computed kernel $\mathcal{K}(W_{\alpha+1})$ is of the same dimension as $\mathcal{K}\big(W_\alpha\big)$. In both cases the dual space $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ is obtained as $\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})$, the multiplicity is $\mathit{dim}\big(\mathcal{D}^\alpha_{\hat{\mathbf{x}}}(\mathcal{I})\big)$ and the depth is α.

The algorithm for computing the dual space $\mathcal{D}_{\hat{\mathbf{x}}}(\mathcal{I})$ can be described in the following pseudo-code.

(6.1)
```
Initialize D^0_x̂(I) = C^0_x̂(I) = span{∂_{0...0}} and W_0 = O_{t×1}
for α = 0, 1, ... do
    get C^{α+1}_x̂(I) = C^α_x̂(I) ⊕ C^α_x̂(I)^⊥ by computing C^α_x̂(I)^⊥
    if C^α_x̂(I)^⊥ = {0} then
      set D_x̂(I) = D^α_x̂(I),    break the for-do loop
    else
      expand W_α to W_{α+1} using new columns for C^α_x̂(I)^⊥
      get K(W_{α+1}) by computing K(W_α)^⊥ in K(W_{α+1})
      if K(W_α)^⊥ = 0, then
        set D_x̂(I) = D^α_x̂(I), break the for-do loop
      else
        set D^{α+1}_x̂(I) by isomorphism to K(W_{α+1})
      end if
    end if
output D_x̂(I)
```

We illustrate the process of computing the dual space in the following example as well as a comparison with the algorithm in [**5**] using the Macaulay matrices.

EXAMPLE 6.1. Consider $\langle f_1,\ f_2\rangle \subset \mathbb{C}[x_1, x_2]$ at the zero $(0, 0)$ where

$$f_1(x_1, x_2) \ = \ x_1 - x_2 + x_1^2, \quad f_1(x_1, x_2) \ = \ x_1 - x_2 + x_2^2.$$

This ideal is taken from [**5**] in which Figure 1 is used to show the expansion of Macaulay matrices S_0, ..., S_3 and the corresponding kernels. These kernels leads to the dual subspaces

$$\mathcal{D}^0_{\mathbf{0}}(\mathcal{I}) \ = \ \mathit{span}\{\partial_{00}\}, \quad \mathcal{D}^1_{\mathbf{0}}(\mathcal{I}) \ = \ \mathit{span}\{\partial_{00},\ \partial_{10}+\partial_{01}\}$$
$$\mathcal{D}^3_{\mathbf{0}}(\mathcal{I}) \ = \ \mathcal{D}^2_{\mathbf{0}}(\mathcal{I}) \ = \ \mathit{span}\{\partial_{00},\ \partial_{10}+\partial_{01},\ -\partial_{10}+\partial_{20}+\partial_{11}+\partial_{02}\}$$

In contrast to Figure 1, the algorithm (5.4) produces closedness subspaces

$$\mathcal{C}^0_{\mathbf{0}}(\mathcal{I}) \ = \ \mathit{span}\{\partial_{00}\},\ \ \mathcal{C}^1_{\mathbf{0}}(\mathcal{I}) \ = \ \mathcal{C}^0_{\mathbf{0}} \oplus \mathit{span}\{\partial_{10},\ \partial_{01}\},\ \ \mathcal{C}^2_{\mathbf{0}}(\mathcal{I}) \ = \ \mathcal{C}^1_{\mathbf{0}} \oplus \mathit{span}\{\partial_{20}+\partial_{11}+\partial_{02}\},$$
$$\mathcal{C}^3_{\mathbf{0}}(\mathcal{I}) \ = \ \mathcal{C}^2_{\mathbf{0}} \oplus \mathit{span}\{\partial_{30}+\partial_{21}+\partial_{12}+\partial_{03}-\partial_{20}+\partial_{02}\}$$

leading to the expansion of the matrices W_0, W_1, W_2 and W_3 shown as follows.

		column index				
		∂_{00}	∂_{10}	∂_{01}	$\partial_{20}+\partial_{11}+\partial_{02}$	$\partial_{30}+\partial_{21}+\partial_{12}+\partial_{03}-\partial_{20}+\partial_{02}$
	f_1	0	1	-1	1	-1
W_3 W_2 W_1 W_0	f_2	0	1	-1	1	1
bases for kernels (transposed as row vectors):						
$\mathcal{K}(W_0)$		1	0	0	0	0
$\mathcal{K}(W_1)$		0	1	1	0	0
$\mathcal{K}(W_3)$ $\mathcal{K}(W_2)$		0	-1	0	1	0

Here, $\mathcal{K}(W_0)$ and $\mathcal{K}(W_1)$ are identical to $\mathcal{K}(S_0)$ and $\mathcal{K}(S_1)$ in Figure 1 respectively. The new basis vector $[0,-1,0,1]^\top \in \mathcal{K}(W_2)$ represents the functional $(-1)\cdot\partial_{10}+1\cdot(\partial_{20}+\partial_{11}+\partial_{02})$ (c.f. the column indices of W_2). As a result, the same dual subspaces are produced with matrices with sizes up to 2×5, much smaller than the Macaulay matrices with sizes up to 12×10.

Macaulay matrices ↘		column index ∂_{00}	∂_{10}	∂_{01}	∂_{20}	∂_{11}	∂_{02}	∂_{30}	∂_{21}	∂_{12}	∂_{03}
	f_1	0	1	−1	1	0	0	0	0	0	0
S_0	f_2	0	1	−1	0	0	1	0	0	0	0
S_1	x_1f_1	0	0	0	1	−1	0	1	0	0	0
	x_1f_2	0	0	0	1	−1	0	0	0	1	0
	x_2f_1	0	0	0	0	1	−1	0	1	0	0
S_2	x_2f_2	0	0	0	0	1	−1	0	0	0	1
	$x_1^2f_1$	0	0	0	0	0	0	1	−1	0	0
	$x_1^2f_2$	0	0	0	0	0	0	1	−1	0	0
	$x_1x_2f_1$	0	0	0	0	0	0	0	1	−1	0
	$x_1x_2f_2$	0	0	0	0	0	0	0	1	−1	0
	$x_2^2f_1$	0	0	0	0	0	0	0	0	1	−1
S_3	$x_2^2f_2$	0	0	0	0	0	0	0	0	1	−1
bases for kernels (transposed as row vectors):											
$\mathcal{K}(S_0)$		1	0	0	0	0	0	0	0	0	0
$\mathcal{K}(S_1)$		0	1	1	0	0	0	0	0	0	0
$\mathcal{K}(S_2)$		0	−1	0	1	1	1	0	0	0	0
$\mathcal{K}(S_3)$											

FIGURE 1. The Macaulay matrices for Example 6.1.

7. Computational experiments

A preliminary Maple implementation with code name `Multiplicity` is tested in comparison with its predecessor `MultiplicityStructure` that was released as a module of the package `ApaTools`[**20**] described in [**19**]. All experiments are carried out using an ASUS R1E notebook computer with Intel Core 2 Duo CPU T7700 at 2.40GHz and 2GB of RAM on the platform of Maple 12 classic worksheet in Windows Vista. The Maple environment variables are set by statements "`UseHardwareFloats := true`" and "`Digits := 16`. All timing measures are elapsed time in seconds. We shall implement the Matlab version and upgrade `ApaTools/Apalab` due to the overwhelming advantage of the closedness subspace method over the previous Macaulay matrix based algorithms.

We first test `Multiplicity` using the same set of benchmark problems described in [**5**]. The new code produces the same multiplicity and the dual space for each test problem. The test results on these polynomial systems are listed in Table 1. Most of these problems are small in terms of system size, multiplicity and

depth, requiring less than one second for both codes. For such small problems, algorithm efficiency is not important. On the four larger systems the advantage of the new code starts to be noticeable. Among them, either the multiplicity is high (e.g. `DZ2` and `DZ1` of multiplicity 16 and 131 respectively) or the system size becomes larger (e.g. `Cyclic 9` of 9×9). The new code `Multiplicity` is about 10, 20, and 50 times faster respectively.

system	zero	depth	multiplicity	Previous time	New time
`cmbs1`	$(0,0,0)$	4	11	.437	.234
`cmbs2`	$(0,0,0)$	3	8	.172	.140
`mth191`	$(0,1,0)$	2	4	.047	.062
`Decker2`	$(0,0)$	3	4	.016	.078
`Ojika2`	$(0,0,1)$	1	2	.000	.047
	$(1,0,0)$	1	2	.000	.031
`Ojika3`	$(0,0,1)$	3	4	.156	.109
	$(-5/2,5/2,1)$	1	2	.016	.031
`KSS`	$(1,1,1,1,1)$	4	16	15.491	0.905
`Caprasse`	$(2,-i\sqrt{3},2,i\sqrt{3})$	2	4	.249	.219
`Cyclic 9`	Z_9	2	4	19.344	0.905
`DZ1`	$(0,0,0,0)$	10	131	642.287	11.731
`DZ2`	$(0,0,-1)$	7	16	4.852	0.421
`DZ4`	$(0,0,0)$	4	5	0.468	.125

TABLE 1. Test results on benchmark problems (cf. [**5**]).

The objective of developing the new method in this paper is to advance the computational efficiency for large systems. For this purpose, we propose three additional benchmark problems for performance tests in this paper and the future development of high efficiency algorithms for computing multiplicity structures. In all three problems, the sequence $\{p_1, p_2, p_3, \ldots\}$ denotes of the sequence of prime numbers $\{2, 3, 5, \ldots\}$.

- *The shifted KSS systems* of $n \times n$

$$(x_1 - \sqrt{p_1}) + \cdots + (x_n - \sqrt{p_n}) + (x_k - \sqrt{p_k})^2 \;=\; 0, \quad \text{for } k = 1, 2, \ldots, n \tag{7.1}$$

  at the zero $(\sqrt{p_1}, \ldots, \sqrt{p_n})$, which is a variation of the KSS system (1.2) [**7**] in the introduction. The multiplicity increases from 4 to 256 as does n from 3 to 9 and the memory demand intensifies even further.

- *The cyclic cubic systems* of $n \times n$ defined as

$$\begin{cases} u_k^3 - u_{k+1}\, u_{k+2} \;=\; 0 \quad \text{for } k = 1, 2, \ldots, n-2, \;\; \text{and} \\ u_{n\text{-}1}^3 - u_n u_1 = 0, \;\; u_n^3 - u_1 u_2 = 0, \;\; \text{with } u_j = \sqrt{p_j}\, x_j \text{ for } j = 1, \ldots, n \end{cases} \tag{7.2}$$

  at the zero $(x_1, \ldots, x_n) = (0, \ldots, 0)$. This sequence of polynomial systems can be considered a generalization and variation of the benchmark problem `cbms1` [**5, 16**]. The multiplicities of the cyclic cubic systems increase rapidly, and the memory is only enough for the previous code up to $n = 5$.

system size n	3	4	5	6	7	8	9
multiplicity, depth	4, 2	11, 4	16, 4	42, 6	64, 6	163, 8	256, 8
previous time	0.06	3.15	15.07	1231.4	—	—	—
Gröbner basis time	0.09	0.92	20.20	603.1	23984.0	—	—
new time	0.11	0.45	0.98	16.0	47.1	1894.1	10850.2

TABLE 2. Timing comparison on $n \times n$ shifted KSS systems (7.1). The previous code runs out of memory for $n = 7, 8, 9$. The Gröbner basis computation was terminated for 8 and 9 due to excessive time.

- *The ten-fold triangle system* of $n \times n$

$$\begin{cases} v_1^5 + v_2^2 + v_3 + \cdots + v_n &= 0 \\ v_2^2 + v_3 + \cdots + v_n &= 0 \\ v_3 + \cdots + v_n &= 0 \\ \ddots \quad \vdots & \vdots \\ v_n &= 0 \end{cases} \qquad \begin{array}{c} \text{with } v_j = x_j - \sqrt{\frac{p_i}{p_n}} \\ \text{for } j = 1, \ldots, n \end{array} \tag{7.3}$$

  at the zero $(x_1, x_2, \ldots, x_n) = \left(\sqrt{\frac{p_1}{p_n}}, \sqrt{\frac{p_2}{p_n}}, \ldots, \sqrt{\frac{p_n}{p_n}}\right)$ has a fixed multiplicity 10 and depth 5 for all $n \geq 2$. The problem size increases in terms of the number of equations and variables.

As n increases, those polynomial sequences increase the demand on the memory requirement and computing time. For instance, the largest Macaulay matrix required for the KSS system of 7×7 is 12012×3432 and the previous code runs out of memory. To put the memory demand in perspective, the largest Macaulay matrix required for the 9×9 KSS system is 218790×48620 or at least 79 GB of memory is needed for the previous method, not to mention the huge amount of computing time on manipulating such a large matrix even if the memory were available.

It is possible to compute multiplicities via Gröbner bases in symbolic computation at the exact zeros. However, round-off errors can not be avoided in the three systems or the zeros using floating point arithmetic, making symbolic methods such as the Gröbner basis unsuitable. For comparison, we also lists the times required for Maple to compute the Gröbner bases on the systems using exact coefficients.

The results in Table 2 show the closedness subspace method is up to 77 times faster than the previous algorithm on KSS systems for $n = 6$, and continues to produce multiplicity structures for $n = 7, 8, 9$ long after the previous code runs out of memory. The result in Table 3 is similar for the cyclic cubic systems on which the new code is up to 179 times faster when both codes can still run, while the new code maintains the capability of computation up to $n = 7$ for multiplicity 338.

system size n	3	4	5	6	7
multiplicity, depth	11, 4	30, 6	62, 7	153, 9	338, 10
previous time	0.453	28.75	629.6	—	—
Gröbner basis time	0.124	2.93	90.9	5033.6	—
new time	0.281	0.91	3.5	42.5	847.5

TABLE 3. Timing comparison on $n \times n$ cyclic cubic systems (7.2). The previous code runs out of memory for $n = 6, 7$. The Gröbner basis computation was terminated for $n = 7$ due to excessive time.

The results in Table 4 on the ten-fold triangle systems give further insight on the advantage of the new method. For the closedness subspace method, the computing time does not seem to increase as much as for methods relying on the Macaulay matrices when the number of indeterminates increases while the multiplicity stay the same. The improvement ratio reaches the level of thousands for 7-variate systems. Again, the previous code runs out of memory at $n = 8$ while the new code continues to run for $n = 100$ when the test stops. This test result show that the cost of the closedness subspace method in computing multiplicity may be practically negligible on large polynomial systems when the multiplicity is low, considering the high cost of solving those systems in the first place.

system size n	5	6	7	8	9	10	$\cdots$	100
multiplicity, depth	identical multiplicity 10 and depth 5 for all n							
previous time	54.21	268.9	1113.7	—	—	—	$\cdots$	—
Gröbner basis time	0.375	6.084	126.3	3100.7	—	—	$\cdots$	—
new time	0.234	0.296	0.312	0.343	0.359	0.406	$\cdots$	56.1

TABLE 4. Timing comparison on $n \times n$ ten-fold triangle systems (7.3). The previous code runs out of memory for $n \geq 8$. The Gröbner basis computation was terminated for $n \geq 9$ due to excessive time.

The experimental results on the test systems clearly show the superiority of the closedness subspace method over its predecessor in both execution time and memory consumption. The new method takes advantage of the low dimension of the closedness subspaces and the resulting smaller matrix sizes in the process of computing dual bases, making it possible to improve efficiency substantially and to solve larger problems.

Concluding remark. When the degree and the number of indeterminates increase, the number of monomials grow rapidly to a prohibitive magnitude. As a result, the vector spaces spanned by straightforward monomial bases quickly become difficult to handle in matrix computation due to high dimensions. A possible solution to overcome this bottleneck is to use proper low dimensional subspaces as domains on which linear transformations correspond to matrices of small sizes. The closedness subspace method in this paper demonstrates the effectiveness of this dimension reduction strategy. Other polynomial computation problems such as multivariate GCD can also benefit tremendously from similar techniques[**21**] using the so-called fewnomial subspaces. Further exploration of subspace methods may lead to efficient algorithms for more applications.

Acknowledgement. The author would like thank Hans Stetter for many insightful discussions and providing the diploma thesis [**17**] of his former student.

References

[1] D. BATES, J. D. HAUENSTEIN, , C. PETERSON, AND A. J. SOMMESE, *A local dimension test for numerically approximated points on algebraic sets*, Preprint, 2008.

[2] D. BATES, J. D. HAUENSTEIN, A. J. SOMMESE, AND C. W. WAMPLER, *Bertini: Software for Numerical Algebraic Geometry*. http://www.nd.edu/∼sommese/bertini, 2006.

[3] ———, *Software for numerical algebraic geometry: A paradigm and progress towards its implementation*, in Software for Algebraic Geometry, IMA Volume 148, M. Stillman, N. Takayama, and J. Verschelde, eds., Springer, 2008, pp. 1–14.

[4] D. J. Bates, C. Peterson, and A. J. Sommese, *A numerical-symbolic algorithm for computing the multiplicity of a component of an algebraic set*, J. of Complexity, 22 (2006), pp. 475–489.
[5] B. Dayton and Z. Zeng, *Computing the multiplicity structure in solving polynomial systems*. Proceedings of ISSAC '05, ACM Press, pp 116–123, 2005.
[6] W. Fulton, *Intersection Theory*, Springer Verlag, Berlin, 1984.
[7] H. Kobayashi, H. Suzuki, and Y. Sakai, *Numerical calculation of the multiplicity of a solution to algebraic equations*, Math. Comp., 67 (1998), pp. 257–270.
[8] A. Leykin, J. Verschelde, and A. Zhao, *Newton's method with deflation for isolated singularities of polynomial systems*, Theoretical Computer Science, (2006), pp. 111–122.
[9] T. Y. Li and Z. Zeng, *A rank-revealing method with updating, downdating and applications*, SIAM J. Matrix Anal. Appl., 26 (2005), pp. 918–946.
[10] F. S. Macaulay, *The Algebraic Theory of Modular Systems*, Cambridge Univ. Press, 1916.
[11] M. G. Marinari, T. Mora, and H. M. Möller, *Gröbner bases of ideals defined by functionals with an application to ideals of projective points*, J. AAECC, 4 (1993), pp. 103–145.
[12] ———, *On multiplicities in polynomial system solving*, Trans. AMS, 348 (1996), pp. 3283–3321.
[13] T. Mora, *Solving Polyonmial Equation Systems II*, Cambridge Univ. Press, London, 2004.
[14] B. Mourrain, *Isolated points, duality and residues*, J. of Pure and Applied Algebra, 117 & 118 (1996), pp. 469–493. Special issue for the Proc. of the 4th Int. Symp. on Effective Methods in Algebraic Geometry (MEGA).
[15] H. J. Stetter, *Numerical Polynomial Algebra*, SIAM, 2004.
[16] B. Sturmfels, *Solving Systems of Polynomial Equations*, Number 97 in CBMS Regional Conference Series in Mathematics, AMS, 2002.
[17] G. H. Thallinger, *Analysis of Zero Clusters in Multivariate Polynomial Systems*. Diploma Thesis, Tech. Univ. Vienna, 1996.
[18] X. Wu and L. Zhi, *Computing the multiplicity structure from geometric involutive form*. Proc. ISSAC'08, ACM Press, pages 325–332, 2008.
[19] Z. Zeng, *ApaTools: A Maple and Matlab toolbox for approximate polynomial algebra*, in Software for Algebraic Geometry, IMA Volume 148, M. Stillman, N. Takayama, and J. Verschelde, eds., Springer, 2008, pp. 149–167.
[20] Z. Zeng, *ApaTools*, `http://www.neiu.edu/~zzeng/apatools.htm`, 2007.
[21] ———, *Regularization and matrix computation in numerical polynomial algebra*. to appear.

Department of Mathematics, Northeastern Illinois University, Chicago, IL 60625

zzeng@neiu.edu