8-2022

# Identifying Trace Affine Linear Sets Using Homotopy Continuation

Julianne McKay
mckay6@clemson.edu

# Identifying Trace Affine Linear Sets using Homotopy Continuation

---

A Thesis
Presented to
the Graduate School of
Clemson University

---

In Partial Fulfillment
of the Requirements for the Degree
Master of Science
Mathematical Sciences

---

by
Julianne McKay
August 2022

---

Accepted by:
Dr. Michael Burr, Committee Chair
Dr. Ryann Cartor
Dr. Shuhong Gao
Dr. Svetlana Poznanović

# Abstract

We investigate how the coefficients of a sparse polynomial system influence the sum, or the trace, of its solutions. We discuss an extension of the classical trace test in numerical algebraic geometry to sparse polynomial systems. Two known methods for identifying a trace affine linear subset of the support of a sparse polynomial system use sparse resultants and polyhedral geometry, respectively. We introduce a new approach which provides more precise classifications of trace affine linear sets than was previously known. For this new approach, we developed software in *Macaulay2* [9].

# Dedication

To Dad, the first person to tell me that math is beautiful (though I didn't believe you at first).

# Acknowledgments

This work was supported in part by NSF award number 1913119. Special thanks is due to my advisor, Dr. Michael Burr, without whom this project would not exist. I am immensely grateful for his excitement, expertise, and attention to detail throughout the many meetings and drafts. I thank my committee members Dr. Ryann Cartor, Dr. Shuhong Gao, and Dr. Svetlana Poznanović for taking the time to ask thoughtful questions and provide comments on my work. I would also like to thank Taylor Brysiewicz for sharing his knowledge with me and providing helpful suggestions early in the project.

I am grateful for my fellow SMSS graduate students, as none of us would get through this alone. I am particularly grateful to Rodrigo Smith and Michael Byrd, who saw this project in the earliest stages and provided encouragement and suggestions.

Finally, I want to thank my family and friends, whose unfailing love and support has been a great blessing to me throughout my life, but especially during this time.

# Table of Contents

# List of Tables

# List of Figures

# Chapter 1

# Background

Algebraic geometry is the study of geometric objects, such as curves and surfaces, that are defined by polynomial systems and their algebraic properties. As background, we introduce concepts from algebraic geometry, and experts may consider skipping to Chapter 2.

## 1.1 Polynomial Systems

We begin by defining and providing examples of fundamental objects in algebraic geometry. We also establish the notation used throughout the paper. See [6] for a more detailed discussion of these topics.

Let $x_1, x_2, \ldots, x_n$ be a finite collection of variables and $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{N}$ be nonnegative integer exponents. Then $\mathbf{x}^\alpha := x_1^{\alpha_1} x_2^{\alpha_2} \cdots x_n^{\alpha_n}$ is a **monomial** in $x_1, \ldots, x_n$. The tuple $\alpha = (\alpha_1, \alpha_2, \ldots, \alpha_n) \in \mathbb{N}^n$ is called the **multidegree** of the monomial. The **total degree** of a monomial $\mathbf{x}^\alpha$ is the sum of the exponents, and it is denoted by $|\alpha|$. Let $k$ be a field. A **term** $c_\alpha \mathbf{x}^\alpha$ is a monomial multiplied by a coefficient $c_\alpha \in k$.

**Example 1.1.** The product $x_1^2 x_2 x_5^4$ is a monomial in $x_1, \ldots, x_5$ with multidegree $\alpha = (2, 1, 0, 0, 4)$. The total degree is $|\alpha| = 2 + 1 + 0 + 0 + 4 = 7$.

**Definition 1.2.** Given a field $k$, a **polynomial** in $k[x_1, \ldots, x_n]$ is a finite sum:

$$f = \sum_{\alpha \in A} c_\alpha \mathbf{x}^\alpha$$

where $c_\alpha \in k$ for each $\alpha$.

**Definition 1.3.** The set of multidegress of the monomials appearing in a polynomial $f$ is called the **support** of $f$, that is $\text{supp}(f) = \{\alpha : c_\alpha \neq 0\}$. We say that $f$ is **supported on** $A$ if $\text{supp}(f) \subseteq A$.

Note that $A \subset \mathbb{N}^n$ since each multidegree $\alpha$ is an $n$-tuple of positive integers. Here, we let $0 \in \mathbb{N}$. We work in the complex field $k = \mathbb{C}$, so examples will be chosen with coefficients in $\mathbb{C}$.

**Example 1.4.** Consider the polynomial $f = x^2y - 4xy + 3xy^2 + 7y + 3 \in \mathbb{C}[x, y]$. The polynomial $f$ is a sum of five terms, and the support of $f$ is $\{(2, 1), (1, 1), (1, 2), (0, 1), (0, 0)\}$.

**Definition 1.5.** Given a field $k$, a **polynomial system** $\mathcal{F} \in (k[x_1, \ldots, x_n])^m$ is a collection of polynomials $(f_1, f_2, \ldots, f_m)$ where each $f_i \in k[x_1, \ldots, x_n]$. If $n = m$ the system is called **square**.

**Definition 1.6.** Let $\mathcal{F} = (f_1, \ldots, f_m)$ be a polynomial system in $(k[x_1, \ldots, x_n])^m$, where each $f_i$ is supported on $A_i$. The collection $\mathcal{A} = (A_1, \ldots, A_m)$ is called the **support of the system** $\mathcal{F}$, and we say $\mathcal{F}$ is **supported on** $\mathcal{A}$.

The **lattice** $L[A]$ of a support $A \subset \mathbb{Z}^n$ is the Abelian group generated by all differences of points in $A$. Since $L[A]$ is a subgroup of $\mathbb{Z}^n$, $L[A] \simeq \mathbb{Z}^r$ for some integer $r \leq n$. The **rank** of $A$ is this $r$. We say $A \subset \mathbb{Z}^n$ is **full rank** if $A$ has rank $n$. Note that a support $A$ may have rank $n$ while $L[A]$ is a proper subset of $\mathbb{Z}^n$. The **lattice** $L[\mathcal{A}]$ of a collection of supports $\mathcal{A} = (A_1, \ldots, A_n)$ is the group generated by $\cup_{i=1}^n L[A_i]$.

The following definition is a technical one that is needed for completeness in Algorithm 4.

**Definition 1.7.** A support $\mathcal{A} = (A_1, \ldots, A_m)$ of a polynomial system $\mathcal{F} = (f_1, \ldots, f_m)$ is **abundant** if each $A_i$ is full rank.

We would like to go beyond the study of polynomials as algebraic equations. The next definitions introduce the connection between polynomial systems and geometric objects. First, define $k^n = \{(a_1, \ldots, a_n) \mid a_1, \ldots, a_n \in k\}$ to be the $n$**-dimensional affine space** over $k$.

**Definition 1.8.** Let $\mathcal{F} = (f_1, \ldots, f_m)$ be a polynomial system in $(k[x_1, \ldots, x_n])^m$. The set of

simultaneous solutions $(a_1, \ldots, a_n) \in k^n$ to the equations

$$\begin{aligned} f_1(x_1, \ldots, x_n) &= 0 \\ f_2(x_1, \ldots, x_n) &= 0 \\ &\vdots \\ f_m(x_1, \ldots, x_n) &= 0 \end{aligned}$$

is the **affine variety**, or simply **variety**, of $\mathcal{F}$, denoted by $\mathcal{V}(\mathcal{F})$.

When we restrict the solution set of the system $\mathcal{F}$ to its nonzero solutions, we write $\mathcal{V}^\times(\mathcal{F}) = \{(a_1, \ldots, a_n) \in \mathcal{V}^\times(\mathcal{F}) \mid a_i \neq 0 \ \forall i\}$. An **irreducible** variety is a variety such that there is no decomposition $\mathcal{V} = V_1 \cup V_2$ into a union of proper subvarieties.

We define a **family of polynomial systems** to be a parameter space together with a continuous map to the space of polynomials, that is $V \hookrightarrow (\mathbb{C}[\mathbf{x}])^m$. The parameter space $V$ is a subvariety of $\mathbb{C}^m$ for some $m$. The family describes a situation in which the coefficients of the system are dependent upon parameters, typically by an algebraic function. In this paper we consider a family of polynomials with a map which identifies point $\mathbf{c} \in \mathbb{C}^m$ with the system which has $\mathbf{c}$ as some of the coefficients. For a polynomial system in the family supported on $\mathcal{A}$, we write $\mathcal{F} \in \mathbb{C}^{\mathcal{A}}$.

**Example 1.9.** The family of polynomials described by $\mathbb{C}^3 \hookrightarrow \mathbb{C}[x]$ where

$$(a, b, c) \mapsto ax^2 + bx + c$$

is the family of polynomials in one variable of degree at most 2. We could also write this as the family of polynomials supported on $A = \{0, 1, 2\}$.

**Definition 1.10.** A **generic** property of a family of polynomials is a property that is true for polynomials in its parameter space except for those on a proper subvariety. A **generic polynomial** in a family of polynomials is a polynomial that has this generic property.

**Example 1.11.** A generic property of the family of polynomials of degree less than or equal to 2 is the existence of two distinct roots. Note that $ax^2 + bx + c$ has two distinct roots if $a \neq 0$ and $b^2 - 4ac \neq 0$. The requirement that $a \neq 0$ ensures the polynomial has degree 2, and the requirement that the discriminant is nonzero ensures that the roots are distinct and there is not a

double root. The collection of polynomials without two distinct roots is the variety $\{ax^2+bx+c \mid a = 0\} \cup \{ax^2+bx+c \mid b^2-4ac=0\}$, which is a subvariety of the family of polynomials. Therefore the generic property holds on the entire family *except* on this subvariety.

**Definition 1.12.** Given a collection of supports $\mathcal{A}$, the **sparse** family of polynomials supported on $\mathcal{A}$ is the collection of all polynomial systems supported on $\mathcal{A}$. A **dense** family of polynomials is one limited by the degrees of the polynomials. The support of a dense family is $A = \{\alpha \ : \ |\alpha| \leq d\}$, where $d$ is the total degree of the polynomial.

**Remark 1.13.** Dense families are sparse since fixing the total degree of the polynomial defines the support. However, sparse families are not always dense because, given support $A$ of some sparse family, we may have $A \neq \{\alpha \ : \ |\alpha| \leq d\}$ for every degree $d$.

**Example 1.14.** Consider the dense family of polynomials described by $\mathbb{C}^6 \hookrightarrow \mathbb{C}[x,y]$, where

$$(a,b,c,d,e,f) \mapsto a+bx+cy+dx^2+exy+fy^2.$$

This family consists of polynomials of degree less than or equal to 2 in two variables. The support for a generic polynomial in this family is $A = \{(0,0),(1,0),(0,1),(2,0),(1,1),(0,2)\}$.

If we remove $(1,0)$ and $(0,2)$ from the support $A$, we have a sparse family of polynomials where the support of a generic polynomial is $B = \{(0,0),(0,1),(2,0),(1,1)\}$, but this family is not dense. Additionally, the parametrization

$$(a,c,d,e) \mapsto a+cy+dx^2+exy$$

shows that the parameter space of this sparse family is $\mathbb{C}^4$.

Given a set $A \subset \mathbb{Z}^n$, we replace each coefficient $c_\alpha \in \mathbb{C}$ from Definition 1.2 with a parameter $u_\alpha$. We say that the *family* $f(u)$ is the universal polynomial supported on the set $A$. The universal polynomial has coefficients which are parameters, but unlike the family of polynomial systems, the universal polynomial does not include a mapping. We extend this notation to a system.

**Definition 1.15.** Given a set of supports $\mathcal{A} = (A_1, \ldots, A_n)$, the **universal system** $\mathcal{F}(u)$ supported on $\mathcal{A}$ is the system $\mathcal{F}(u) = (f_1(u), \ldots, f_n(u))$, where each $f_i(u)$ is the universal polynomial supported on $A_i$.

Let $\mathcal{A} = (A_1, \ldots, A_m)$ be a set of supports where each $A_i \subset \mathbb{Z}^n$. Then the universal system $\mathcal{F}(u)$ supported on $\mathcal{A}$ is a system whose polynomials are $f_i = \sum_{\alpha_i \in A_i} u_{\alpha_i} \mathbf{x}^{\alpha_i}$. Each $f_i$ is from the family $f_i(u)$ supported on $A_i$.

## 1.2 Resultants and Sparse Resultants

In algebraic geometry, one use of the *resultant* is to detect whether overdetermined polynomial systems share solutions. The resultant of two polynomials in one variable can be computed by most computer algebra systems, and it equals zero if and only if the two polynomials have a common zero. The resultant can be extended to *sparse* systems. We follow [6] and the interested reader can find more detail there, especially in Chapters 3 and 7. We begin with a discussion and examples for two polynomials.

**Definition 1.16.** Let $f, g \in k[x]$ be two nonzero polynomials of degrees $n$ and $m$, respectively, where $f = a_n x^n + \cdots + a_1 x + a_0$ and $g = b_m x^m + \cdots + b_1 x + b_0$ such that $a_n, b_m \neq 0$. The **resultant of the system** $\mathcal{F} = (f, g)$ is denoted by $\mathrm{Res}(\mathcal{F})$ or $\mathrm{Res}(f, g)$, and is defined to be an integer polynomial in the coefficients of the polynomials $f, g$ such that $\mathrm{Res}(f, g) = 0$ if and only if $f$ and $g$ share some common solution.

One way to evaluate the resultant of two polynomials in one variable is by using the *Sylvester resultant*:

**Definition 1.17.** The **Sylvester resultant** is the determinant of the $(n+m) \times (n+m)$ coefficient matrix of the system:

$$\mathrm{Res}(f,g) = \det \begin{pmatrix}
a_n & & & & b_m & & & \\
a_{n-1} & a_n & & & b_{m-1} & b_m & & \\
a_{n-2} & a_{n-1} & \ddots & & b_{m-2} & b_{m-1} & \ddots & \\
\vdots & a_{n-2} & \ddots & a_n & \vdots & b_{m-2} & \ddots & b_m \\
a_0 & \vdots & \ddots & a_{n-1} & b_0 & \vdots & \ddots & b_{m-1} \\
 & a_0 & & a_{n-2} & & b_0 & & b_{m-2} \\
 & & \ddots & \vdots & & & \ddots & \vdots \\
 & & & a_0 & & & & b_0
\end{pmatrix}$$

The coefficients of $f$ are repeated $\deg(g) = m$-many times. Similarly, the column of coefficients from $g$ is repeated $\deg(f) = n$-many times.

**Example 1.18.** Consider the following two polynomials in one variable $f, g \in \mathbb{C}[x]$:

$$f = x^2 - 6x + 3 \text{ and } g = -x^3 + 5x^2 + 3x - 3$$

The resultant of the system $\mathcal{F} = (f, g)$ is:

$$\text{Res}(x^2 - 6x + 3, -x^3 + 5x^2 + 3x - 3) = \det \begin{pmatrix} 1 & 0 & 0 & -1 & 0 \\ -6 & 1 & 0 & 5 & -1 \\ 3 & -6 & 1 & 3 & 5 \\ 0 & 3 & -6 & -3 & 3 \\ 0 & 0 & 3 & 0 & -3 \end{pmatrix} = 0$$

Since the resultant equals zero, by definition, the varieties defined by these two polynomials share at least one solution. Although the resultant does not find the solution, it is not difficult to solve the system in this case. One way is to compute the greatest common divisor of the two polynomials. In this case, the greatest common divisor is $f$ itself, as we observe that $g$ is divisible by $f$:

$$-x^3 + 5x^2 + 3x - 3 = (-x - 1)(x^2 - 6x + 3).$$

The two roots of $f = x^2 - 6x + 3$ are $x = 3 \pm \sqrt{6}$ by the quadratic formula. Since $g$ is divisible by $f$, these are also roots of $g$.

In general, we can extend the definition of the resultant to systems of $n + 1$ polynomials in $n$ variables. The resultant of a system $\mathcal{F} = (f_1, \ldots, f_{n+1})$ in $n$ variables, as expected from Definition 1.16, is an integer polynomial in the coefficients of the polynomials $f_i$, such that the resultant equals 0 if and only if the $f_i$ share at least one common solution. The resultant of $\mathcal{F}$ equals 0 if and only if the variety $\mathcal{V}(\mathcal{F})$ is nonempty.

Additionally, we extend the definition of the resultant to universal systems. Let $\mathcal{A}$ be a collection of supports such that $\mathcal{F} \in \mathbb{C}^{\mathcal{A}}$ is a system with $n + 1$ polynomials in $n$ variables. Then the universal system $\mathcal{F}(u)$ supported on $\mathcal{A}$ has a resultant polynomial, defined as follows:

**Definition 1.19.** The **resultant polynomial of the universal system**, $\text{Res}(\mathcal{F}(u))$, is a polyno-

mial in the parameters $u_{\alpha_i}$, [6, Section 3.5]. Generic points in the variety of $\mathrm{Res}(\mathcal{F}(u))$ correspond to systems $\mathcal{F}$ which have a common affine solution.

**Example 1.20.** Let $\mathcal{A} = (A_1, A_2)$ where $A_1 = \{3, 2, 1, 0\}$ and $A_2 = \{3, 2, 1, 0\}$. Let $\mathcal{F}(u)$ be the universal system supported on $\mathcal{A}$. Then $\mathcal{F}(u)$ is a system of two generic polynomials $f_1(u)$ and $f_2(u)$. We denote the parameter $u_{i,j}$ to be the coefficient in $f_i(u)$ associated with the monomial $x^j$, so that

$$f_1(u) = u_{1,3}x^3 + u_{1,2}x^2 + u_{1,1}x + u_{1,0} \qquad \text{and} \qquad f_2(u) = u_{2,3}x^3 + u_{2,2}x^2 + u_{2,1}x + u_{2,0}.$$

The resultant of the universal system, using the Sylvester resultant from Definition 1.17, is:

$$\mathrm{Res}(\mathcal{F}(u)) = \det \begin{pmatrix} u_{1,3} & & & u_{2,3} & & \\ u_{1,2} & u_{1,3} & & u_{2,2} & u_{2,3} & \\ u_{1,1} & u_{1,2} & u_{1,3} & u_{2,1} & u_{2,2} & u_{2,3} \\ u_{1,0} & u_{1,1} & u_{1,2} & u_{2,0} & u_{2,1} & u_{2,2} \\ & u_{1,0} & u_{1,1} & & u_{2,0} & u_{2,1} \\ & & u_{1,0} & & & u_{2,0} \end{pmatrix} =$$

$$\begin{aligned} & u_{1,3}^3u_{2,0}^3 - u_{1,2}u_{1,3}^2u_{2,0}^2u_{2,1} + u_{1,1}u_{1,3}^2u_{2,0}u_{2,1}^2 - u_{1,0}u_{1,3}^2u_{2,1}^3 + u_{1,2}^2u_{1,3}u_{2,0}^2u_{2,2} \\ & - u_{1,1}u_{1,2}u_{1,3}u_{2,0}u_{2,1}u_{2,2} + 3u_{1,0}u_{1,3}^2u_{2,0}u_{2,1}u_{2,2} + u_{1,0}u_{1,2}u_{1,3}u_{2,1}^2u_{2,2} \\ & + u_{1,1}^2u_{1,3}u_{2,0}u_{2,2}^2 - 2u_{1,0}u_{1,2}u_{1,3}u_{2,0}u_{2,2}^2 - u_{1,0}u_{1,1}u_{1,3}u_{2,1}u_{2,2}^2 + u_{1,0}^2u_{1,3}u_{2,2}^3 \\ & - u_{1,2}^3u_{2,0}^2u_{2,3} + 3u_{1,1}u_{1,2}u_{1,3}u_{2,0}^2u_{2,3} - 3u_{1,0}u_{1,3}^2u_{2,0}^2u_{2,3} + u_{1,1}u_{1,2}^2u_{2,0}u_{2,1}u_{2,3} \\ & - 2u_{1,1}^2u_{1,3}u_{2,0}u_{2,1}u_{2,3} - u_{1,0}u_{1,2}u_{1,3}u_{2,0}u_{2,1}u_{2,3} - u_{1,0}u_{1,2}^2u_{2,1}^2u_{2,3} + 2u_{1,0}u_{1,1}u_{1,3}u_{2,1}^2u_{2,3} \\ & - u_{1,1}^2u_{1,2}u_{2,0}u_{2,2}u_{2,3} + 2u_{1,0}u_{1,2}^2u_{2,0}u_{2,2}u_{2,3} + u_{1,0}u_{1,1}u_{1,3}u_{2,0}u_{2,2}u_{2,3} \\ & + u_{1,0}u_{1,1}u_{1,2}u_{2,1}u_{2,2}u_{2,3} - 3u_{1,0}^2u_{1,3}u_{2,1}u_{2,2}u_{2,3} - u_{1,0}^2u_{1,2}u_{2,2}^2u_{2,3} \\ & + u_{1,1}^3u_{2,0}u_{2,3}^2 - 3u_{1,0}u_{1,1}u_{1,2}u_{2,0}u_{2,3}^2 + 3u_{1,0}^2u_{1,3}u_{2,0}u_{2,3}^2 - u_{1,0}u_{1,1}^2u_{2,1}u_{2,3}^2 \\ & + 2u_{1,0}^2u_{1,2}u_{2,1}u_{2,3}^2 + u_{1,0}^2u_{1,1}u_{2,2}u_{2,3}^2 - u_{1,0}^3u_{2,3}^3 \end{aligned} \quad (1.1)$$

We note that the resultant is a polynomial in the eight variables $u_{1,3}, u_{1,2}, u_{1,1}, u_{1,0}, u_{2,3}, u_{2,2}, u_{2,1}, u_{2,0}$. A point $(a_3, a_2, a_1, a_0, b_3, b_2, b_1, b_0) \in \mathbb{C}^8$ is a solution of the resultant when setting $u_{1,3} = a_3, u_{1,2} = a_2, \ldots, u_{2,0} = b_0$ makes the resultant equal to 0. The set of solution points to the resultant

polynomial is the affine variety $\mathcal{V}(\mathrm{Res}(\mathcal{F}(u)))$. Geometrically, the affine variety of the resultant is a hypersurface in 8-dimensional space. Thus, a generic point $(a_3, a_2, a_1, a_0, b_3, b_2, b_1, b_0)$ in the variety $\mathcal{V}$ corresponds to a system supported on $\mathcal{A}$ with coefficients $a_i, b_i$ which has a solution.

For some sparse families of polynomials, however, obtaining a zero resultant may not guarantee that the system has a nonzero common solution. For instance, it is possible that a system within the sparse family has support specialized enough from the overall support that the resultant polynomial is the zero polynomial.

**Example 1.21.** The resultant in Equation (1.1) has at least one of $u_{1,0}$ or $u_{2,0}$ as a factor in every term. Consider the family of sparse systems supported on $\mathcal{A} = (\{3, 2, 1\}, \{3, 2, 1\})$. This is equivalent to setting $u_{1,0} = 0$ and $u_{2,0} = 0$ in the system in Example 1.20. Then, the resultant polynomial of this universal system is the zero polynomial, which means that a system in this sparse family with any coefficients always has a common solution. Clearly the polynomials $f_1 = a_3x^3 + a_2x^2 + a_1x$ and $f_2 = b_3x^3 + b_2x^2 + b_1x$ have a common solution at $x = 0$ for any coefficients $a_3, \ldots, b_1 \in \mathbb{C}$. But there may not be any nontrivial solutions.

Consider the systems where $(a_3, a_2, a_1, a_0) = (1, 1, -5, 0)$ and $(b_3, b_2, b_1, b_0) = (3, 1, 1, 0)$. We have $f_1 = x^3 + x^2 - 5x$ and $f_2 = 3x^3 + x^2 + x$, which have a common zero at $(0, 0)$, as expected by the resultant. However, there are no nontrivial solutions. Table 1.1 lists the solutions of $f_1$ and $f_2$. This indicates that we need a more sensitive version of the resultant which is not the zero polynomial.

| Solutions to $f_1$ | Solutions to $f_2$ |
|---|---|
| 1.791 | $-0.167 + 0.553i$ |
| 0 | 0 |
| $-2.791$ | $-0.167 - 0.553i$ |

Table 1.1: Table of solutions to polynomials $f_1$ and $f_2$. The only common solution is the trivial solution, $x = 0$.

The sparseness of the family combined with the resultant, as we have defined it, is not enough to decide whether there are systems with nontrivial solutions in this family. There is a generalization of the resultant to deal with sparse families. We follow [6, Section 7.2].

**Definition 1.22.** The **sparse resultant** of the universal system $\mathcal{F}(u)$ over its support $\mathcal{A}$, denoted by $\mathrm{Res}_{\mathcal{A}}(\mathcal{F}(u))$, is an irreducible polynomial in the coefficients of $\mathcal{F}(u)$ whose solutions are generically the coefficients $c$ where the system $\mathcal{F}(c)$ has a solution in $(\mathbb{C}^{\times})^n$.

**Example 1.23.** The sparse resultant for the system supported on $\mathcal{A} = (\{3,2,1\},\{3,2,1\})$ in Example 1.21 is $\mathrm{Res}_{\mathcal{A}}(f_1(u), f_2(u)) = \mathrm{Res}(f_1(u)/x, f_2(u)/x)$, where Res is the standard resultant. Dividing each of $f_1(u)$ and $f_2(u)$ by $x$ decreases the total degree of the system by one, and gets rid of the solution at $x = 0$. Then we compute the resultant of $(f_1(u)/x, f_2(u)/x)$ using the Sylvester resultant:

$$\mathrm{Res}_{\mathcal{A}}(\mathcal{F}(u)) = \det \begin{pmatrix} u_{1,3} & & u_{2,3} & \\ u_{1,2} & u_{1,3} & u_{2,2} & u_{2,3} \\ u_{1,1} & u_{1,2} & u_{2,1} & u_{2,2} \\ & u_{1,1} & & u_{2,1} \end{pmatrix} =$$

$$u_{1,3}^2 u_{2,1}^2 - u_{1,2}u_{1,3}u_{2,1}u_{2,2} + u_{1,1}u_{1,3}u_{2,2}^2 + u_{1,2}^2 u_{2,1}u_{2,3}$$
$$- 2u_{1,1}u_{1,3}u_{2,1}u_{2,3} - u_{1,1}u_{1,2}u_{2,2}u_{2,3} + u_{1,1}^2 u_{2,3} \quad (1.2)$$

A generic solution $(a_3, a_2, a_1, 0, b_3, b_2, b_1, 0) \in \mathbb{C}^8$ to the sparse resultant corresponds to a system which has a nontrivial common solution.

**Definition 1.24.** Given a subset $S \subset \mathbb{C}^n$, the smallest affine variety in $\mathbb{C}^n$ containing $S$ is the **Zariski closure** of $S$.

Consider the family of polynomial systems of $n+1$ equations in $n$ variables supported on $\mathcal{A}$. Let $S$ be the subset of the parameter space of this family which maps to systems with common solutions. The Zariski closure of $S$ is the variety of the sparse resultant of the family, see [8, Chapter 8].

**Example 1.25.** Consider the family of polynomial systems $(ax + b, cx + d)$. The parameter space for this family is $\mathbb{C}^4$. The sparse resultant of this family is $\mathrm{Res} = ad - bc$. Note that the points $(0, b, 0, d)$ are solutions to the sparse resultant. Geometrically, the points $(0, b, 0, d)$ correspond to a system of two constant functions at $b$ and $d$.

Let $S$ be the subset of $\mathbb{C}^4$ which contains points $(a, b, c, d)$ that map to systems $(ax+b, cx+d)$ which have at least one common solution. Setting both $a = 0$ and $c = 0$ gives the system $(b, d)$, which generically does not have common solutions, since we do not expect $b = d = 0$, in general. That is, $(0, b, 0, d) \notin S$. However, $(0, b, 0, d)$ is in the Zariski closure of $S$, which corresponds to these points being in the variety of the sparse resultant.

## 1.3 Homotopy Continuation

Homotopy continuation is a standard computational method applied to polynomial systems. In algebraic geometry, homotopy continuation is an approach best suited for finding common solutions of a square polynomial system $\mathcal{F} = (f_1, f_2, \ldots, f_n) \in (\mathbb{C}[x_1, \ldots, x_n])^n$. The approach involves defining a function, called a *homotopy*, which deforms the system $\mathcal{F}$ to a simpler system. Numerical continuation methods are used to track the solutions of the simpler system to solutions of $\mathcal{F}$. See [2], Chapter 2 for details beyond those presented here.

**Definition 1.26.** Let $\mathcal{F}$ and $\mathcal{G}$ be polynomial systems in $(\mathbb{C}[\mathbf{x}])^m$. A **homotopy** between $\mathcal{F}$ and $\mathcal{G}$ is a continuous function $H(\mathbf{x}, t) : \mathbb{C}^n \times [0, 1] \to \mathbb{C}^n$ such that $H(\mathbf{x}, 0) = \mathcal{F}(\mathbf{x})$ and $H(\mathbf{x}, 1) = \mathcal{G}(\mathbf{x})$ for all $\mathbf{x} \in \mathbb{C}^n$. We call $\mathcal{F}$ the **target system**, and the system $\mathcal{G}$ is the **start system**. Often a system $\mathcal{G}$ is chosen to have known solutions.

The parameter $t$ is introduced for the homotopy. As $t$ moves from 1 to 0, the system $\mathcal{G}(\mathbf{x})$ is deformed to $\mathcal{F}(\mathbf{x})$. We require $H(\mathbf{x}, t)$ to be a continuous function of $\mathbf{x}$ and $t$ together so that the deformation from $\mathcal{G}$ to $\mathcal{F}$ is continuous. For generic $\hat{\mathbf{x}}$, $H(\hat{\mathbf{x}}, t)$ is a continuous function in $t$ which can be thought of as a path from the point $\mathcal{G}(\hat{\mathbf{x}})$ to the point $\mathcal{F}(\hat{\mathbf{x}})$. The next definition describes a type of basic homotopy which deforms between systems affine linearly.

**Definition 1.27.** Let $\mathcal{G}(\mathbf{x})$ be the target system and $\mathcal{F}(\mathbf{x})$ be the start system. The **straight-line homotopy** between $\mathcal{F}$ and $\mathcal{G}$ is $H(\mathbf{x}, t) := t\mathcal{F}(\mathbf{x}) + (1 - t)\mathcal{G}(\mathbf{x})$.

**Example 1.28.** Consider the system $\mathcal{F} = (f, g) \in (\mathbb{C}[x, y])^2$ where:

$$f = x^2y - 2xy + 3y - 4 \qquad \text{and} \qquad g = xy^2 - 2x^2 + 2y + 3$$

Also consider the system $\mathcal{G} = (f', g') \in (\mathbb{C}[x, y])^2$ where:

$$f' = x^2y - 15xy + 3y - 4 \qquad \text{and} \qquad g' = xy^2 - 2x^2 + 15y + 3$$

We wish to track the solutions of $\mathcal{F}$ to solutions of $\mathcal{G}$. We define a straight-line homotopy between $\mathcal{F}$ and $\mathcal{G}$ to be $H(x, y; t) = t\mathcal{F}(x, y) + (1 - t)\mathcal{G}(x, y)$, so that we get the desired systems as $t$ goes

from 1 to 0.

$$H(x,y;t) = \begin{cases} f_t & = x^2y - (15-13t)xy + 3y - 4 \\ g_t & = xy^2 - 2x^2 + (15-13t)y + 3 \end{cases}$$

The second part of homotopy continuation is *path tracking*. Once we have defined the homotopy $H$ between systems $\mathcal{F}$ and $\mathcal{G}$, we use *predictor-corrector methods* to track the solutions to $\mathcal{G}$ to the solutions to $\mathcal{F}$. In Table 1.2, we see that six solution paths were tracked, but only five were successfully completed. It is possible for a solution path tracking to fail due to a singularity or divergence to infinity. Given some solution $\mathbf{x}_1$ to $\mathcal{G}$, the function $\mathbf{x}_1(t)$ is a solution path. Then $H(\mathbf{x}_1(t),t) = 0$ for all $t$, since the homotopy is continuous. Taking the implicit derivative with respect to $t$, we see that

$$JH(\mathbf{x}_1(t),t) \cdot \mathbf{x}_1'(t) + \frac{\partial}{\partial t}(H(\mathbf{x}_1(t),t)) = 0,$$

where $J$ denotes the Jacobian. Given that the path begins at $t = 1$ with the initial solution $\mathbf{x}_1$, we have the following initial value problem:

$$\mathbf{x}_1'(t) = -[JH(\mathbf{x}(t),t)]^{-1}\frac{\partial}{\partial t}(H(\mathbf{x}_1(t),t)) \qquad \text{with} \quad \mathbf{x}_1(1) = \mathbf{x}_1.$$

To solve this initial value problem, we choose a predictor and path $\mathbf{x}(t)$, beginning at $t = 1$, with initial value $\mathbf{x}(1) = p_0$. The computation constructs a decreasing sequence of $t_i$'s from 1 to 0, and each point along the path $\mathbf{x}(t_i)$ is approximated by $p_i$. See [2, Section 2.2] for more details.

A simple predictor method is Euler's method, which computes approximations:

$$p_{i+1} = p_i - JH(p_i,t_i)^{-1}\frac{\partial H(p_i,t_i)}{\partial t}\Delta t_i$$

with $\Delta t_i = t_{i+1} - t_i$. Geometrically, Euler's method predicts the next point $p_i$ by drawing a line tangent to the solution path $H(\mathbf{x},t)$ at $p_{i-1}$, and then following the line a distance of $\Delta t$.

Since we are tracking solution paths, we want each $p_i \approx \mathbf{x}(t_i)$ to be as close as possible to satisfying $H(p_i,t) = 0$ for all $t$. The predictions resulting from Euler's method can be made more accurate by following each prediction with a correction. A simple corrector to employ is Newton's

method. Starting with $p_i$, Newton's method uses the iteration:

$$\hat{p_i} = p_i - [JH(p_i, t_{i+1})]^{-1} H(p_i, t_i)$$

According to [2], one or two iterations of Newton's method typically gives a much more accurate prediction of $\mathbf{x}(t_i)$. In the predictor-corrector method, the predicted value $p_i$ is replaced with the corrected value $\hat{p_i}$ before the next predict-correct cycle is initiated.

We use the program *Bertini* [1], to solve systems and construct homotopies. There are other software packages which could be used for these same purposes, which include the *Julia* package *HomotopyContinuation.jl* [4], the *Macaulay2* package *NumericalAlgebraicGeometry* [13], and the packages *PHCpack* [17] and *HOM4PS-2.0* [12].

**Example 1.29.** We continue Example 1.28. The tracked solutions of the homotopy $H$ from $\mathcal{F}$ to $\mathcal{G}$ are shown in Table 1.2. We used *Bertini*, as previously mentioned, to track the solutions. There were six initial solutions tracked but only five paths were tracked successfully. Path tracking may fail due to a singularity in the system, or divergence to infinity. In this case, we can only expect five solutions from $\mathcal{G}$ since there is a divergence to infinity.

| Solutions to $\mathcal{F}$ | Tracked solutions to $\mathcal{G}$ |
|---|---|
| (2.09486, 1.2505) | (14.8524, 4.95011) |
| (1.192+1.91039i, -2.05504-.934788i) | (.206525, -72.4356) |
| (-1.35625, .529666) | (-1.5843, .136638) |
| (.438696+1.25604i, 1.16496+2.22755i) | (.88717+.810399i, -.186351+.196254i) |
| (.438696-1.25604i, 1.16496-2.22755i) | (.88717-.810399i, -.186351-.196254i) |
| (1.192-1.91039i, -2.05504+.934788i) | |

Table 1.2: Solutions of polynomial system $\mathcal{F}$ are tracked to solutions of target system $\mathcal{G}$, via a straight-line homotopy. Values in the table were computed using the *Bertini* package [1].

# Chapter 2

# Traces

We introduce traces in numerical algebraic geometry. Considering the trace as a function, its linearity or nonlinearity is a check for completeness of positive-dimensional algebraic varieties. The classical trace test verifies completeness of algebraic varieties using the trace. We review the sparse trace test from [5], which similarly uses linearity of the trace to verify completeness in the zero-dimensional case. The sparse trace test requires a subset of the support characterized by a linear effect on the trace. Using hidden variable sparse resultants, we completely classify elements of the support of a polynomial system by their effect on the trace. We discuss an alternate method of classification which involves polyhedral geometry.

**Definition 2.1.** Let $S \subseteq \mathbb{C}^n$ be a set of points $c = (c_1, \ldots, c_n) \in S$. The **trace in the** $i^{th}$ **coordinate** of $S$ is the sum $\Sigma_i(S) := \sum_{c \in S} c_i$. The **trace** of $S$ is $\Sigma(S) := (\Sigma_1(S), \ldots, \Sigma_n(S))$, the coordinate-wise sum of the points of $S$.

Let $\mathcal{F} = (f_1, \ldots, f_n) \in (\mathbb{C}[x_1, \ldots, x_n])^n$ be a square polynomial system. The *trace in the first coordinate* of the variety of $\mathcal{F}$ is the trace $\Sigma_1(\mathcal{V}^{\times}(\mathcal{F}))$. It is the sum of $x_1$-values of points which appear in the nonzero solutions to $\mathcal{F}$. The following results are stated in terms of the trace in the first coordinate, but they extend to traces in other coordinates by symmetry.

## 2.1 Classical Trace Test

In the study of varieties in algebraic geometry, the trace is used to check if a subset of an affine slice of a positive-dimensional variety contains the entire slice. The ideas presented here are

due to [15], and we follow [14] to write the Classical Trace Test in Lemma 2.4.

**Definition 2.2.** Let $X$ be a variety. The **dimension** of $X$ is the number of generic hyperplanes needed such that their intersection with $X$ is a nonzero, but finite, number of points. The **codimension** of $X$ is the difference between the dimension of the ambient space containing $X$ and the dimension of $X$.

**Example 2.3.** Let $X \subseteq \mathbb{R}^2$ be a curve. In $\mathbb{R}^2$, a hyperplane is simply a line. A generic line $\ell \subseteq \mathbb{R}^2$ intersects $X$ in at least one point. Therefore only one hyperplane is needed to intersect $X$ in a nonzero set of points, so we say that $X$ has dimension 1. If $X$ is a surface in $\mathbb{R}^3$, intersecting with one hyperplane results in a curve, and intersecting with a second hyperplane results in a set of points. Thus a surface in $\mathbb{R}^3$ has dimension 2.

Let $L$ denote the affine space such that $X \cap L$ is a set of points. Then $L$ has dimension equal to the codimension of $X$. A subset of a variety $V \subseteq X \cap L$ is **complete** if $V = X \cap L$.

**Lemma 2.4** (Classical Trace Test)**.** *Let $X \subseteq \mathbb{C}^n$ be an irreducible variety, and $L_t$ a parallel family of affine spaces, each with codimension equal to the dimension of $X$. For generic $L_t$, the intersection $X \cap L_t$ is a set of points in $\mathbb{C}^n$. A nonempty subset $V_t \subseteq X \cap L_t$ varies continuously at $t$ changes. Then $V_t$ is complete if and only if the trace $\Sigma(V_t)$ is an affine linear function of $t$.*

In practice, we can determine the completeness of a subset by implementing Lemma 2.4 in an algorithm. The Classical Trace Test in Algorithm 1 serves as a completeness test since it gives a condition for determining whether a set contains the complete slice of a variety.

**Algorithm 1:** Classical Trace Test

**Input:** • a positive dimensional variety $\mathcal{V}(\mathcal{F})$
• a family of parallel generic affine spaces $L_t$ of codimension $\dim(\mathcal{V}(\mathcal{F}))$
• a nonzero subset of the variety $S_t \subseteq \mathcal{V}^{\times}(\mathcal{F}) \cap L_t$

**Output:** if $S = \mathcal{V}(\mathcal{F})$, then `pass`, else `fail`

1 Use homotopy continuation, beginning with $t = 1$, to track $S_t \subseteq \mathcal{V}(\mathcal{F}) \cap L_t$ to $t = \frac{1}{2}$ and $t = 0$
2 Compute traces $\Sigma_1(S_0)$, $\Sigma_1(S_{1/2})$, and $\Sigma_1(S_1)$
3 **if** $(0, \Sigma_1(S_0)), (1/2, \Sigma_1(S_{1/2})), (1, \Sigma_1(S_1))$ *are collinear* **then**
  **return** `pass`
**else**
  **return** `fail`

**Example 2.5** ([14])**.** Consider the polynomial system $f \in \mathbb{C}[x, y]$ where $f = x^3 + y^3 - 3xy$. The affine variety $\mathcal{V}(\mathcal{F}) = \{(x, y) \in \mathbb{C}^2 \mid x^3 + y^3 = 3xy\}$ is known as the folium of Descartes. We choose

a family of parallel lines $L_t$, as seen in Figure 2.1, defined by $x - y + t = 0$. For varying values of $t$, $L_t$ is a line passing through the folium in three points, counting complex values and multiplicity. In other words, the intersection $V_t = \mathcal{V}(\mathcal{F}) \cap L_t$ contains three points for each $t$. The traces $\Sigma(V_t)$ for four $t$ values are labeled in Figure 2.1, and together the traces are collinear.

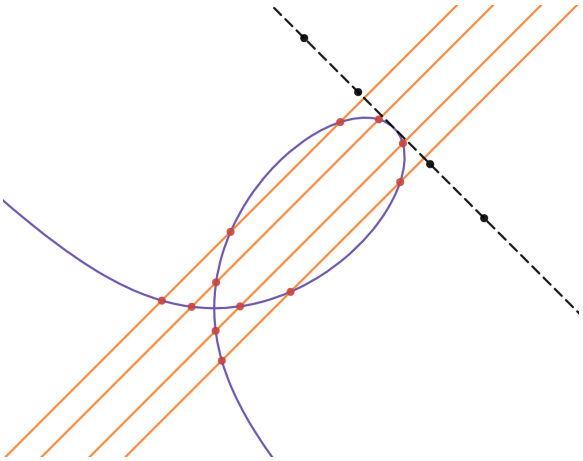


Figure 2.1: The folium of Descartes $X$ in purple, with a parallel family of lines $L_t$ in orange. Four $t$-values are illustrated. The points contained in $X \cap L_t$ for each $t$ are in red, and the trace of each set is a black point. The traces are collinear, as shown by the dotted black line. Image generated using Desmos [7].

**Example 2.6.** Let $X$ be as in Example 2.5. Consider $W_t \subseteq X \cap L_t$ containing only two points. For varying values of $t$ we get the traces $\Sigma(W_t)$, as seen in Figure 2.2. The traces are not collinear, and in fact the relationship among the trace points is cubic. Thus, by the Classical Trace Test 2.4, the subset $W_t \subseteq X \cap L_t$ is not complete.

## 2.2 Sparse Trace Test

Given a collection of supports $\mathcal{A} = (A_1, \ldots, A_n)$ with each $A_i \subseteq \mathbb{Z}^m$, let $\mathcal{F}(u)$ be the universal sparse system supported on $\mathcal{A}$. We call $\mathcal{F}(u)$ the *sparse universal system*, and, from this point forward, we assume polynomials are generic from sparse families, unless otherwise stated. We refine the Classical Trace Test above to the sparse case with a new algorithm. First, we introduce *Newton polytopes*, which encode important information about the variety of a system. Here they are used as a visual aid in the study of the support, but we discuss geometric applications in Section 2.4.

**Definition 2.7.** Given a set $S \subseteq \mathbb{R}^n$, the **convex hull** of $S$, denoted by $\text{Conv}(S)$, is the smallest

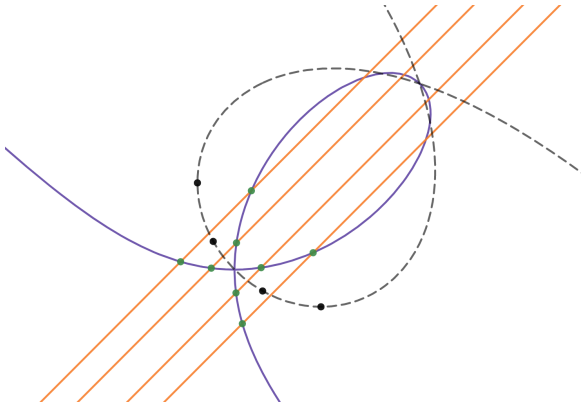


Figure 2.2: This intersection of the folium of Descartes $X$ with the linear family $L_t$ is missing one point. The traces of the incomplete solution sets $W_t$ are not collinear, as seen by the black dashed curve, which goes through the traces. The relationship between the traces is cubic, not linear. Image generated using Desmos [7].

convex set in $\mathbb{R}^n$ which includes $S$. A **polytope** is the convex hull of a finite set in $\mathbb{R}^n$. Given a polynomial $f \in \mathbb{C}[x_1, \ldots, x_n]$, the **Newton polytope** of $f$ is the convex hull of the support $A \subseteq \mathbb{N}^n$ of $f$.

**Example 2.8.** Let $f, g \in \mathbb{C}[x, y]$ where

$$f = x^2y - 2xy + 3y - 4 \qquad \text{and} \qquad g = xy^2 - 2x^2 + 2y + 3$$

We draw the Newton polytope of $f$ and $g$ in Figure 2.3. The horizontal axis of the lattice represents increasing powers on the $x$ variable, while the vertical axis represents increasing powers on the $y$ variable. Each open circle represents an exponent appearing in the polynomial. The shaded region is the Newton polytope. Lattice points in the interior of the Newton polytope are not required to be in the support, as we see in this example.


Figure 2.3: The polynomials $f$ and $g$ in $\mathbb{C}[x, y]$ are each supported by a set in $\mathbb{Z}^2$. The Newton polytope of each polynomial is depicted by the shaded region and is the convex hull of the support.

Any convex polytope $P \subseteq \mathbb{R}^n$ has an $n$-dimensional volume. Letting $x_1, \ldots, x_n$ be coordinates on $\mathbb{R}^n$, we compute the volume by the polynomial:

$$\mathrm{Vol}_n(P) = \int \cdots \int_P 1 \; dx_1 \cdots dx_n.$$

Furthermore, the vector space structure of $\mathbb{R}^n$ allows us to define two operations on polytopes, for more details, see [6]. The *Minkowski sum* of two polytopes $P, Q$ in $\mathbb{R}^n$ is the set of all sums of points in $P$ and $Q$, i.e., $P+Q := \{p+q \mid p \in P \text{ and } q \in Q\}$. Given a real number $\lambda \geq 0$, the scalar multiple $\lambda P$ of $P$ is the set of all points in $P$ scaled by $\lambda$, i.e., $\lambda P := \{\lambda p \mid p \in P\}$. Using these operations, we measure the volume of the support of a polynomial system.

**Definition 2.9.** Given a collection of polytopes $P_1, \ldots, P_n$, consider the function $\lambda_1, \ldots, \lambda_n \mapsto \mathrm{Vol}_n(\lambda_1 P_1 + \cdots + \lambda_n P_n)$. Then $\mathrm{Vol}_n(\lambda_1 P_1 + \cdots + \lambda_n P_n)$ is a polynomial in $\lambda_1, \ldots, \lambda_n$. The $n$-dimensional **mixed volume** of the collection, denoted by $MV(P_1, \ldots, P_n)$, is the the coefficient of the monomial $\lambda_1 \lambda_2 \cdots \lambda_n$ in the volume $\mathrm{Vol}_n(\lambda_1 P_1 + \cdots + \lambda_n P_n)$. Let $\mathcal{A} = (A_1, \ldots, A_n)$ be a collection of supports, where $A_i \subseteq \mathbb{Z}^m$, and let $P_i = \mathrm{Conv}(A_i)$ be the Newton polytope of the $i^{th}$ support. The $n$-dimensional **mixed volume** of $\mathcal{A}$, denoted by $MV(\mathcal{A})$, is the mixed volume of the collection of polytopes $P_1, P_2, \ldots, P_n$.

The concept of the mixed volume for a polynomial system relates the mixed volume to the number of solutions, as in the following theorem due to [3, 10, 11].

**Theorem 2.10** (BKK, [3, 10, 11])**.** *Let $\mathcal{F}$ be a generic polynomial system with square support $\mathcal{A}$ such that $MV(\mathcal{A}) \geq 0$. If $\mathcal{F}$ is outside of a Zariski closed subset, then the cardinality of $\mathcal{V}^{\times}(\mathcal{F})$, when counted with multiplicity, is $MV(\mathcal{A})$. Moreover, for all systems $\mathcal{F}$ supported on $\mathcal{A}$, the cardinality of $\mathcal{V}^{\times}(\mathcal{F})$ counted with multiplicity is at most $MV(\mathcal{A})$.*

Generically, $MV(\mathcal{A}) = 0$ implies there are no nonzero solutions to a system supported on $\mathcal{A}$, so we consider systems where $MV(\mathcal{A}) > 0$. For our purposes, we focus on polynomial systems for which this BKK bound is met.

**Definition 2.11.** Let $\mathcal{A} = (A_1, \ldots, A_n)$ be a collection of supports, where $A_i \subseteq \mathbb{Z}^m$. A **Bernstein-generic system** $\mathcal{F}$ supported on $\mathcal{A}$ is one which has exactly the same number of roots (each with multiplicity 1) as the mixed volume $MV(\mathcal{A})$.

Following [5], we introduce the Sparse Trace Test in Algorithm 2. This algorithm serves as a completeness test for a subset $W \subseteq \mathcal{V}^{\times}(\mathcal{F})$, which contains some nonzero solutions of the sparse polynomial system $\mathcal{F}$. The inputs are a system of polynomials and a subset $W$ of $\mathcal{V}^{\times}(\mathcal{F})$. The output is `pass` if the subset of the variety is complete and `fail` if the subset is incomplete. The algorithm deals with the trace in the first coordinate and analogous results are true for traces in other coordinates.

The Sparse Trace Test in Algorithm 2 is similar to the Classical Trace Test because both tests track a set of solution points by $t$ so that the trace of the set $\Sigma_1(S_t)$ is a function of $t$. Both tests state that the function is linear if and only if the solution set is complete. The Sparse Trace Test is so named because, instead of beginning with a specific variety or system, the first input to the test is a sparse support $\mathcal{A}$. Additionally, the Classical Trace Test works on nonsquare systems.

**Algorithm 2:** Sparse Trace Test [5]

**Input:** • a collection of supports $\mathcal{A}$ such that $L[\mathcal{A}] = \mathbb{Z}^n$ and $MV(\mathcal{A}) > 0$
• a Bernstein-generic system $\mathcal{F} \in \mathbb{C}^{\mathcal{A}}$
• a nonzero subset of the variety $S \subseteq \mathcal{V}^{\times}(\mathcal{F})$
• a collection $\mathcal{B} \subseteq \mathcal{A}$ which is trace affine linear and abundant

**Output:** if $S = \mathcal{V}^{\times}(\mathcal{F})$, then `pass`, else `fail`

1 Choose a generic system $\mathcal{H} \in \mathbb{C}^{\mathcal{B}}$ and construct $\mathcal{G} \in \mathbb{C}^{\mathcal{A}}$ such that $\mathcal{G} = \mathcal{H}$ over support $\mathcal{B}$ and $\mathcal{G} = \mathcal{F}$ otherwise
2 Use homotopy continuation, beginning with $S = S_1$, to track $S_t \subseteq \mathcal{V}^{\times}(t\mathcal{F} + (1-t)\mathcal{G})$ to $t = \frac{1}{2}$ and $t = 0$
3 Compute traces $\Sigma_1(S_0)$, $\Sigma_1(S_{1/2})$, and $\Sigma_1(S_1)$
4 **if** $(0, \Sigma_1(S_0)), (1/2, \Sigma_1(S_{1/2})), (1, \Sigma_1(S_1))$ *are collinear* **then**
  **return** `pass`
**else**
  **return** `fail`

One input to Algorithm 2 is a *trace affine linear* subset $\mathcal{B} \subseteq \mathcal{A}$ of the support. This type of subset is a collection of the support which collectively influences the trace in an affine linear manner. Trace affine linear sets are discussed in more detail in the following sections.

## 2.3 Finding Trace Affine Linear Sets

Given a support $\mathcal{A}$, the trace is a function $\Sigma_1 : \mathbb{C}^{\mathcal{A}} \to \mathbb{C}$, and a generic element $\mathcal{F} \in \mathbb{C}^{\mathcal{A}}$ is a system with coefficients in $\mathbb{C}^{\mathcal{A}}$. Let the coefficients of the system be affine linearly dependent on some $\mathbf{t} \in \mathbb{C}^{\mathcal{A}}$. Then $\mathcal{F}_{\mathbf{t}} \in \mathbb{C}^{\mathcal{A}}$ is an affine linear function of $\mathbf{t}$. We can define a function $h : \mathbb{C}^{\mathcal{A}} \to (\mathbb{C}[\mathbf{x}])^m$

such that values for $\mathbf{t}$ are mapped to systems $\mathcal{F}_{\mathbf{t}} \in (\mathbb{C}[\mathbf{x}])^m$. Thus the composition $\Sigma_1 \circ h$ is a function $\mathbb{C}^{\mathcal{A}} \to \mathbb{C}$ that maps a value $\mathbf{t} \in \mathbb{C}^{\mathcal{A}}$ to the trace $\Sigma_1(\mathcal{V}^{\times}(\mathcal{F}_{\mathbf{t}}))$, where

$$\mathbb{C}^{\mathcal{A}} \xrightarrow{h} (\mathbb{C}[\mathbf{x}])^m \xrightarrow{\Sigma_1} \mathbb{C}$$

$$\mathbf{t} \longmapsto \mathcal{F}_{\mathbf{t}} \longmapsto \Sigma_1(\mathcal{V}^{\times}(\mathcal{F}_{\mathbf{t}})).$$

Note that the function $\mathbf{t} \mapsto \mathcal{F}_{\mathbf{t}}$ is affine linear. Then if the function from $\mathbf{t}$ to $\Sigma_1(\mathcal{V}^{\times}(\mathcal{F}_{\mathbf{t}})$ is nonlinear, we know it must be the trace function which is acting nonlinearly. So, by observing the linearity as $\mathbf{t} \mapsto \Sigma_1(\mathcal{V}^{\times}(\mathcal{F}_{\mathbf{t}}))$, we can isolate the effect certain coefficients, and therefore elements of the support, have on the trace.

**Definition 2.12.** Let $\mathcal{F}$ be a polynomial system supported on $\mathcal{A}$. A subset of the support $\mathcal{B} \subseteq \mathcal{A}$ is **trace affine linear** if the trace $\Sigma(\mathcal{V}^{\times}(\mathcal{F}))$ is an affine linear function of the coefficients which are indexed by elements in $\mathcal{B}$.

In practice, it can be challenging to confirm a subset is trace affine linear. We introduce a formula for the trace of a variety which depends on elements of the support of the system. Then we classify elements of the support based on how they appear in this formula. The formula in the sparse case involves sparse resultants.

In the univariate case, we can write down a simple formula for the trace. This formula goes back to Newton. We write the polynomial $f \in \mathbb{C}[x]$ as $f(x) = a(x - r_1)(x - r_2)\cdots(x - r_n)$, where the $n$ roots of $f$ are $r_1, \ldots, r_n$. Expanding the polynomial, we rewrite $f$ as $f = ax^n - a(r_1 + \cdots + r_n)x^{n-1} + \cdots - ar_1r_2\cdots r_n$. The negative coefficient of the $x^{n-1}$ term divided by the coefficient of the $x^n$ term is the trace:

$$-\frac{-a(r_1 + \cdots + r_n)}{a} = r_1 + \cdots + r_n = \Sigma_1(f) \tag{2.1}$$

Writing the expanded form of the polynomial in one variable as $f = c_nx^n + c_{n-1}x^{n-1} + \cdots + c_1x + c_0$, where $c_n = a$, $c_{n-1} = a(r_1 + \cdots + r_n)$, and so on, the formula for the trace is $-\frac{c_{n-1}}{c_n}$. The coefficient $c_{n-1}$ influences the formula for the trace in the univariate case linearly. Also, $c_n$ influences the trace non-linearly, since it shows up in the denominator of the formula for the trace. All other coefficients of $f$ do not influence the trace. Equivalently, the other coefficients influence the trace in a constant manner. We introduce new vocabulary to categorize this classification.

**Definition 2.13.** Let $\mathcal{A} = (A_1, \ldots, A_n)$ be a set of supports and $\mathcal{F}(u)$ the universal system supported on $\mathcal{A}$. Let $\mathcal{F}_t$ be a generic system supported on $\mathcal{A}$ where $t$ is the coefficient of the monomial $\mathbf{x}^{\alpha}_{i,j}$. Then $\alpha_{i,j} \in \mathcal{A}$ has **agency** classified by behavior of the function $t \mapsto \Sigma_1(\mathcal{V}(\mathcal{F}_t))$.

- $\alpha_{i,j}$ has **independent agency** if $\Sigma_1(\mathcal{V}(\mathcal{F}_t))$ is a constant function of $t$
- $\alpha_{i,j}$ has **linear agency** if $\Sigma_1(\mathcal{V}(\mathcal{F}_t))$ is an affine linear function of $t$
- $\alpha_{i,j}$ has **nonlinear agency** if $\Sigma_1(\mathcal{V}(\mathcal{F}_t))$ is a nonlinear function of $t$

We also introduce a system of shading elements of the support to visually represent their agency. Open circles represent elements with independent agency, crosshatched circles are elements with linear agency, and shaded circles are elements with nonlinear agency. In Figure 2.4, the univariate formula for the trace is depicted.

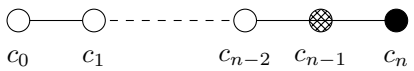



Figure 2.4: The trace of the univariate polynomial $f(x) = c_n x^n + c_{n-1}x^{n-1} + \cdots + c_1 x + c_0$ is $-\frac{c_{n-1}}{c_n}$. The points of the Newton polytope of the support are shaded according to agency: open for independent, crosshatched for linear, or filled for nonlinear.

Beyond the univariate case, one formula for $\Sigma_1(\mathcal{V}^{\times}(\mathcal{F}))$ comes from the *hidden variable sparse resultant* of the system, see Lemma 2.17. Let $\mathcal{F}$ be a square system supported on $\mathcal{A}$, with each $A_i \subseteq \mathbb{Z}^n$. Recall from Section 1.2 that the resultant is defined for a system $\mathcal{F}$ with one more equation than variables. To compute the resultant of the square system $\mathcal{F}$ with $n$ equations in $n$ variables, we "hide" one of the variables by considering it as a coefficient, resulting in a system of $n$ equations and $n-1$ variables.

**Definition 2.14.** Let $\mathcal{F} = (f_1, \ldots, f_n) \in (\mathbb{C}[x_1, \ldots, x_n])$ be a square polynomial system. The **hidden variable resultant** of $\mathcal{F}$ in the $i^{th}$ coordinate is $\mathrm{Res}(\mathcal{F}; x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n) = \mathrm{Res}(\mathcal{G})$ where $\mathcal{G} = (f_1, \ldots, f_n) \in (\mathbb{C}[x_i][x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n])^n$.

The hidden variable resultant of $\mathcal{F}$ in the $i^{th}$ coordinate is a polynomial in $x_i$. Generically, the roots of this resultant are the $x_i$-coordinates of the solutions to the system; see [6, Section 3.5] for the proof.

**Example 2.15.** Consider the system $\mathcal{F} = (f, g) \in (\mathbb{C}[x, y])^2$ where:

$$f = x^2y - 2xy + 3y - 4 \qquad \text{and} \qquad g = xy^2 - 2x^2 + 2y + 3$$

The system is supported on $\mathcal{A} = (\{(2,1),(1,1),(0,1),(0,0)\},\{(1,2),(2,0),(0,1),(0,0)\})$. The mixed volume is $MV(\mathcal{A}) = 6$, so we expect six common solutions. We consider $f$ and $g$ as polynomials in $y$ whose coefficients are polynomials in $x$. Using Definition 1.16, the resultant is a polynomial in $x$ whose solutions are the $x$-values of the common solutions of $f$ and $g$:

$$\begin{aligned}\mathrm{Res}(f,g,y) &= \det\begin{pmatrix} x^2-2x+3 & 0 & x \\ -4 & x^2-2x+3 & 2 \\ 0 & -4 & -2x^2+3 \end{pmatrix} \\ &= -2x^6+8x^5-17x^4+12x^3+20x^2-36x+51 \end{aligned} \tag{2.2}$$

Solving the polynomial $\mathrm{Res}(f,g,y)$ for $x$ using *Bertini* [1], there are two real roots and three complex roots, listed in the first column of Table 2.1. These are the $x$-values which make the resultant polynomial equal to zero, so they are exactly the $x$-values where $f$ and $g$ have common solutions.

For comparison, directly computing the solutions of the system $\mathcal{F} = (f,g)$ in *Bertini* gives the solutions depicted in the second and third columns of Table 2.1. The $x$-values of these solutions exactly match the values found by the hidden variable resultant technique.

| $x$-value, sol. of Res. | $x$-value, directly solved | $y$-value, directly solved |
|---|---|---|
| 2.09486 | 2.09486 | 1.2505 |
| $.438696 + 1.25604i$ | $.438696 + 1.25604i$ | $1.16496 + 2.22755i$ |
| $1.192 + 1.91039i$ | $1.192 + 1.91039i$ | $-2.05504 - .934788i$ |
| $-1.35625$ | $-1.35625$ | .529666 |
| $1.192 - 1.91039i$ | $1.192 - 1.91039i$ | $-2.05504 + .934788i$ |
| $.438696 - 1.25604i$ | $.438696 - 1.25604i$ | $1.16496 - 2.22755i$ |

Table 2.1: Solutions to $\mathcal{F} = (f,g)$ computed using the hidden variable resultant are compared with solutions computed using *Bertini* zero dimensional solver [1].

When applied to a universal system of polynomials $\mathcal{F}(u)$, the hidden variable resultant is a polynomial in $x_i$ where the coefficients are polynomials in $x_i$ and the parameters $u_\alpha$ of $\mathcal{F}(u)$. Solutions of $\mathrm{Res}(\mathcal{F}(u), x_1, \ldots, x_{i-1}, x_{i+1}, \ldots, x_n)$ correspond to coefficients and $x_i$-coordinates for which a system $\mathcal{F}$ has a solution. The next example looks at the universal system associated with the system of Example 2.15.

**Example 2.16.** Consider the universal system $\mathcal{F}(u)$ supported on $\mathcal{A} = (A_1, A_2)$ where $A_1 = \{(2,1),(1,1),(0,1),(0,0)\}$ and $A_2 = \{(1,2),(2,0),(0,1),(0,0)\}$; see Figure 2.3. We write the univer-

sal family as $\mathcal{F}(u) = (f, g)$ with:

$$f = u_{2,1}x^2y + u_{1,1}xy + u_{0,1}y - u_{0,0} \qquad \text{and} \qquad g = v_{1,2}xy^2 + v_{2,0}x^2 + v_{0,1}y + v_{0,0}.$$

Each coefficient corresponds to an element of the support of the system. For example, $u_{2,1}$ corresponds to the element $(2, 1)$ in support of $f$. Similarly, $v_\alpha$ coefficients correspond to elements $\alpha$ in the support of $g$. In order to use the hidden variable resultant, we interpret $f$ and $g$ as polynomials in $y$ whose coefficients are polynomials in $x$:

$$f = (u_{2,1}x^2 + u_{1,1}x + u_{0,1})y - u_{0,0} \qquad \text{and} \qquad g = (v_{1,2}x)y^2 + (v_{0,1})y + v_{2,0}x^2 + v_{0,0}.$$

The support of the system with $x$ hidden is the projection of the original support onto the $y$-axis. Now that we have a system of two polynomials in one variable, we compute the resultant:

$$\begin{aligned} h = \mathrm{Res}(f, g; y) &= \det \begin{pmatrix} u_{2,1}x^2 + u_{1,1}x + u_{0,1} & 0 & v_{2,0}x \\ u_{0,0} & u_{2,1}x^2 + u_{1,1}x + u_{0,1} & v_{0,1} \\ 0 & u_{0,0} & v_{1,2}x^2 + v_{0,0} \end{pmatrix} \\ &= (u_{2,1}^2v_{1,2})x^6 + (2u_{2,1}u_{1,1}v_{1,2})x^5 + (u_{1,1}^2v_{1,2} + 2u_{2,1}u_{0,1}v_{1,2} + u_{2,1}^2v_{0,0})x^4 \\ &+ (2u_{1,1}u_{0,1}v_{1,2} + 2u_{2,1}u_{1,1}v_{0,0})x^3 + (u_{0,1}^2v_{1,2} - u_{2,1}u_{0,0}v_{0,1}^2 + u_{1,1}^2v_{0,0} + 2u_{2,1}u_{0,1}v_{0,0})x^2 \\ &+ (u_{0,0}v_{2,0} - u_{1,1}u_{0,0}v_{0,1} + 2u_{1,1}u_{0,1}v_{0,0})x - u_{0,1}u_{0,0}v_{0,1} + u_{0,1}^2v_{0,0} \end{aligned} \tag{2.3}$$

Note that this resultant, as expected, is a polynomial in one variable, $x$, whose coefficients are polynomials in the coefficients of $f$ and $g$. In fact, if we substitute the coefficients used in Example 2.15 for the $u$'s and $v$'s, we would get the polynomial shown in Equation (2.2). From the trace function for a univariate polynomial, Equation (2.1),

$$\Sigma_1(h) = -\frac{c_5}{c_6} = -\frac{2u_{2,1}u_{1,1}v_{1,2}}{u_{2,1}^2v_{1,2}} = -\frac{2u_{1,1}}{u_{2,1}}.$$

Analyzing this formula, we see that the trace is a linear function of $u_{1,1}$, so $(1, 1)$ has linear agency. The only coefficient appearing non-linearly is $u_{2,1}$, so $(2, 1)$ has nonlinear agency. All other coefficients correspond to elements with independent agency. We shade the points of the Newton polytope of the universal system according to this information in Figure 2.5.

Support of $f$ Support of $g$

Figure 2.5: Elements of the support are shaded according to their agency in the $x$-coordinate trace of the system. Elements with non-linear agency are shaded, ones with linear agency are cross-hatched, and those with independent agency are left white.

The following lemma clarifies a formula for the trace computed via sparse resultants. The degree of the hidden variable sparse resultant for generic $\mathcal{F}$ is the same as the number of solutions in $\mathcal{V}^{\times}(\mathcal{F})$. Additionally, the cardinality of $\mathcal{V}^{\times}(\mathcal{F})$ is $MV(\mathcal{A})$ by Theorem 2.10. Therefore the degree of the hidden variable sparse resultant polynomial is the mixed volume of $\mathcal{A}$. See [6] for more details.

**Lemma 2.17** (Formula for Trace)**.** *Let $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ be a square set of supports such that $MV(\mathcal{A}) = m > 0$. Let $\mathcal{F}(u)$ be the universal system having support $\mathcal{A}$. Then the sparse hidden variable resultant which hides the first variable $x_1$ is the polynomial:*

$$Res_{\mathcal{A}}(\mathcal{F}(u), x_2, \ldots, x_n) = d_m(u)x_1^m + d_{m-1}(u)x_1^{m-1} + \cdots + d_1(u)x_1 + d_0(u),$$

*where each $d_i(u)$ is an expression in the parameters $u_\alpha$ of the universal system $\mathcal{F}(u)$. Then the trace of the solutions to the system in the $x_1$-coordinate is given by the formula*

$$\Sigma_1(\mathcal{V}^{\times}(\mathcal{F})) = -\frac{d_{m-1}(u)}{d_m(u)}.$$

**Proof sketch:** For a detailed proof, see [5]. The sparse resultant $h = \mathrm{Res}_{\mathcal{A}}(\mathcal{F}(u), x_2, \ldots, x_n)$ is a polynomial in $x_1$ with coefficients in the parameters $u_\alpha$ of the universal system. For generic $\mathcal{F} \in \mathbb{C}^{\mathcal{A}}$ in the universal system, we have values $c_\alpha \in \mathbb{C}$ to replace parameters $u_\alpha$, so the sparse resultant $h$ is a univariate polynomial in $\mathbb{C}[x_1]$ whose roots are the $x_1$-solutions of the system $\mathcal{F}$. A generic $\mathcal{F}$ has $d_m(c) \neq 0$, so the trace of $\mathcal{V}^{\times}(\mathcal{F})$ in the first coordinate is as according to Equation 2.1:

$$\Sigma_1(\mathcal{V}^{\times}(\mathcal{F})) = -\frac{d_{m-1}(c)}{d_m(c)}.$$

Since this property holds for generic $\mathcal{F} \in \mathbb{C}^{\mathcal{A}}$, it holds on the universal system $\mathcal{F}(u)$. □

**Remark 2.18.** Given a support $\mathcal{A}$, one way to obtain a candidate trace affine linear subset of the support $\mathcal{B} \subseteq \mathcal{A}$ is to compute the hidden variable sparse resultant of a universal system $\mathcal{F}(u) \in \mathbb{C}^{\mathcal{A}}$ and observe which coefficients appear in the formula $-\frac{d_{m-1}(u)}{d_m(u)}$. We saw this at work with a Sylvester resultant in Example 2.16, and the above lemma extends the idea to computations involving sparse resultants.

The main result of this section is that we can identify a trace affine linear subset of the support directly from the formula for the trace given by the hidden variable sparse resultant.

**Lemma 2.19** (Individual Agency Classification)**.** *Let $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ be a square set of supports such that $MV(\mathcal{A}) > 0$. Let $\mathcal{F}(u)$ be the universal system with support $\mathcal{A}$ where coefficient $u_{\alpha_{i,j}}$ is associated with element $\alpha_{i,j} \in A_i$. We have the following:*

- *parameter $u_{\alpha_{i,j}}$ does not appear in the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ in Lemma 2.17 if and only if $\alpha_{i,j}$ has independent agency.*
- *parameter $u_{\alpha_{i,j}}$ appears in the numerator to degree at most one and does not appear in the denominator in the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ if and only if $\alpha_{i,j}$ has linear agency.*
- *parameter $u_{\alpha_{i,j}}$ appears in the numerator to degree greater than one or appears in the denominator the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ if and only if $\alpha_{i,j}$ has nonlinear agency.*

**Proof.** Let $\mathbf{t}$ contain fixed coefficients for all elements of $\mathcal{A}$, except, for $\alpha_{i,j}$, let the coefficient be parameter $u_{\alpha_{i,j}}$. Then $\mathbf{t} \mapsto \Sigma_1(\mathcal{V}^\times(\mathcal{F}_\mathbf{t}))$ is a function which maps values for $u_{\alpha_{i,j}}$ to values of the trace $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$.

- $u_{\alpha_{i,j}}$ does not appear in the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ if and only if $\Sigma_1(\mathcal{V}^\times(\mathcal{F}_\mathbf{t}))$ is a constant function of $\mathbf{t}$. So $\alpha_{i,j}$ has independent agency by defnition.
- $u_{\alpha_{i,j}}$ appears in the numerator to degree at most one and does not appear in the denominator in the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ if and only if $\Sigma_1(\mathcal{V}^\times(\mathcal{F}_\mathbf{t}))$ is an affine linear function of $\mathbf{t}$. So $\alpha_{i,j}$ has linear agency by definition.
- $u_{\alpha_{i,j}}$ appears in the numerator to degree greater than one or appears in the denominator the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ if and only if $\Sigma_1(\mathcal{V}^\times(\mathcal{F}_\mathbf{t}))$ is a nonlinear function of $\mathbf{t}$. So $\alpha_{i,j}$ has nonlinear agency by definition. □

**Lemma 2.20.** *Let $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ be a square set of supports such that $MV(\mathcal{A}) > 0$. Let $\mathcal{F}(u)$ be the universal system having support $\mathcal{A}$, where coefficient $u_{\alpha_{i,j}}$ is associated with element $\alpha_{i,j} \in A_i$. A subset $\mathcal{B} \subseteq \mathcal{A}$ is trace affine linear if and only if $\mathcal{B}$ contains only elements $\alpha_{i,j}$ which have independent or linear agency, and each monomial in the numerator of the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ has at most one element of $\mathcal{B}$.*

**Proof.** ($\Rightarrow$) Suppose $\mathcal{B} \subseteq \mathcal{A}$ is a trace affine linear subset. By definition, the trace $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ is an affine linear function of the coefficients which are indexed by elements $\alpha_{i,j} \in \mathcal{B}$. The trace $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ must then be an affine linear function of each coefficient individually, so every element $\alpha_{i,j} \in \mathcal{B}$ has independent or linear agency. Additionally, suppose some monomial in the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ has two elements of $\mathcal{B}$. Then $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ would be a nonlinear function of these two coefficients, which is a contradiction since $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ is an affine linear function of all coefficients indexed by $\mathcal{B}$.

($\Leftarrow$) Now suppose $\mathcal{B}$ contains only elements $\alpha_{i,j}$ which have independent or linear agency, and each monomial in the numerator of the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ has at most one element of $\mathcal{B}$. Since each element $\alpha_{i,j} \in \mathcal{B}$ has independent or linear agency, by Lemma 2.19, $u_{\alpha_{i,j}}$ appears at most in the numerator to degree one. Let $\mathbf{t}$ contain fixed coefficients for all elements of $\mathcal{A}$, except, for $\alpha_{i,j} \in \mathcal{B}$, let the coefficient be parameter $u_{\alpha_{i,j}}$. Since no monomial of $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ can contain more than one of these $u_{\alpha_{i,j}}$'s, $\Sigma_1(\mathcal{V}^\times(\mathcal{F}_\mathbf{t}))$ is an affine linear function of $\mathbf{t}$. Therefore $\mathcal{B}$ is a trace affine linear subset of the support. □

**Definition 2.21.** Let $\mathcal{A} = (A_1, \ldots, A_n)$ be a set of supports and $\mathcal{F}(u)$ the universal system supported on $\mathcal{A}$. Let $\mathcal{F}_\mathbf{t}$ be a generic system supported on $\mathcal{A}$ where the two coefficients $u_{\alpha_{i_1,j_1}}, u_{\alpha_{i_2,j_2}}$ associated with the elements $\alpha_{i_1,j_1}, \alpha_{i_2,j_2} \in \mathcal{A}$ are affine linearly dependent on the parameter $\mathbf{t}$. Then we say the pair has **joint linear agency** if $\Sigma_1(\mathcal{V}(\mathcal{F}_\mathbf{t}))$ is an affine linear function of $\mathbf{t}$.

**Theorem 2.22.** *Let $\mathcal{A} = (A_1, A_2, \ldots, A_n)$ be a square set of supports such that $MV(\mathcal{A}) > 0$, and let $\mathcal{B} \subseteq \mathcal{A}$. $\mathcal{B}$ is a trace affine linear subset of the support if and only if $\mathcal{B}$ contains only elements of independent and linear agency, and for all pairs in $\mathcal{B}$ with linear agency, the pair has joint linear agency.*

**Proof.** ($\Rightarrow$) Suppose $\mathcal{B} \subseteq \mathcal{A}$ is trace affine linear. By Lemma 2.20, $\mathcal{B}$ contains only elements $\alpha_{i,j}$ which have independent or linear agency, and each monomial in the numerator of the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ has at most one element of $\mathcal{B}$. Consider any pair $\alpha_{i_1,j_1}, \alpha_{i_2,j_2} \in \mathcal{B}$ with linear agency.

By Lemma 2.19, the corresponding coefficients $u_{\alpha_{i_1,j_1}}, u_{\alpha_{i_2,j_2}}$ appear in the numerator to degree at most one and do not appear in the denominator of $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$. Let $\Sigma_1(\mathcal{V}(\mathcal{F}_\mathbf{t}))$ be a function where coefficients $u_{\alpha_{i_1,j_1}}, u_{\alpha_{i_2,j_2}}$ are affine linearly dependent on parameter $\mathbf{t}$. Now $u_{\alpha_{i_1,j_1}}$ and $u_{\alpha_{i_2,j_2}}$ only appear linearly in the numerator, and cannot appear in the same monomial in the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$. Now $\Sigma_1(\mathcal{V}(\mathcal{F}_\mathbf{t}))$ is an affine linear combination of monomials of the form $a\mathbf{t}$, so it is an affine linear function of $\mathbf{t}$. Thus the pair $\alpha_{i_1,j_1}, \alpha_{i_2,j_2}$ has joint linear agency. This is true for every pair with linear agency in $\mathcal{B}$.

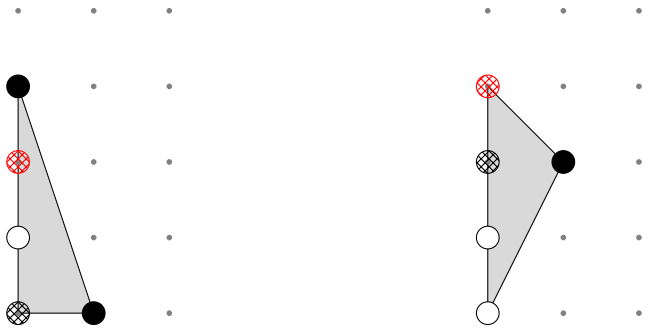
($\Leftarrow$) Now suppose $\mathcal{B}$ contains only elements of independent and linear agency, and for all pairs in $\mathcal{B}$ with linear agency, the pair has joint linear agency. Let $\alpha_{i_1,j_1}, \alpha_{i_2,j_2} \in \mathcal{B}$ be some pair with linear agency. The function $\Sigma_1(\mathcal{V}(\mathcal{F}_\mathbf{t}))$, where coefficients $u_{\alpha_{i_1,j_1}}, u_{\alpha_{i_2,j_2}}$ are affine linearly dependent on parameter $\mathbf{t}$, is an affine linear function of $\mathbf{t}$. Then there can be no monomial in $\Sigma_1(\mathcal{V}(\mathcal{F}_\mathbf{t}))$ which has both $u_{\alpha_{i_1,j_1}}$ and $u_{\alpha_{i_2,j_2}}$, since this would make the function quadratic in $t$. This is true for all possible pairs with linear agency in $\mathcal{B}$, so each monomial in the numerator of the formula for $\Sigma_1(\mathcal{V}^\times(\mathcal{F}))$ has at most one element of $\mathcal{B}$. Thus, by Lemma 2.20, $\mathcal{B}$ is a trace affine linear subset of the support. □

The following example illustrates the need for Definition 2.21 and the theorem relating trace affine linear sets and joint linear agency.

**Example 2.23.** Let $\mathcal{F}(u)$ be the universal system supported on $\mathcal{A}$ such that:

$$\mathcal{F}(u) = \begin{cases} f = a_{(0,0)} + a_{(1,0)}x + a_{(0,1)}y + a_{(0,2)}y^2 + a_{(0,3)}y^3 \\ g = b_{(0,0)} + b_{(0,1)}y + b_{(1,2)}xy^2 + b_{(0,2)}y^2 + b_{(0,3)}y^3 \end{cases}$$

Figure 2.6 shows the support classified by individual agency. Note that the element $(0,2)$ in the support of $f$ and the element $(0,3)$ in the support of $g$ are each identified as having linear agency. However, we compute the hidden variable sparse resultant and obtain a formula for the trace, see Equation (2.4). The product $a_{(0,2)}b_{(0,3)}$ appears in the second term in the numerator of the formula for the trace. Thus, when considering the elements together, the trace is a nonlinear (quadratic) function of $a_{(0,2)}$ and $b_{(0,3)}$. The two elements $(0,2) \in A_1$ and $(0,3) \in A_2$, despite having individual linear agency, do not form a trace affine linear set. Thus any subset of the support containing both of these elements is not trace affine linear.



Support of $f$ Support of $g$

Figure 2.6: Classification of elements in the support $\mathcal{A}$ by individual agency. Two support elements which individually have linear agency do not have joint linear agency. These two elements, shaded in red, cannot both be in a trace affine linear set.

$$\Sigma_1(\mathcal{V}^{\times}(\mathcal{F})) = \frac{-3a_{(0,3)}a_{(1,0)}b_{(0,2)} + a_{(0,2)}a_{(1,0)}b_{(0,3)} - 2a_{(0,0)}a_{(0,3)}b_{(1,2)}}{a_{(0,3)}a_{(1,0)}b_{(1,2)}} \tag{2.4}$$

By Theorem 2.22, in order to obtain a maximal trace affine linear subset of the support, we need to check whether all pairs with linear agency have joint linear agency. The support elements $(0,2) \in A_1$ and $(0,3) \in A_2$ do not have joint linear agency, so we must exclude at least one of them from any trace affine linear set. The set of all support elements with independent or linear agency without $(0,2) \in A_1$ is a maximal trace affine linear subset of the support. Similarly, the set of all support elements with independent or linear agency without $(0,3) \in A_2$ is also maximal trace affine linear subset.

We conclude from these theorems that the formula for the trace given by the hidden variable sparse resultant provides all information needed to completely describe a trace affine linear subset of the support. However, even though some hidden variable sparse resultants can be computed using symbolic algebra software, it is not difficult to come up with a system for which the sparse resultant is too large to compute.

**Example 2.24.** Let $\mathcal{F} = (f, g, h)$ be a polynomial system in $(\mathbb{C}[x, y, z])^3$, where:

$$f = a_{(0,0,0)} + a_{(1,0,1)}xz + a_{(1,2,0)}xy^2 + a_{(2,2,1)}x^2y^2z$$
$$g = b_{(0,0,0)} + b_{(0,1,1)}yz + b_{(2,0,1)}x^2z + b_{(1,2,2)}xy^2z^2$$
$$h = c_{(0,0,0)} + c_{(1,1,1)}xyz + c_{(1,1,2)}xyz^2 + c_{(2,1,2)}x^2yz^2$$

We consider each polynomial in $\mathbb{C}[x,y][z]$, that is, as a polynomial in $z$ whose coefficients are polynomials in $x$ and $y$. The support $\mathcal{A} = (A_1, A_2, A_3)$ projected onto the $xy$-plane is:

$$\begin{aligned} A_1 &= \{(0,0),(1,0),(1,2),(2,2)\}, \\ A_2 &= \{(0,0),(0,1),(2,0),(1,2)\}, \quad \text{and} \\ A_3 &= \{(0,0),(1,1),(1,1),(2,1)\}. \end{aligned}$$

To compute the hidden variable sparse resultant, we use the *SparseResultants* package [16] in *Macaulay2* [9]. This computation was run on a laptop with MacOS Catalina 10.15.7, 2.9 GHz dual-core intel core i5 processor, and 8GB of memory. After running this example for several hours, the computation did not finish. The hidden variable resultant of this three variable system, even though the system is not of large total degree, is too much for a typical computer to handle.

Example 2.24 shows that the sparse resultant approach to finding trace affine linear subsets does not scale well. This motivates a search for different methods of classifying a trace affine linear subset, since the Sparse Trace Test of Algorithm 2 requires a trace affine linear subset of the support.

## 2.4 Estimating Trace Affine Linear Subsets

Aspects of polyhedral geometry can be used to estimate candidate sets that are trace affine linear. In this section, we follow [5] to illustrate these techniques.

**Definition 2.25.** Let $A \subseteq \mathbb{Z}^n$ be a support, and let $e_i$ be the basis vector in the direction of the $i^{th}$ coordinate. For some distance $k \in \mathbb{R}^+$, the set of **k-offset** points of $A$ in the $i^{th}$ coordinate is

$$\text{offset}_i(A,k) = \{\alpha \in A \mid \alpha + (k+\epsilon)e_i \notin \text{Conv}(A) \text{ for all } \epsilon > 0\}$$

**Example 2.26.** Consider the support from Figure 2.3, where $f$ and $g$ are supported on $A_1 = \{(2,1),(1,1),(0,1),(0,0)\}$ and $A_2 = \{(1,2),(2,0),(0,1),(0,0)\}$, respectively. For $\mathcal{A} = (A_1, A_2)$, Figure 2.7 identifies $\text{offset}(\mathcal{A}, 0.5)$ and $\text{offset}_1(\mathcal{A}, 1)$ by shifting the Newton polytopes, as seen by the dashed lines.

Offset points relate to trace affine linear sets, as classified by the following lemma.


Figure 2.7: Support of $f$ and $g$ with elements shaded according to shifts in the convex hull. Fully shaded elements are the 0.5-offset points, while 1-offset points are crosshatched.

**Lemma 2.27** ([5])**.** *Let $\mathcal{A} = (A_1, \ldots, A_n)$ be a square set of supports such that $MV(\mathcal{A}) > 0$. Then $\mathcal{A}$* ***without*** *its 0.5-offset points is a trace affine linear subset $\mathcal{B} \subseteq \mathcal{A}$. Additionally, $\mathcal{A}$* ***without*** *its 1-offset points is a subset of the elements of $\mathcal{A}$ which have independent agency.*

Lemma 2.27 provides candidate subsets which are trace affine linear, but the classification given by the polyhedral geometry is more conservative than what would result from the resultant technique. That is, the set $\mathcal{A} \setminus \text{offset}(\mathcal{A}, 0.5)$ may not be a maximal trace affine linear subset, and the set $\mathcal{A} \setminus \text{offset}(\mathcal{A}, 1)$ may not contain every element with independent agency. Indeed, in Figure 2.5 from Section 2.3, we see that every element of $A_2$ has independent agency, but in Figure 2.7 above, two of those elements were shaded as nonlinear, according to the geometry.

The next example has a larger support and includes elements which are in the interior of the Newton polytope.

**Example 2.28.** Let $\mathcal{F} = (f_1, f_2)$ be a polynomial system supported on $\mathcal{A} = (A_1, A_2)$ where

$$A_1 = \{(0,0), (0,2), (1,0), (1,1), (1,2), (1,3), (1,4), (2,0), (2,2), (2,3), (2,4), (3,1)\} \text{ and}$$
$$A_2 = \{(0,0), (0,1), (0,2), (0,3), (1,0), (1,1), (1,2), (1,3), (2,0), (2,1), (2,2), (2,3)\}.$$

We apply Lemma 2.27 in the $x$-direction, as seen in Figure 2.8. According to the polyhedral geometry, a subset of the support which affects the trace $\Sigma_1$ affine linearly is $\mathcal{B} = (B_1, B_2)$, where

$$B_1 = \{(0,0), (0,2), (1,0), (1,1), (1,2), (1,3), (1,4), (2,2)\} \text{ and}$$
$$B_2 = \{(0,0), (0,1), (0,2), (0,3), (1,0), (1,1), (1,2), (1,3)\}.$$

Next we apply the lemma to the support in the $y$-direction, as seen in Figure 2.9. A subset

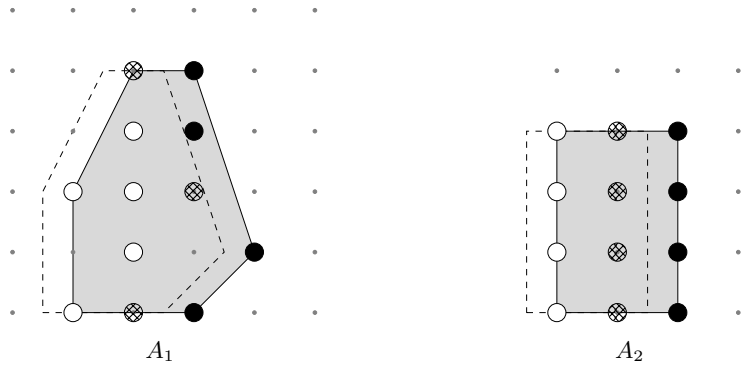



Figure 2.8: Supports $A_1$ and $A_2$ with offset points in the $x$-direction. 0.5-offset points are fully shaded and 1-offset points are crosshatched.

which affects the trace in the second coordinate, $\Sigma_2$, affine linearly is $\mathcal{B} = (B_1, B_2)$ where

$$B_1 = \{(0,0),(1,0),(1,1),(1,2),(1,3),(2,0),(2,2),(2,3)\} \text{ and}$$
$$B_2 = \{(0,0),(0,1),(0,2),(1,0),(1,1),(1,2),(2,0),(2,1),(2,2)\}.$$

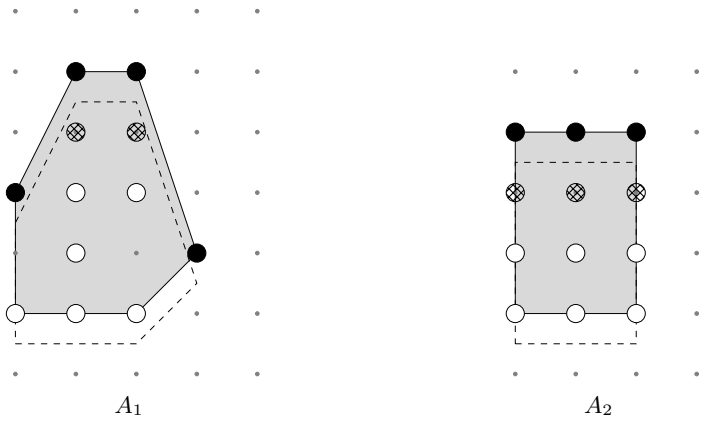



Figure 2.9: Supports $A_1$ and $A_2$ with offset points in the $y$-direction. 0.5-offset points are fully shaded and 1-offset points are crosshatched.

Working with the polyhedral geometry is a good place to start, but the lemma gives a trace affine linear set which may not be maximal. It may be that some elements classified as having nonlinear agency in the polyhedral geometry actually have independent or linear agency in practice. We compare trace affine linear sets estimated by polyhedral geometry to the maximal trace affine linear set in Section 3.3.

# Chapter 3

# Identifying Trace Affine Linear Sets

Let $\mathcal{F} = (f_1, \ldots, f_n) \in (\mathbb{C}[x_1, \ldots, x_n])^n$ be a sparse polynomial system over a square collection of supports $\mathcal{A} = (A_1, \ldots, A_n)$. In Sections 2.3 and 2.4 we identified trace affine linear subsets of the support by computing the sparse resultant and analyzing the formula for the trace given by the coefficients of the sparse resultant. However, we have seen that the computation of sparse resultants is not always feasible in practice. Additionally, we have identified a trace affine linear subset of the support using polyhedral geometry, but we wish to find maximal trace affine linear subsets.

We introduce a method for identifying a maximal trace affine linear set which uses homotopy continuation to avoid the use of hidden variable sparse resultants. We outline and implement the strategy. Finally we discuss some results.

## 3.1 Trace Affine Linear Identification with Homotopy

Given some subset of the support $\mathcal{B} \subseteq \mathcal{A}$, we construct a homotopy to track the solutions of the system by deforming the coefficients associated with elements of $\mathcal{B}$. We continue to use the function $t \mapsto \Sigma_1(\mathcal{V}^\times(\mathcal{F}_t))$ as introduced in Section 2.3. As before, $t$ affine linearly changes coefficients associated with elements in $\mathcal{B}$, but we now explicitly define the function using a homotopy. Then we compute the solutions of the system at some $t$-values along the homotopy. The trace of these

solution sets $\Sigma_1(\mathcal{V}^\times(\mathcal{F}_t))$ as a function of $t$ is linear if and only if $\mathcal{B}$ is a trace affine linear subset.

A schematic describing our use of homotopies is shown in Figure 3.1. Let $\mathcal{F}$ be a polynomial system supported on $\mathcal{A}$. Choose some subset of the support $\mathcal{B} \subseteq \mathcal{A}$. Let $H(x,t)$ be a straight-line homotopy from $\mathcal{F}$ to a system $\mathcal{G}$ such that coefficients associated with chosen support elements in $\mathcal{B}$ are deformed as $t$ goes from 1 to 0. $H$ is constructed so that the coefficients in $\mathcal{B}$ are deformed affine linearly. Then we compute the solutions of the system at the three $t$-values $t = 0$, $t = \frac{1}{2}$, and $t = 1$ along the homotopy, see Figure 3.1. Any differences between the three solution sets are due to the change in the coefficients indexed by $\mathcal{B}$, since everything else was left the same. We classify the relationship among the traces of the solution sets as constant, linear, or non-linear.

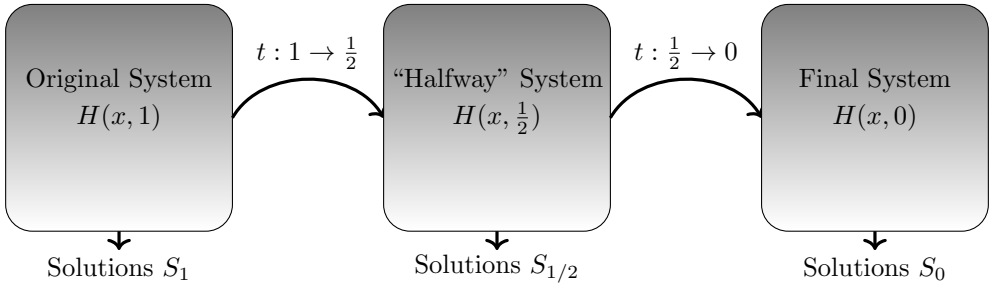



Figure 3.1: Schematic of a two-step homotopy at work. We obtain three solution sets from the homotopy and then compare the traces of those sets to determine linearity.

By Theorem 2.22, a subset $\mathcal{B}$ of the support $\mathcal{A}$ is a trace affine linear set if every element of $\mathcal{B}$ has independent or linear agency, and every pair of elements with linear agency has joint linear agency. Thus, the homotopy used does not need to deform all of the coefficients of the system at once. Instead, we use several homotopies or small subsets of $\mathcal{B}$ to classify behavior. To do this, we introduce three procedures. The Agency Classification Algorithm uses homotopy continuation to classify individual agency of support elements. The Joint Agency Classification Algorithm uses homotopy continuation to classify joint linear agency. And a sub-procedure, the Degree Test, determines the degree relationship among traces of the solution sets which result from homotopy continuation.

We determine a trace affine linear subset $\mathcal{B}$ by combining these three methods. Given a support $\mathcal{A}$, we perform the Agency Classification Algorithm, which makes use of the Degree Test, to determine the individual agencies of each support element. Then we collect all elements of $\mathcal{A}$ which have linear agency, and perform the Joint Agency Classification Algorithm on pairs of these elements to determine which pairs have joint linear agency, using the Degree Test. Finally, a maximal trace

affine linear set $\mathcal{B} \subseteq \mathcal{A}$ contains as many elements with independent and linear agency as possible without having any pairs of linear elements with nonlinear joint agency.

### 3.1.1 Degree Test

In order to determine agency and joint agency, we need to identify the degree relationship among a collection of traces. Lemma 3.2, the Degree Test Lemma, determines the degree of a univariate polynomial $f$ using the alternating sum of binomial coefficients. First, we introduce Lemma 3.1 to support the Degree Test Lemma.

**Lemma 3.1.** *Let $f(x)$ be a univariate polynomial of degree $d \geq 1$. Then $f(x) - f(x+1)$ is a polynomial of degree $d-1$.*

**Proof.** Let $f(x) = \sum_{i=0}^{d} c_i x^i$, where $c_i \in \mathbb{C}$ for all $i$. Then:

$$\begin{aligned}
f(x) - f(x+1) &= \sum_{i=0}^{d} c_i x^i - \sum_{i=0}^{d} c_i (x+1)^i \\
&= \sum_{i=0}^{d} c_i x^i - \sum_{i=0}^{d} c_i \left( \sum_{j=0}^{d} \binom{i}{j} x^j \right) \\
&= \sum_{i=0}^{d} c_i x^i - \sum_{j=0}^{d} \left( \sum_{i=j}^{d} c_i \binom{i}{j} \right) x^j \\
&= \sum_{j=0}^{d} \left( c_j - \sum_{i=j}^{d} c_i \binom{i}{j} \right) x^j \\
&= -\sum_{j=0}^{d} \left( \sum_{i=j+1}^{d} c_i \binom{i}{j} \right) x^j.
\end{aligned}$$

The $x^d$ monomial in this polynomial has a coefficient of 0 since setting $j = d$ pushes outside the index of the sum:

$$\left( \sum_{i=d+1}^{d} c_i \binom{i}{d} \right) x^d = 0 \cdot x^d.$$

Thus the polynomial $f(x) - f(x+1)$ has degree less than $d$. Additionally, the $x^{d-1}$ monomial has a nonzero coefficient:

$$\left( \sum_{i=d}^{d} c_i \binom{i}{d-1} \right) x^{d-1} = c_d d x^{d-1}.$$

Note that $c_d \neq 0$ since $f$ is of degree $d$, and $d \neq 0$ since $d \geq 1$. Thus $c_d d x^{d-1}$ is the leading term of $f(x) - f(x+1)$, and this polynomial has degree $d-1$. □

**Lemma 3.2** (Degree Test Lemma)**.** *Let $f(x)$ be a univariate polynomial of degree $d$.*

- *If $d < n$ then the polynomial $\sum_{k=0}^{n}(-1)^k\binom{n}{k}f(x+k)$ is the zero polynomial.*
- *If $d \geq n$ then $\sum_{k=0}^{n}(-1)^k\binom{n}{k}f(x+k)$ is not the zero polynomial.*

**Proof.** We prove the above by induction on $n$. Let $f(x)$ be a univariate polynomial of degree $d$.

*Base case.* Set $n = 1$. Then we have the following sum:

$$\sum_{k=0}^{1}(-1)^k\binom{1}{k}f(x+k) = f(x) - f(x+1).$$

If $d < 1$, the degree of $f$ is 0, so $f(x) = f(x+1)$ and the above is the zero polynomial. On the other hand, if $d \geq 1$, then $f(x) - f(x+1) \neq 0$ since the difference of the leading terms is nonzero. Let the leading term of $f(x)$ be $ax^d$, $a \neq 0$. Then the difference is $ax^d - a(x+1)^d = -adx^{d-1} + \cdots$, and since $a, d \neq 0$ we expect this to be nonzero.

*Induction step.* Suppose that $d < n$ implies $\sum_{k=0}^{n}(-1)^k\binom{n}{k}f(x+k) = 0$, while $d \geq n$ implies $\sum_{k=0}^{n}(-1)^k\binom{n}{k}f(x+k) \neq 0$. Consider the case for $n+1$, where we have the following sum:

$$\begin{aligned}
\sum_{k=0}^{n+1}(-1)^k\binom{n+1}{k}f(x+k) &= \sum_{k=0}^{n+1}(-1)^k\left(\binom{n}{k} + \binom{n}{k-1}\right)f(x+k) \\
&= \sum_{k=0}^{n+1}(-1)^k\binom{n}{k}f(x+k) + \sum_{k=1}^{n+1}(-1)^k\binom{n}{k-1}f(x+k) \\
&= \sum_{k=0}^{n}(-1)^k\binom{n}{k}f(x+k) + \sum_{k=0}^{n}(-1)^{k+1}\binom{n}{k}f(x+k+1) \\
&= \sum_{k=0}^{n}(-1)^k\binom{n}{k}(f(x+k) - f(x+k+1)),
\end{aligned}$$

where the first equality comes from Pascal's identity and the third equality is a result of reindexing the second sum.

Define $g(x) := f(x) - f(x+1)$. By Lemma 3.1 the degree of $g$ is $\hat{d} = d - 1$. Let the degree of $f$ be $d < n+1$, so that the degree of $g$ is $\hat{d} < n$. Then, by the induction assumption,

$$\sum_{k=0}^{n+1}(-1)^k\binom{n+1}{k}f(x+k) = \sum_{k=0}^{n}(-1)^k\binom{n}{k}g(x+k) = 0.$$

On the other hand, if $d \geq n+1$, then $\hat{d} \geq n$ so

$$\sum_{k=0}^{n+1}(-1)^k\binom{n+1}{k}f(x+k) = \sum_{k=0}^{n}(-1)^k\binom{n}{k}g(x+k) \neq 0.$$

Therefore, the sum $\sum_{k=0}^{n}(-1)^k\binom{n}{k}f(x+k)$ decides the degree $d$ of $f$. □

The procedure in Algorithm 3 heuristically calculates the degree of a fitting curve to the set of traces according to the sum in Lemma 3.2. The application of Lemma 3.2 determines whether the degree of the curve fitting the trace points is less than $i+1$. In particular, a $j$-step homotopy gives $j+1$ traces which have a degree one relationship if and only if the points $(0, \Sigma_1(S_0)), (1, \Sigma_1(S_1), \ldots, (j, \Sigma_1(S_j))$ are collinear. To check if the degree of the fitting curve is one, the Degree Test checks for each consecutive threesome of traces whether they fit the binomial alternating sum, that is, whether $\Sigma_1(S_j) - 2\Sigma_1(S_{j+1}) + \Sigma_1(S_{j+2}) = 0$. Then the test extends this calculation to higher degrees.

**Algorithm 3:** Degree Test

**Input:** • a list of lists of solutions along a homotopy $\{S_0, S_1, \ldots, S_n\}$
• a coordinate $m$
**Output:** degree of relationship between the traces of the solutions

```
1 Compute traces (Σ_m(S_0), Σ_m(S_1), ..., Σ_m(S_n))
2 for i ← 0, n − 1 do
3 |   for j ← 0, n − i − 1 do
  |   |   if Σ_{k=0}^{k=i+1} (−1)^k C(i+1, k) Σ_m(S_{j+k}) ≠ 0 then
  |   |   |   break
  |   if degree i fit for every set of j traces then
4 |   |   return i
5 return −1
```

The input to the Degree Test is a set of solution sets computed by the homotopy, along with a coordinate in which to compute the trace. The output of the Degree Test is the degree of the fitting curve to the set of traces, or $-1$ if no degree was found. We take $-1$ to signify an unknown, but nonlinear, relationship.

### 3.1.2 Individual Agency Classification

The next important procedure is the Agency Classification with Homotopy in Algorithm 4. The algorithm takes as input a support $\mathcal{A} = (A_1, \ldots, A_n)$ and a coordinate in which to test the

agency. Then the algorithm computes the agency of each individual element of the support using homotopy continuation and the Degree Test.

We use homotopy continuation to identify the way a set of solutions changes as we change the coefficient of a monomial. We construct a homotopy which deforms one coefficient of the original system, see Definition 1.26. As $t$ varies from 0 to 1, we take different values of $t$ as checkpoints. Each checkpoint along the homotopy corresponds to a system for which we compute the solutions. The sums of the solutions are then input into the Degree Test to classify agency of the support element associated with the deformed coefficient.

The Agency Classification Algorithm performs this procedure for every support element. The output lists the determined agency of each support element.

**Algorithm 4:** Agency Classification with Homotopy

**Input:** • a collection of supports $\mathcal{A} = (A_1, \ldots, A_n)$
• a ring $R[x_1, \ldots, x_n]$
• a coordinate $m$

**Output:** a list of degrees; each denotes the agency of an element of the support.

Generate generic polynomials $f_i \in R[x_1, \ldots, x_n]$ supported on each $A_i$
Construct a homotopy ring $S = R[x_1, \ldots, x_n][t]$
Compute initial solutions $S_0$ of $\mathcal{F} = (f_1, \ldots, f_n)$
**for** $i \leftarrow 1, n$ **do**
  **for** *each* $\alpha \in A_i$ **do**
    Select random $\beta \in \mathbb{C}$.
    Construct homotopy $H(\mathbf{x}, t)$ where $H(\mathbf{x}, 1) = \mathcal{F}$ and $H(\mathbf{x}, 0) = \mathcal{F} + j\beta\mathbf{x}^\alpha$.
    Collect $j + 1$ systems $H = \{\mathcal{F}, \mathcal{F} + \beta\mathbf{x}^\alpha, \ldots, \mathcal{F} + j\beta\mathbf{x}^\alpha\}$.
    Compute solutions $X = (S_0, S_1, \ldots, S_j)$ of $H$
    **if** *degreeTest*$(\{\Sigma_m(S_k)\}_{k=1}^{j}) = d$ **then**
      **list** $d$
    **else**
      **list** $-1$

### 3.1.3 Joint Agency Classification

In order to check that a given subset of elements $\mathcal{B} \subseteq \mathcal{A}$ is trace affine linear, it is not enough to check that each element has independent or linear agency. By Theorem 2.22, in order for $\mathcal{B}$ to be a trace affine linear subset, each pair of elements in $\mathcal{B}$ with linear agency must have joint linear agency.

We therefore introduce a procedure in Algorithm 5 to check for joint linear agency. The

main difference between this algorithm and Algorithm 4 is that in the joint case we only need to check pairs of support elements within $\mathcal{B}$ where both elements have linear agency, by Theorem 2.22. To accomplish the check between pairs, we construct a homotopy which deforms the coefficients of two monomials of the original system at once. As $t$ varies from 0 to 1, we again take different values of $t$ as checkpoints and compute the solutions of the systems at those checkpoints. Then the Degree Test determines the degree relationship of the traces of the solutions sets. If the degree is one, then the two support elements associated with the two deformed coefficients have joint linear agency.

The Joint Agency Classification Algorithm performs this procedure on every pair of support elements which individually have linear agency. Therefore, it is necessary to perform the Agency Classification Algorithm first in order to determine which elements have linear agency.

**Algorithm 5:** Joint Linear Agency Classification

**Input:** • a collection of supports $\mathcal{A}$
• a ring $R[x_1, \ldots, x_n]$
• a coordinate $m$

**Output:** a list of degrees; each denotes the joint agency of a pair

$\mathcal{L} = \{$pairs of elements in $\mathcal{A}$ with linear agency$\}$
Generate generic polynomials $f_i \in R[x_1, \ldots, x_n]$ supported on each $A_i$
Construct a homotopy ring $S = R[x_1, \ldots, x_n][t]$
Compute initial solutions $S_0$ of $\mathcal{F} = (f_1, \ldots, f_n)$
**for** *each pair* $(\alpha_1, \alpha_2) \in \mathcal{L}$ **do**
    Select random $\beta, \gamma \in \mathbb{C}$.
    Construct homotopy $H(\mathbf{x}, t)$ where $H(\mathbf{x}, 1) = \mathcal{F}$ and $H(\mathbf{x}, 0) = \mathcal{F} + j\beta\mathbf{x}^{\alpha_1} + j\gamma\mathbf{x}^{\alpha_2}$.
    Collect $j + 1$ systems $H = \{\mathcal{F}, \mathcal{F} + \beta\mathbf{x}^{\alpha_1} + \gamma\mathbf{x}^{\alpha_2}, \ldots, \mathcal{F} + j\beta\mathbf{x}^{\alpha_1} + j\gamma\mathbf{x}^{\alpha_2}\}$.
    Compute solutions $X = (S_0, S_1, \ldots, S_j)$ of $H$
    **if** *degreeTest*$(\{\Sigma_m(S_k)\}_{k=1}^{j}) = d$ **then**
        **list** $d$
    **else**
        **list** $-1$

## 3.2 Implementation

The three procedures from Section 3.1 are implemented in *Macaulay2* [9]. See the appendix for code. We present some examples and notes on implementation for each of the main algorithms.

We first demonstrate constructing a homotopy and obtaining solutions sets for different $t$-values along the homotopy, a process used by both Algorithms 4 and 5. The function `homotopyN` constructs a homotopy as follows: Let $\mathcal{F}(x) = H(x, 1)$ be the original system, and suppose we desire

three solution sets. From $t = 1$ to $t = \frac{1}{2}$, the homotopy multiplies the chosen coefficients of the system by a random complex number to obtain the "halfway" system $H(x, \frac{1}{2})$. Then the homotopy multiplies the coefficient attached to the same monomials again by the same complex number to obtain the final system $H(x, 0) = \mathcal{G}(x)$. The only difference between the original system and the final system is that the coefficients of the chosen monomials have been linearly deformed. Coefficients associated with all other monomials are left the same. The function `BertSolsN` tracks the solution set along the homotopy as $t$ changes to obtain the desired number of solution sets.

**Example 3.3.** Consider again $\mathcal{F} = (f, g)$ with support illustrated in Figure 2.3:

$$f = x^2y - 2xy + 3y - 4 \qquad \text{and} \qquad g = xy^2 - 2x^2 + 2y + 3.$$

We want to identify the agency of the element $(2, 1) \in A_1$. We construct a homotopy to deform the coefficient of the monomial $x^2y$ in $f$. Let $\alpha$ be some random complex number, which we call the *deformation constant*. In this example, we demonstrate a two-step homotopy, so we choose three $t$-values at which to compute solutions. Let $\mathcal{F}_0 = (f', g)$ where $f' = (1 + 2\alpha)x^2y - 2xy + 3y - 4$, which is the original system with the coefficient of the $x^2y$ monomial in $f$ increased by $2\alpha$. Then define the homotopy $H(x, y, t)$ such that

$$H(x, y, t) = (1 - t)\mathcal{F}(x, y) + t\mathcal{F}_0(x, y).$$

Now let $\mathcal{F}_t(x, y) = H(x, y, t)$ so that the start system is $\mathcal{F}_1 = \mathcal{F}$ and the target system is $\mathcal{F}_0$. The homotopy continuously and affine linearly deforms the coefficient of the monomial $x^2y$ in $f$. The middle system is $\mathcal{F}_{1/2} = \frac{1}{2}\mathcal{F}(x, y) + \frac{1}{2}\mathcal{F}_0(x, y) = (f'', g)$ where $f'' = (1 + \alpha)x^2y - 2xy + 3y - 4$, which is the original system with the coefficient of the monomial $x^2y$ in $f$ increased by $\alpha$. We use the software package *Bertini* [1] to track the solutions of the start system $\mathcal{F}_1$ to the new solutions of $\mathcal{F}_{1/2}$ and $\mathcal{F}_0$ via homotopy continuation. The resulting three solution sets are $S_1$, $S_{1/2}$, and $S_0$, as in the schematic in Figure 3.1.

For example, letting $\alpha = 9.677744 + 5.4167144i$ gives the homotopy

$$\mathcal{F}_t = (2(9.677744 + 5.4167144i)x^2yt + x^2y - 2xy + 3y - 4, xy^2 - 2x^2 + 2y + 3).$$

The function `homotopyN` constructs the homotopy as a list of two systems, one for each desired

solution set (not including the solutions of the original system).

```
H =  homotopyN(f, g, x^2*y, t, alpha, 2)
>>{{(- 9.67774 - 5.41671*ii)x^2 y*t + (10.6777 + 5.41671*ii)x^2 y - 2x*y + 3y
- 4, x*y^2  - 2x^2  + 2y + 3},
       {(- 9.67774 - 5.41671*ii)x^2 y*t + (20.3555 + 10.8334*ii)x^2 y - 2x*y
       + 3y - 4, x*y^2  - 2x^2  + 2y + 3}}
```

The function `BertSolsN` performs homotopy continuation using *Bertini* to track solution points between the original system and the two constructed systems. Then the output of `BertSolsN(H,sols,s,t)` are the three solution sets listed in Table 3.1.

| $S_1$ | $S_{1/2}$ | $S_0$ |
|---|---|---|
| $.438696 - 1.25604i$, | $1.30183 - .036502i$ | $1.26535 - .020743i$ |
| $2.09486$ | $-1.28046 + .017119i$ | $-1.25883 + .013447i$ |
| $1.192 + 1.91039i$ | $.114215 + .459048i$ | $.0846946 + .33903i$ |
| $.438696 + 1.25604i$ | $.274827 + .593957i$ | $.184299 + .439915i$ |
| $-1.35625$ | $-.018269 - .70639i$ | $-.043743 - .507199i$ |
| $1.192 - 1.91039i$ | $-.0942076 - .478374i$ | $-.0786426 - .345948i$ |
| $\Sigma_1(S_1) = 4$ | $\Sigma_1(S_{1/2}) = .29793 - .15114i$ | $\Sigma_1(S_2) = .15313 - .08149i$ |

Table 3.1: Sets of solutions of systems along the homotopy for three $t$-values. We list the $x$-values of the solutions and first coordinate traces.

Example 3.3 demonstrates the use of `homotopyN` and `BertSolsN` to deform the coefficient of one monomial, which is how these functions are implemented within the Agency Classification Algorithm, Algorithm 4. Within the Joint Agency Classification Algorithm, Algorithm 5, these same functions are used to deform coefficients of two monomials at once.

Next we discuss implementation of the Degree Test from Algorithm 3. We continue with the same system as in Example 3.3.

**Example 3.4.** Previously, we defined a homotopy which continuously deformed the coefficient associated with $x^2y$ in $f$. Table 3.1 lists the three solution sets along this homotopy when we select three $t$-values. The set of three solution sets $\mathcal{S}$ is the input for `degreeTest`, along with a coordinate in which to take the trace. In this example we compute the trace in the $x$-coordinate, implemented by `degreeTest(S, 0)`.

The $x$-coordinate traces are: $\Sigma_1(S_1) = 4$, $\Sigma_1(S_{1/2}) = .29793 - .15114i$, and $\Sigma_1(S_0) = .15313 - .08149i$. There are only three traces, so the Degree Test checks for relationships of degrees

$i = 0$ and $i = 1$. There is a change between the calculated traces, hence $\Sigma_1$ is not a constant function of $t$ and the element $(2, 1)$ in the support cannot have independent agency. That is, $\Sigma_1(S_1) - \Sigma_1(S_{1/2}) = 4 - (.29793 - .15114i) \neq 0$.

Next `degreeTest` checks if the relationship between the traces is linear according to Lemma 3.2. The Degree Test computes Equation (3.1), and decides that the relationship between these three calculated traces is not linear.

$$\begin{aligned}\Sigma_1(S_1) - 2\Sigma_1(S_{1/2}) + \Sigma_1(S_0) &= 4 - 2(.29793 - .15114i) + .15313 - .08149i \\ &= 3.55727 + .22079i \quad (3.1)\end{aligned}$$

Neither degree $i$ checked by the Degree Test was successful, so the output of `degreeTest(S, 0)` is $-1$, which signifies that the degree relationship among the traces is nonlinear. Since the only feature that was changed to get these different traces was the coefficient of the monomial $x^2y$ in $f$, we conclude that $(2, 1) \in A_1$ has nonlinear agency.

Algorithm 4, the Agency Classification with Homotopy, is implemented by the function `traceTestN`. When we implement this procedure, we write each support as a matrix whose columns are elements of the support. Note that each support matrix has the same number of rows, since there are as many rows as variables in the system. However, supports may not all have the same number of columns because some supports may be larger than others. Additionally, we include a tolerance with the inputs to the function. The tolerance is a small number which is used to account for numerical error.

The output of `traceTestN` is a list of integers greater than or equal to $-1$ corresponding to the agency of each element of the support. If the output is non-negative, then it is the degree relationship of the trace for that element of the support. For example, 0 denotes independent agency, 1 denotes linear agency, and 2 or above denotes nonlinear agency. An output of $-1$ indicates that the element has nonlinear agency, but without a degree estimate. We illustrate the implementation by continuing the previous example.

**Example 3.5.** The system $\mathcal{F} = (f, g)$ from Example 2.15 has support $\mathcal{A} = (A_1, A_2)$, where $A_1 = \{(2, 1), (1, 1), (0, 1), (0, 0)\}$ and $A_2 = \{(1, 2), (2, 0), (0, 1), (0, 0)\}$. The function `traceTestN` describes the agency of elements in $\mathcal{A}$, so the results apply to generic systems in $\mathbb{C}^{\mathcal{A}}$. Written as

matrices, the supports are

$$M_1 = \begin{bmatrix} 0 & 0 & 1 & 2 \\ 0 & 1 & 1 & 1 \end{bmatrix} \qquad \text{and} \qquad M_2 = \begin{bmatrix} 0 & 0 & 2 & 1 \\ 0 & 1 & 0 & 2 \end{bmatrix}. \tag{3.2}$$

Then the input to `traceTestN` is the list $\{M_1, M_2\}$. In this example, we set the tolerance equal to $10^{-4}$, we use a two-step homotopy, and we compute the trace in the $x$-coordinate. The procedure `traceTestN(listM, R, 1e-4, 0 , 2)` first generates a random polynomial system supported on $\mathcal{A}$. As part of this process, it creates a list of monomials which could appear in $\mathcal{F}$ in order of total degree and support. The list of monomials for $\{M_1, M_2\}$ in order is $L = \{\{1, y, xy, x^2y\}, \{1, y, x^2, xy^2\}\}$.

Then for each element $a_{i,j} \in \mathcal{A}$, a homotopy is constructed and solved as in Example 3.3, and the agency of $a_{i,j}$ is determined as in Example 3.4. The agencies are denoted by integers, displayed in a list whose order is the same as the order of elements in the support in $L$. In the output

```
traceTestN(\{M1, M2\}, R, 1e-4, 0 , 2)
>> {{0, 0, 1, -1}, {0, 0, 0, 0}},
```

the entry of 1 corresponds to the $xy$ monomial from $f_1$, and the entry of $-1$ corresponds to the $x^2y$ monomial from $f_2$. That is, $(1, 1)$ has linear agency, $(2, 1)$ has nonlinear agency, and all other points have independent agency.

In addition to the printed and listed output, the function `tikzOutput` takes as input the list of agencies and outputs an encoding of the Newton polytope with agency shaded according to the list. The picture generated by `tikzOutput` for the list $\{\{0, 0, 1, -1\}, \{0, 0, 0, 0\}\}$ is Figure 3.2. The Newton polytope with shaded support elements aids in the display and interpretation of support agency.

The function `TALTestN` combines the processes of both Algorithms 4 and 5. Similar to `traceTestN`, we input the support of the system as matrices. The function `traceTestN` is called as part of `TALTestN`, in order to obtain individual agency classification of the support. Then a similar procedure checks joint agency, but only for those pairs of elements which were found to have linear agency by `traceTestN`.

In the printed output of `TALTestN`, each pair is written as a sequence containing the two monomials, along with numbers designating which polynomial each monomial came from. For

example, the printed output `{(x^2, 1), (x*y^3 , 2)} observed effect is linear` means that the monomial $x^2$ comes from the first polynomial $f$ and the monomial $xy^3$ comes from the second polynomial $g$. Then the elements $(2,0) \in A_1$ and $(1,3) \in A_2$ have joint linear agency. The function also outputs a list of degrees associated with the joint agency of the pairs.

## 3.3 Results

In this section we work through a few examples. Each example illustrates an aspect of Algorithm 4 as well as theory from [5].

**Example 3.6.** Let $\mathcal{F} = (f, g)$ be the simple system from Example 2.15, where:

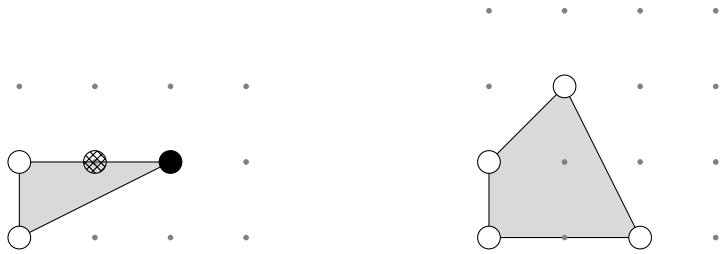
$$f = x^2y - 2xy + 3y - 4 \qquad \text{and} \qquad g = xy^2 - 2x^2 + 2y + 3$$

In Example 2.16, we generated a classification of the support with shading as in Figure 2.5. We aim to generate the same shading using our new method of trace test with homotopy continuation.

We consider generic polynomial systems with the same support as $\mathcal{F}$. The support of $f$ is $A_1 = \{(2,1),(1,1),(0,1),(0,0)\}$ and the support of $g$ is $A_2 = \{(1,2),(2,0),(0,1),(0,0)\}$. We investigate the trace in the $x$-coordinate. Running Algorithm 4 on $\mathcal{A} = (A_1, A_2)$ gives the individual classifications seen in Table 3.2 and Figure 3.2. Let $\mathcal{B}$ be the collection of elements in $\mathcal{A}$ which have linear or independent agency. At this stage, we check for joint linear agency, using Algorithm 5, to obtain the final trace affine linear set $\mathcal{B} \subseteq \mathcal{A}$. In this case all pairs in $\mathcal{B}$ have joint linear agency, so $\mathcal{B}$ is trace affine linear.

| **Monomial** | **Degree Output** | **Classification** |
|---|---|---|
| **Polynomial:** $-10x^2y - 3xy + 2y - 3$ | | |
| 1 | 0 | constant |
| $y$ | 0 | constant |
| $xy$ | 1 | linear |
| $x^2y$ | $-1$ | non-linear |
| **Polynomial:** $3xy^2 - 5x^2 + y - 10$ | | |
| 1 | 0 | constant |
| $y$ | 0 | constant |
| $x^2$ | 0 | constant |
| $xy^2$ | 0 | constant |

Table 3.2: Table listing agency in the $x$-coordinate for each element of the support from Equation (3.2).



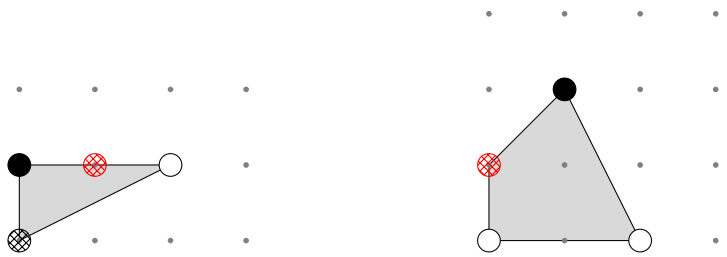
Support of $f_1$ Support of $f_2$

Figure 3.2: Support from Equation (3.2) with agency classification according to the $x$-coordinate.

Thus a trace affine linear subset of the support is $\mathcal{B} = (A_1 \setminus \{x^2y\}, A_2)$. The classification is different in the $y$-coordinate, as we see in Table 3.3 and the associated shading in Figure 3.3. In the $y$-coordinate case, the pairwise agency check has output `{(xy,1), (y, 2)} observed effect is non-linear` and reveals that the pair of elements $(1,1) \in A_1$ and $(0,1) \in A_2$ cannot be together in a trace affine linear set.

| **Monomial** | **Degree Output** | **Classification** |
|---|---|---|
| **Polynomial:** $-10x^2y - 3xy + 2y - 3$ | | |
| 1 | 1 | linear |
| $y$ | $-1$ | non-linear |
| $xy$ | 1 | linear |
| $x^2y$ | 0 | constant |
| **Polynomial:** $3xy^2 - 5x^2 + y - 10$ | | |
| 1 | 0 | constant |
| $y$ | 1 | linear |
| $x^2$ | 0 | constant |
| $xy^2$ | $-1$ | non-linear |

Table 3.3: Table listing agency in the $y$-coordinate for each element of the support of a simple from Equation (3.2).



Support of $f_1$ Support of $f_2$

Figure 3.3: Support from Equation (3.2) with agency classifcation according to the $y$-coordinate. Elements that are crosshatched and color red have individual linear agency but nonlinear joint agency, so both cannot be included in a maximal trace affine linear set.

In Example 3.6 there are a few differences exhibited from the polyhedral geometry strategy in Example 2.26. The trace affine linear set estimated by the polyhedral geometry does not include elements $(2,0)$ and $(1,2)$ from $A_2$, since these two elements were classified by the geometry to have potential nonlinear agency. However, by using the agency classifying algorithm, we see that a trace affine linear set for the trace in the first coordinate can include these two elements because they both have independent agency.

**Example 3.7.** Consider the system $\mathcal{F} = (f, g) \in (\mathbb{C}[x, y])^2$ over the support from Example 2.28, $\mathcal{A} = (A_1, A_2)$ where:

$$A_1 = \{(0,0), (0,2), (1,0), (1,1), (1,2), (1,3), (1,4), (2,0), (2,2), (2,3), (2,4), (3,1)\}$$
$$A_2 = \{(0,0), (0,1), (0,2), (0,3), (1,0), (1,1), (1,2), (1,3), (2,0), (2,1), (2,2), (2,3)\}$$

The universal polynomials $f$ and $g$ supported on $\mathcal{A}$ are:

$$f = a_0x^3y + a_1x^2y^4 + a_2x^2y^3 + a_3x^2y^2 + a_4x^2 + a_5xy^4 + a_6xy^3 + a_7xy^2 + a_8xy + a_9x + a_{10}y^2 + a_{11},$$
$$g = b_0x^2y^3 + b_1x^2y^2 + b_2x^2y + b_3x^2 + b_4xy^3 + b_5xy^2 + b_6xy + b_7x + b_8y^3 + b_9y^2 + b_{10}y + b_{11}$$

We compute the individual agencies of elements in the support using Algorithm 4. The results can more clearly be seen in Table 3.4 as well as depicted on the Newton polytope in Figure 3.4. Then let $\mathcal{B}$ be the collection of elements which have independent or linear agency.

Support of $f_1$ Support of $f_2$

Figure 3.4: Support from Example 3.7 classified according to agency in the $x$-coordinate. One element with linear agency which may be removed to obtain a trace affine linear set is colored red.

We focus on the trace in the $x$-coordinate. To verify $\mathcal{B}$ is a trace affine linear subset, we also apply Algorithm 5 to this example. Out of 21 possible pairs of elements with linear agency in

| **Monomial** | **Degree Output** | **Classification** |
|---|---|---|
| **Polynomial:** $f = 10x^2y^4 + 4x^2y^3 - 7x^3y - 6x^2y^2 + 3xy^3 + 5x^2 + 9xy + 2y^2 - 5x + 1$ | | |
| $1$ | $0$ | constant |
| $y^2$ | $0$ | constant |
| $x$ | $0$ | constant |
| $xy$ | $0$ | constant |
| $xy^2$ | $0$ | constant |
| $xy^3$ | $0$ | constant |
| $xy^4$ | $0$ | constant |
| $x^2$ | $1$ | linear |
| $x^2y^2$ | $1$ | linear |
| $x^2y^3$ | $1$ | linear |
| $x^2y^4$ | $1$ | linear |
| $x^3y$ | $-1$ | non-linear |
| **Polynomial:** $g = 6x^2y^3 + 4x^2y^2 - 4xy^3 - 4x^2y + 7xy^2 - 3y^3 - 6x^2 + 8xy + 9y^2 - 7x + 2y + 3$ | | |
| $1$ | $0$ | constant |
| $y$ | $0$ | constant |
| $y^2$ | $0$ | constant |
| $y^3$ | $0$ | constant |
| $x$ | $1$ | linear |
| $xy$ | $0$ | constant |
| $xy^2$ | $0$ | constant |
| $xy^3$ | $0$ | linear |
| $x^2$ | $-1$ | non-linear |
| $x^2y$ | $1$ | linear |
| $x^2y^2$ | $-1$ | non-linear |
| $x^2y^3$ | $-1$ | non-linear |

Table 3.4: Table listing agency in the $x$-coordinate for each element of the support $\mathcal{A}$.

$\mathcal{B}$, the algorithm identifies three pairs which do not have joint linear agency. Those are $(2,0) \in A_1$ and $(2,1) \in A_2$, $(2,3) \in A_1$ and $(2,1) \in A_2$, and $(2,4) \in A_1$ and $(2,1) \in A_2$. Note that $(2,1) \in A_2$ is in each of these pairs, so in order for $\mathcal{B}$ to be trace affine linear it must not contain $(2,1) \in A_2$, or none of the other three mentioned elements.

We omit the table for the $y$-coordinate, but the results can be seen in the shading of the support in the Newton polytope in Figure 3.5.

Support of $f_1$ Support of $f_2$

Figure 3.5: Support from Example 3.7 classified according to agency in the $y$-coordinate.

There are some cases when every element of the support is classified as having independent agency. According to [5], this may be if a system has a *lacunary* support. Additionally, as the next example shows, some subset of the support may be lacunary.

**Definition 3.8.** Let $\mathcal{A}$ be a collection of supports and $L[\mathcal{A}]$ the lattice of integer points represented in the support. $\mathcal{A}$ is **lacunary** if the index $[\mathbb{Z}^n : L[\mathcal{A}]]$ is greater than 1.

**Example 3.9.** Consider the following polynomial system in $(\mathbb{C}[x,y])^2$:

$$f = 9x^3y - 3x^2 + xy - 2y + 4 \qquad \text{and} \qquad g = 4x^3 - 4x^2y - 5x + 7$$

The support $\mathcal{A} = (A_1, A_2)$ written in matrix form is:

$$M_1 = \begin{bmatrix} 0 & 0 & 1 & 2 & 3 \\ 0 & 1 & 1 & 0 & 1 \end{bmatrix} \text{ and } M_2 = \begin{bmatrix} 0 & 1 & 2 & 3 \\ 0 & 0 & 1 & 0 \end{bmatrix}$$

At first glance, the support of this system is not lacunary, since the lattice generated by difference between points is $\mathbb{Z}^2$. However, when we remove elements which have independent agency according to the polyhedral geometry, the remaining support is lacunary. By Lemma 2.27, each element of $\mathcal{A}$

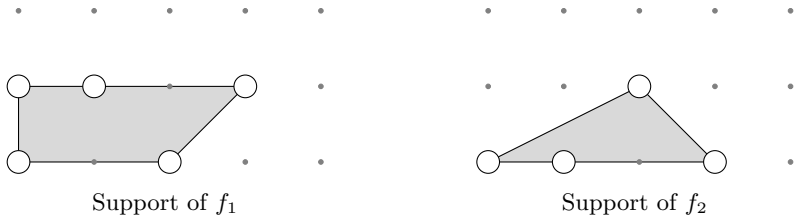



Figure 3.6: Support of $\mathcal{F}$ shaded according to agency in the $x$-coordinate. Every element in the support has independent agency.


Figure 3.7: Support of $\mathcal{F}$ with all but $1-$offset points removed. The remaining support elements form a lattice which has an index within $\mathbb{Z}^2$ not equal to one. Thus, the set is lacunary.

that is not a $1-$offset point in the $x$-direction has independent agency. The remaining support, depicted in Figure 3.7, is lacunary since points in the lattice generated by the remaining support are of the form $\{(a, b) : a + b \text{ is even}\}$. Then the index of this lattice within $\mathbb{Z}^2$ is greater than one. In this lacunary system, every element in the support is found to have an independent agency in the $x$-coordinate, see Figure 3.6. However, the system is not lacunary in the $y$-direction, and in the $y$-coordinate there are some linear and nonlinear classifications, as we see in Figure 3.8.

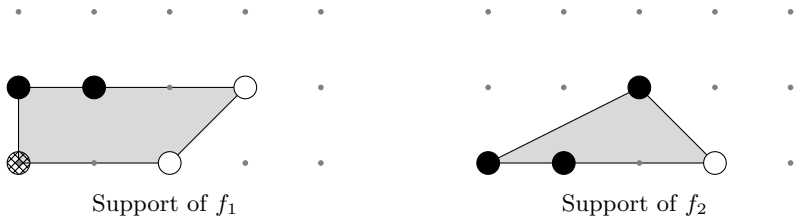



Figure 3.8: Classification of the lacunary system's support in the $y$-coordinate.

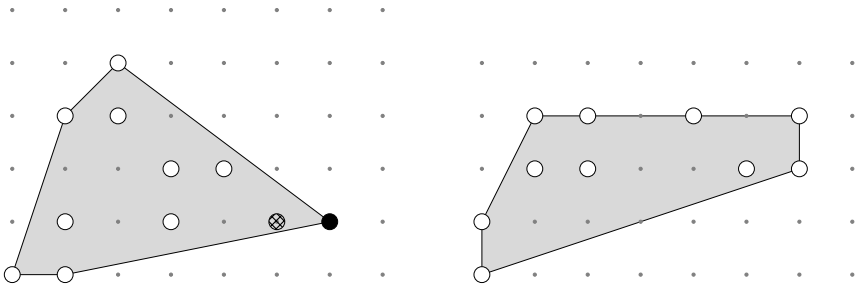
Next we consider an example where the difference between the estimation done by polyhedral geometry and the computed result are very different.

**Example 3.10.** Consider $\mathcal{F} = (f_1, f_2) \in (\mathbb{C}[x, y])^2$ supported on $\mathcal{A} = (A_1, A_2)$ such that the support

written in matrix form is:

$$M_1 = \begin{bmatrix} 0 & 1 & 1 & 2 & 1 & 3 & 3 & 4 & 2 & 5 & 6 \\ 0 & 0 & 1 & 3 & 3 & 1 & 2 & 2 & 4 & 1 & 1 \end{bmatrix} \quad \text{and} \quad M_2 = \begin{bmatrix} 0 & 0 & 1 & 1 & 2 & 2 & 4 & 5 & 6 & 6 \\ 0 & 1 & 2 & 3 & 2 & 3 & 3 & 2 & 2 & 3 \end{bmatrix}$$

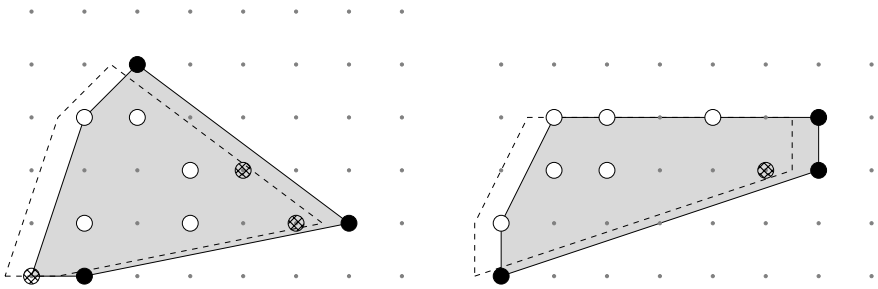
This support is on the larger side when it comes to computing a hidden variable sparse resultant. It is one example where waiting for the computation to finish is unreasonable. Applying our Algorithm 4 to this support, however, provides insight on the agency of elements of the support, as seen in Figure 3.10. There is only one element with linear agency and only one element with nonlinear agency.



Support of $f_1$. Support of $f_2$.

Figure 3.9: Support of a large system classified according to agency in the $x$-coordinate due to the Agency Classification Algorithm.

On the other hand, if we investigate this support by using the polyhedral geometry discussed in Section 2.4, we get Figure 3.10. The polyhedral geometry estimates a much smaller trace affine linear set, since there are many 0.5-offset points which actually have independent agency.



Support of $f_1$. Support of $f_2$.

Figure 3.10: Support of a large system with agency due to polyhedral geometry in the $x$-coordinate.

# Appendices

# Appendix A Individual Agency Classification

The following code was written in *Macaulay2* [9]. We used the package *Bertini* [1].

The function which classifies individual agency is called `traceTestN`. The visual outputs of the Newton polytopes seen in this paper were generated by the function `tikzOutput`. The rest of the functions listed here build up to the individual agency classification. A main subprocedure is `degreeTest`, which implements the Degree Test Lemma.

```
needsPackage "Bertini"


homotopyN = (f1, f2, mon, t, sclr, N) -> (
    --INPUT
    -- f1, f2: two polynomials
    -- mon: a monomial in the support of f1
    -- t: homotopy variable
    -- sclr: scalar to adjust size of homotopy change
    -- N: number (ge 2) of homotopies to define
    --OUTPUT
    -- a list containing N homotopy lists
    S:= ring t;
    -- construct a homotopy
    -- record N systems along the homotopy in the following list
    X:= for i in 0..N-1 list(
        {sub(f1,S) + i*sclr*mon + (1-t)*sclr*mon, sub(f2,S)});
    X
)


BertSolsN = (X,sols,s,t) -> (
    --INPUT
    --X: a list of homotopies of the initial polynomials
    --sols: initial solution set
    --s,t: homotopy variables
```

```
    --OUTPUT
    -- a list of lists of solutions
    tempsol:= sols;
    listSols:= for H in X list(
         -- this line performs the homotopy continuation of the solutions
         tempsol= bertiniUserHomotopy(s,{t=>s},H,tempsol, \\
             BertiniInputConfiguration=>{FinalTol=>1e-50},M2Precision=>1000)
    );
    prepend(sols, listSols)
)

degreeTest = (homSols, tol, coord) ->(
    -- INPUT
    -- homSols: a list of lists of homotopy solutions, including initial solution
    -- tol: desired tolerance level, typically 1e-4
    -- coord: coordinate in which to test the degree, 0 is first coordinate
    -- OUTPUT
    -- degree of relationship between traces, if there are at least two more \\
         traces calculated than degree
    -- or -1, if not enough traces to calculate a degree that high
    local coordSum;
    --add each set of solutions together to get a list of traces
    T := apply(homSols, i->sum apply(i, j->(coordinates j)_coord));
    for i in 0..#T-2 do(
        --if the flag stays true throughout entire loop then the degree fits
        flag:= true;
        --to check if the relationship is degree i, check all adjacent combos \\
            of i+2 traces
        for j in 0..#T-i-2 do(
            --we check whether all combinations meet the binomial pattern
            coordSum= sum toList apply(0..i+1, k->(-1)^k*binomial(i+1,k)*T_(j+k));
```

```
            if not clean_tol coordSum == 0 then (flag = false; break)
        );
        if flag then return i
    );
    -1
)

supportToMonomials = (M,R) ->(
    --INPUT:
    --M: a matrix containing the support for some generic polynomial
    --R: a ring whose variables are the variables of the polynomial
    --OUTPUT:
    --a list of monomials supported by M (multivar of R raised to multideg in M)
    for i in 0..numColumns M-1 list(
        product for j in 0..numRows M - 1 list(
            (vars R)_(0,j)^(M_(j,i))
        )
    )
)

supportToPolynomial = (M,R) ->(
    --INPUT:
    --M: a matrix containing the support for some generic polynomial
    --R: a ring whose variables are the variables of the polynomial
    --OUTPUT:
    --a polynomial supported on M with random coefficients between -10 and 10
    L:= supportToMonomials(M,R);
    --add together monomials from list with random coefficients
    sum apply(L, i-> i*random(-10.,10.))
)
```

```
traceTestN = (listM, R, tol, coord, N) ->(
    --INPUT:
    --listM: a list of matrices each containing the support for some polynomial
    --R: a ring whose variables are the variables of the polynomials
    --tol: desired tolerance level, typically 1e-4
    --coord: coordinate in which to test the degree, 0 is the first coordinate
    --N: number (ge 2) of homotopies to define
    --OUTPUT:
    --flagList: a list of agencies; 0 if independent, 1 if linear, \\
          -1 if non-linear
    --prints statements describing these degrees as effects of the monomial \\
      on the polynomial
    -- first write the matrix of support as a polynomial with random coefficients
    f1:= supportToPolynomial(listM_0, R);
    f2:= supportToPolynomial(listM_1, R);
    -- next define an extended ring S with homotopy parameters t and s
    S := CC[first entries vars R|{t,s}];
    -- solve the system using Bertini
    sols:= bertiniZeroDimSolve({f1,f2},BertiniInputConfiguration=>\\
        {FinalTol=>1e-50},M2Precision=>1000);
    -- set up a counter to consider support of each polynomial separately
    i:= 0;
    --in flagList we store flags signifying the degree effect of each monomial
    local flagList;
    -- choose a random complex number to be deformation scalar
    sclr = 10*random CC;
    while norm(sclr) < 6 do sclr = 10*random CC;
    -- in the outer for loop, run through once for each polynomial
    for f in {f1,f2} list(
        << endl;
        -- trick to consider both combinations of polynomials
```

```
        p1 = f;
        p2 = f1 + f2 - f;
        << "For poly " << f << endl;
        -- In the inner for loop, run through once for each monomial in the
        -- support of given polynomial. Save flag of each monomial in flagList.
        flagList = for mon in supportToMonomials(listM_i,S) list(
            << endl;
            -- define a homotopy that changes only the given polynomial
            H = homotopyN(p1,p2,mon, t, sclr, N);
            -- solve the system at each connection point in the homotopy
            homSols = BertSolsN(H, sols, s,t);
            -- find the degree that fits the changes in these solutions
            -- print corresponding signifying statements
            flag = degreeTest(homSols, tol, coord);
            if flag==0 then (<< mon << " observed effect is constant " << endl)
            else if flag==1 then (<< mon << " observed effect is linear"  << endl)
            else (<< mon << " observed effect is non-linear" << endl);
            flag);
        -- once finished with one polynomial and its support, move on to the next
        i= i+1;
        flagList
    )
)

tikzOutput = (listM, L) ->(
    --INPUT
    --listM: a list of matrices each containing the support for some polynomial
    --L: a list of flags, one for each element of each support, denoting degree
    --OUTPUT
    --LaTeX code that formats a lattice of the support colored for the degree
    out = "\\begin{figure}\n\\centering\n";
```

```
    --each polynomial has its own support and lattice
    for j in 0..length listM-1 do(
        out = out | "\\begin{subfigure}[b]{0.4\\}\n\\centering\n";
        out = out | "\\begin{tikzpicture}\n" | "\\foreach\\i in {0,...," | 1 + \\
            max first entries listM_j  |"}{";
        out = out | "\\foreach\\j in {0,...," | 1+ max last entries listM_j |"} \\
            {\\filldraw[color=gray] (\\i,\\j) circle (.03);}}";
        for i in 0..numColumns listM_j -1 do(
            out = out | "\\filldraw";
            --constant effect are left white
            if L_j_i==0 then
                out = out | "[fill = white]";
            --linear effect are crosshatched
            if L_j_i ==1 then
                out = out | "[pattern = crosshatch]";
            --default is filled in with black (non-linear)
            --any flag that is not 0 or 1 will result in the default
            out = out | " ("
                | listM_j_(0,i) | "," | listM_j_(1,i)
                | ") circle(.15);\n";
        );
        out = out | "\\end{tikzpicture}\n";
        out = out | "\\caption*{Support of $f_" | j+1 | "$}\n\\end{subfigure}\n";
    );
    out = out | "\\end{figure}";
    << out
)
```

# Appendix B Joint Agency Classification

The main function in this section is the joint agency classification function, which is called `TALTestN`. The other functions listed here support the joint agency classification.

```
pairHomotopy1 = (f1, f2, mon1, mon2, t, sclr, N) -> (
    --INPUT
    -- f1, f2: two polynomials
    -- mon1, mon2: distinct monomials in the support of f1
    -- t: homotopy variable
    -- sclr: scalar to adjust size of homotopy change
    -- N: number (ge 2) of homotopies to define
    --OUTPUT
    -- a list containing N homotopy lists
    S:= ring t;
    X:= for i in 0..N-1 list(
        {sub(f1,S) + i*sclr*mon1 + (1-t)*sclr*mon1 + i*sclr*mon2 \\
            + (1-t)*sclr*mon2, sub(f2,S)});
    X
)


pairHomotopy12 = (f1, f2, mon1, mon2, t, sclr, N) -> (
    --INPUT
    -- f1, f2: two polynomials
    -- mon1, mon2: distinct monomials in the support of f1
    -- t: homotopy variable
    -- sclr: scalar to adjust size of homotopy change
    -- N: number (ge 2) of homotopies to define
    --OUTPUT
    -- a list containing N homotopy lists
    S:= ring t;
    X:= for i in 0..N-1 list(
```

```
        {sub(f1,S) + i*sclr*mon1 + (1-t)*sclr*mon1, sub(f2,S) + \\
            i*sclr*mon2 + (1-t)*sclr*mon2});
    X
)

pairHomotopy2 = (f1, f2, mon1, mon2, t, sclr, N) -> (
    --INPUT
    -- f1, f2: two polynomials
    -- mon1, mon2: distinct monomials in the support of f1
    -- t: homotopy variable
    -- sclr: scalar to adjust size of homotopy change
    -- N: number (ge 2) of homotopies to define
    --OUTPUT
    -- a list containing N homotopy lists
    S:= ring t;
    X:= for i in 0..N-1 list(
        {sub(f1,S), sub(f2,S) + i*sclr*mon1 + (1-t)*sclr*mon1 + \\
            i*sclr*mon2 + (1-t)*sclr*mon2});
    X
)

linearAgencyToMonomialPairs = B ->(
    --INPUT:
    --B: a set containing only elements with linear agency
    --R: a ring
    --OUTPUT:
    --a list of all possible monomial pairings within B
    --pairs are listed with designation of which support they came from
    monPairs = {};
    for i in 0..length B-1 do(
        for j in 0..length B-1 when not j==i do(
```

```
            monPairs = insert(0,{B_i,B_j},monPairs)
        );
    );
    monPairs
)

TALTestN = (listM, R, tol, coord, N) ->(
    --INPUT:
    --listM: a list of matrices each containing the support for some polynomial
    --R: a ring whose variables are the variables of the polynomials
    --tol: desired tolerance level, typically 1e-4
    --coord: coordinate in which to test the degree, 0 is the first coordinate
    --N: number (ge 2) of homotopies to define
    --OUTPUT:
    --degreeList: a list of degrees of each joint relationship to trace
    --prints statements describing these degrees as effects of the \\
        monomial pair on the trace
    -- first write the matrix of support as a polynomial with random coefficients
    f1:= supportToPolynomial(listM_0, R);
    f2:= supportToPolynomial(listM_1, R);
    -- next define an extended ring S with homotopy parameters t and s
    S := CC[first entries vars R|{t,s}];
    -- solve the system using Bertini
    sols:= bertiniZeroDimSolve({f1,f2},BertiniInputConfiguration=>\\
        {FinalTol=>1e-50},M2Precision=>1000);
    -- set up an empty TAL set
    B := {};
    -- in degreeList, store degrees which are the effect of each monomial pair
    local degreeList;
    -- choose a random complex number to be deformation scalar
    sclr = 10*random CC;
```

```
while norm(sclr) < 6 do sclr = 10*random CC;
--make a list of all monomials in same order as agencies
monos := {supportToMonomials(listM_0,S), supportToMonomials(listM_1,S)};
--get the individual agencies and add only the linear ones to B
agencies := traceTestN(listM, R, tol, coord, N);
for i in 0..length agencies-1 do(
    for j in 0..length agencies_i-1 do(
        if (agencies_i)_j == 1 then B = insert(0, ((monos_i)_j, i+1), B)
    );
);
-- list all pairs possible among elements with linear agency
monPairs:= linearAgencyToMonomialPairs B;
-- check for joint agency among each pair
degreeList = for monPair in monPairs list(
    << endl;
    -- define a homotopy that changes only in selected monomials
    mon1 := sub((monPair_0)_0,S);
    mon2 := sub((monPair_1)_0,S);
    local H;
    -- construct homotopy according to which polynomials
    H = if (monPair_0)_1 ==1 and (monPair_1)_1 ==1 then \\
             pairHomotopy1(f1, f2, mon1, mon2, t, sclr, N)
         else if (monPair_0)_1 == 1 and (monPair_1)_1==2 then \\
             pairHomotopy12(f1, f2, mon1, mon2, t, sclr, N)
         else pairHomotopy2(f1, f2, mon1, mon2, t, sclr, N);
    -- solve the system at each connection point in the homotopy
    homSols = BertSolsN(H, sols, s,t);
    -- find the degree that fits the changes in these solutions
    deg = degreeTest(homSols, tol, coord);
    -- print corresponding signifying statements
    if deg==0 then (<< monPair << " observed effect is constant " << endl)
```

```
        else if deg==1 then (<< monPair << " observed effect is linear"  << endl)
        else (<< monPair << " observed effect is non-linear" << endl);
        deg
    );
    degreeList
)
```

# Bibliography


[1] D. J. Bates, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler. Bertini: Software for numerical algebraic geometry. Available at `bertini.nd.edu` with permanent doi: dx.doi.org/10.7274/R0H41PB5.

[2] D. J. Bates, J. D. Hauenstein, A. J. Sommese, and C. W. Wampler. *Numerically Solving Polynomial Systems with Bertini*. SIAM, 2013.

[3] D. N. Bernstein. The number of roots of a system of equations. *Functional Analysis Applications*, 9:183–185, 1975.

[4] P. Breiding and S. Timme. HomotopyContinuation.jl: A Package for Homotopy Continuation in Julia. In *International Congress on Mathematical Software*, pages 458–465. Springer, 2018.

[5] T. Brysiewicz and M. A. Burr. Sparse trace tests. Technical Report arXiv:2201.04268 [math.AG], arXiv, 2022.

[6] D. A. Cox, J. Little, and D. O'Shea. *Using Algebraic Geometry*. Springer, 2 edition, 2005.

[7] Desmos. Desmos graphing calculator. Available at `desmos.com/calculator`.

[8] I. Gelfand, M. Kapranov, and A. Zelevinsky. *Discriminants, Resultants, and Multidimensional Determinants*. Birkhäuser, 1994.

[9] D. R. Grayson and M. E. Stillman. Macaulay2, a software system for research in algebraic geometry. Available at `http://www.math.uiuc.edu/Macaulay2/`.

[10] A. G. Khovanskii. Newton polyhedra and the genus of full intersections. *Funksional. Anal. i Prilozhen*, 12:51–61, 1978.

[11] A. G. Kushnirenko. The Newton polyhedron and the number of solutions of a system of $k$ equations in $k$ unknowns. *Uspekhi Mat. Nauk*, 30:266–267, 1975.

[12] T. L. Lee, T. Y. Li, and C. H. Tsai. HOM4PS-2.0: a software package for solving polynomial systems by the polyhedral homotopy continuation method. *Computing*, 83(109), 2008.

[13] A. Leykin and R. Krone. Numerical algebraic geometry. *The Journal of Software for Algebra and Geometry: Macaulay2*, 3, May 2011.

[14] A. Leykin, J. I. Rodriguez, and F. Sottile. Trace test. *Arnold Mathematical Journal*, 4:113–125, 2018.

[15] A. J. Sommese, J. Verschelde, and C. W. Wampler. Symmetric functions applied to decomposing solution sets of polynomial systems. *SIAM Journal of Numerical Analysis*, 40:2026–2046, 2002.

[16] G Staglianò. A package for computations with sparse resultants. *Journal of Software for Algebra and Geometry*, 11(1):61–69, Dec 2021.

[17] J. Verschelde. Algorithm 795: PHCpack: a general-purpose solver for polynomial systems by homotopy continuation. *ACM Transactions on Mathematical Software*, 25:251–276, June 1999.