12-17-2019 10:30 AM

Algorithms for Mappings and Symmetries of Differential Equations

Zahra Mohammadi
The University of Western Ontario

Supervisor
Gregory Reid
The University of Western Ontario

Graduate Program in Applied Mathematics
A thesis submitted in partial fulfillment of the requirements for the degree in Doctor of Philosophy

Abstract

Differential Equations are used to mathematically express the laws of physics and models in biology, finance, and many other fields. Examining the solutions of related differential equation systems helps to gain insights into the phenomena described by the differential equations. However, finding exact solutions of differential equations can be extremely difficult and is often impossible. A common approach to addressing this problem is to analyze solutions of differential equations by using their symmetries.

In this thesis, we develop algorithms based on analyzing infinitesimal symmetry features of differential equations to determine the existence of invertible mappings of less tractable systems of differential equations (e.g., nonlinear) into more tractable systems of differential equations (e.g., linear). We also characterize features of the map if it exists. An algorithm is provided to determine if there exists a mapping of a non-constant coefficient linear differential equation to one with constant coefficients. These algorithms are implemented in the computer algebra language `Maple`, in the form of the `MapDETools` package. Our methods work directly at the level of systems of equations for infinitesimal symmetries. The key idea is to apply a finite number of differentiations and eliminations to the infinitesimal symmetry systems to yield them in the involutive form, where the properties of Lie symmetry algebra can be explored readily without solving the systems.

We also generalize such differential-elimination algorithms to a more frequently applicable case involving approximate real coefficients. This contribution builds on a proposal by Reid et al. of applying Numerical Algebraic Geometry tools to find a general method for characterizing solution components of a system of differential equations containing approximate coefficients in the framework of the Jet geometry. Our numeric-symbolic algorithm exploits the fundamental features of the Jet geometry of differential equations such as differential Hilbert functions. Our novel approach establishes that the components of a differential equation can be represented by certain points called critical points.

Keywords: Symmetry, Lie algebra, equivalence mappings, differential elimination. Linearization, differential algebra, differential elimination, involutivity, Jet Geometry, Cartan Kuranishi algorithm, Numerical Jet Geometry, Critical points

Summary for Lay Audience

Differential Equations are used to mathematically express the governing laws of physics and models in biology, finance, and other fields. However, such equations can be difficult to analyze or solve analytically or numerically. For example, they may be nonlinear, or even if they are linear, may have non-constant coefficients. In this thesis, we develop algorithms to determine whether an invertible mapping of a nonlinear system of differential equations to a linear system exists. Once existence is established, it can determine features of the map and if possible explicitly determine the mapping by integration. We also provide an algorithm to determine if a mapping of a non-constant coefficient linear differential equation onto a simple constant coefficient differential equation exists. These algorithms are implemented in the symbolic computation language `Maple`, as a part of the `MapDETools` package. So, our methods are available to a wide audience through user-friendly interfaces.

The above methods depend on analyzing the symmetry properties of the input (e.g., nonlinear) systems for features that characterize the (e.g. linear) target. The methods also employ an exact differential-elimination algorithm that applies a finite number of differentiations and eliminations to the system of differential equations for the symmetries and reduces them to the involutive form, where their properties are readily determined. We also generalize such differential-elimination algorithms to the more realistic case of input systems with approximate real coefficients. This algorithm exploits fundamental features of the Jet geometry of differential equations such as differential Hilbert functions.

Co-Authorship Statement

Chapters 2 - 4 of this thesis consist of the following papers:

- Chapter 2: Zahra Mohammadi, Gregory J. Reid and Tracy Shih-lung Huang. Introduction of the MapDE Algorithm for Determination of Mappings Relating Differential Equations. Proceedings of International Symposium on Symbolic and Algebraic Computation (ISSAC'19), ACM, Pages 331-338, 2019.

- Chapter 3: Zahra Mohammadi, Gregory J. Reid and Tracy Shih-lung Huang. Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDS to linear PDS. Submitted to Mathematics in Computer Science (Revision requested 15-MAY-2019), Technical Report arXiv:1903.03727v1 [math.AP]. ArXiV. 2019.

- Chapter 4: Zahra Mohammadi, Gregory J. Reid. A Completion algorithm for real differential polynomial systems to Involutive Form.

As the leading author of the above articles in chapters 2 and 3, I did the experiments under the supervision of Dr. G. Reid. The implementations were done by Dr. Reid and the author. The primary draft for both articles was prepared by the author and Dr. Reid and followed by performing the subsequent revisions. Dr. Tracy Huang primarily helped with the use and understanding of the `LieAlgebrasOfVectorFields` package.

I prepared the original draft of chapter 4. All implementation and numeric computation with `Bertini`, a numerical solver for polynomial systems, has been done by the author. The analytical computations have been done by the author under the supervision of Dr. Greg Reid.

To my parents for their endless love,
encouragements and support.

Acknowledgments

I would like to thank professor Gregory Reid, my Ph.D. supervisor, who made this thesis possible. I wish to express my deepest gratitude and appreciation to him for his kind guidance, unlimited support, and encouragement. First, I am thankful to Greg for sharing his knowledge and thought. He introduced me to the world of differential equations and different approaches to studying them. His insight about Symmetry motivated me to implement the mapping differential equation algorithms. Also, my work on approximate differential equations follows his idea about the relation between the formal theory of partial differential equations and Numerical Algebraic Geometry. Second, I thank him for teaching me to be a researcher, supporting my attendance at various conferences and workshops, and providing me with the internship opportunity to work at Maplesoft.

I am indebted to him for supporting me, academically and emotionally, when I lost my dad at the beginning of my Ph.D. journey. He has been my best teacher and best friend during these four years.

Additionally, I wish to thank my committee members Professor Thomas Wolf, Professor Marc Moreno Maza, Professor Martin Pinsonnault, and Professor Olga Trichtchenko for their interest in my work.

I would like to thank my coauthor, Tracy Haung. Our collaboration gave me a good start to learning professional programming.

I would like to thank each member of the MapleSoft company, especially Allan Wittkopf and Erik Postma. I am indebted to them for every single moment they have patiently spent teaching me the mpldoc language, a documentation language in Maplesoft, and helping me integrate the `LieAlgebraVectorFields` package into `Maple`.

I appreciate all the guidance I have received from Jonathan Hauenstein about the `Bertini` software and Allan Wittkopf about the interface feature of `Bertini` with `Maple`.

My sincere thanks go to Prof. Rob Corless, Prof. David Jeffrey, and Prof. Marc Moreno Maza for creating an inspiring environment in the ORCCA lab. It was an honor for me to be their student. I learned professional academic behavior as well as the correct way to research from them.

I wish to express my appreciation to Mrs. Audrey Kager, academic program coordinator in the Applied Mathematics department, for her continuous work and valuable assistance during my study.

On a personal note, I would like to show my gratitude to my mother, Masomeh, and my sisters, Maryam, Zohreh, and Sahar, for their support, sacrifices, and great love. I want to pay special attention to my late father, Esfadiar, who always encouraged and supported his family to accomplish their goals. We lost him in November 2015 and words cannot describe how much we miss him. I also thank my friends Aida Ahmadi and Sepideh Moghimi Nezhad for supporting me when I needed them.

Contents

Abstract i

Summary for Lay Audience ii

Co-Authorship Statement iii

Dedication iv

Acknowledgments v

List of Figures ix

List of Appendices x

1 Introduction and Mathematical Background 1
- 1.1 Symmetry . . . 2
- 1.2 Construction of Mappings . . . 5
- 1.3 Symbolic differential elimination completion algorithms . . . 12
- 1.4 Numerical algebraic geometry . . . 15
- 1.5 Thesis Outline . . . 17
  - 1.5.1 Contents of Chapter 2: "Introduction of the MapDE Algorithm for Mappings Relating Differential Equations" . . . 17
  - 1.5.2 Contents of Chapter 3: "Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDS to linear PDS" . . . 18
  - 1.5.3 Contents of Chapter 4: "A Completion algorithm for real differential polynomial systems to Involutive Form" . . . 18

Bibliography 18

2 Introduction of the MapDE Algorithm for Mappings Relating Differential Equations 22
- 2.1 Introduction . . . 22
- 2.2 Preliminaries & Mapping Equations . . . 25
  - 2.2.1 Mapping Equations . . . 26
  - 2.2.2 Algorithms EquivDetSys and DimEquivTest . . . 28
- 2.3 MapDE Algorithm . . . 28
  - 2.3.1 The MapDE Agorithm for specific R and $\hat{R}$ . . . 28

2.3.2 MapDE for mapping Linear Homogeneous DE to Constant Coefficient Linear DE . . . 30
2.4 Examples . . . 31
2.4.1 Equivalence . . . 31
2.4.2 Mapping to constant coefficient DE . . . 33
2.5 Discussion . . . 35

Bibliography **36**

3 Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE **41**
3.1 Introduction . . . 41
3.2 Differential-elimination algorithms, initial data and Hilbert functions . . . 43
3.3 Symmetries & Mapping Equations . . . 46
3.3.1 Symmetries . . . 46
3.3.2 Bluman-Kumei Mapping Equations . . . 49
3.4 Algorithms and Preliminaries for the MapDE Algorithm . . . 51
3.4.1 Symmetries of the linear target and the derived algebra . . . 51
3.4.2 Algorithm PreEquivTest for excluding obvious nonlinearizable cases . . . 52
3.4.3 Algorithm ExtractTarget for extracting the linear target system . . . 53
3.4.4 Heuristic integration for the Mapping functions in MapDE using PDSolve . . . 54
3.5 The MapDE Algorithm . . . 54
3.5.1 Pseudo-code for the MapDE algorithm . . . 54
3.5.2 Notes on the MapDE Algorithm with Target = LinearDE . . . 55
3.5.3 Proof of correctness of the MapDE Algorithm . . . 56
3.6 Examples . . . 58
3.7 Discussion . . . 63

Bibliography **66**
Index . . . 70

4 A completion algorithm for approximate real differential polynomial systems to Involutive Form **72**
4.1 Introduction . . . 72
4.2 Mathematical Background . . . 74
4.2.1 Symbolic-Numeric Method for Linear Homogeneous DPS . . . 76
4.2.2 Critical Point Approach for Nonlinear PDS . . . 77
4.3 Implementation . . . 81
4.3.1 Symbolic-Numeric completion algorithm for approximate real DPS . . . 81
4.3.2 HybridInvolutiveFormLHPDE . . . 83
4.4 Example . . . 84
4.4.1 Illustrative example . . . 84
4.4.2 Application to approximate symmetry . . . 86
4.5 Conclusion . . . 87

Bibliography 88

5 Conclusion and Future work 93

Bibliography 95

A Demo for the MapDETools Package 98

B Copyright Release 128

Curriculum Vitae 130

List of Figures

1.1 The vector field of the rotation group 4

1.2 The Figure represents the relationship between various solution sets: solutions of the symmetry defining system S generating Lie algebra $\mathscr{L}$ for R and $\hat{S}$ its image under the mapping $\Psi = (\psi, \phi)$. S^* is the symmetry defining system for the $\hat{R}$ in (x, u) coordinates and $\hat{S^*}$ is its image under the mapping Ψ (note $S^* \setminus S$ is usually non-empty). S' is the symmetry defining system for a Lie subalgebra of $\mathscr{L}'$ with the property that $S' \cap S^* = S \cap S^*$. Also $\hat{S'}$ is its image under the mapping Ψ. 12

2.1 Regions of approximate symmetry in the $x-y$ plane for $\nabla^2 u = f(x, y, z)$. Purple Region: $\dim \mathscr{L} = 1$, $\mathscr{L} \approx$so(2); Yellow Region $\dim \mathscr{L} = 3$,$\mathscr{L} \approx$so(3); Red Region $\dim \mathscr{L} = 11$. Between the red, yellow and purple regions are transition bands in which the approximate Lie algebra was not stably computed. 37

3.1 In the figure, the empty circles $\circ$ correspond to parametric derivatives, and solid circles $\bullet$ corresponding to derivatives of leaders. The Hilbert Series is obtained by diagonal counting of the empty circles. 45

3.2 The graph represents the CPU times for $\left(\frac{d}{dx}\right)^d (u(x)^2) + u(x)^2 = 0$. Timings from $t = 0$ (start of `MapDE`) to the time to LGMLin linearization Existence confirmation (Red), time to Hilbert Existence confirmation (Yellow), time from $t = 0$ to existence and construction of the linearizing transformations (Green). . 61

4.1 Curtain Pendulum([37]) 74

4.2 $v^2 - (1 - u^2)^3 = 0$ (left), $v^2 - (1 - u^2)^3 + z_1 = 0$ (right) 79

4.3 System F1 85

4.4 System F2 85

4.5 Whitney umbrella (This picture is taken from Wikipedia) 86

List of Appendices

Appendix A Demo for the MapDETools Package 98
Appendix B Copyright Release 128

Chapter 1

Introduction and Mathematical Background

Mathematical models from applications in science and engineering usually involve equations between rates of change of unknown quantities and lead to differential equations. For this reason determining information about solutions and solving differential equations approximately or exactly is of fundamental importance. The fact that most differential equations are impossible to solve exactly highlights the importance of determining if they can be mapped to more desirable forms, or expressed in more suitable coordinates. The first major breakthrough for addressing such problems was made in the 19th century by Sophus Lie with his theory of continuous symmetry groups of differential equations. Such Lie symmetry groups map equations to themselves, and have many applications such as the determination of invariant solutions, reduction of order and the determination of mappings between equations. Symmetry analysis of differential equations is still under intense development [3, 14].

This thesis is aimed at developing theory and algorithms to determine the existence of invertible mappings of less tractable systems of differential equations to more tractable systems and characterize the map if it exists. The approach is based on analyzing symmetry features of differential equations. Let us first give an overview and examples to help the non-expert reader.

Example 1.0.1 *Consider the famous Black Schole's equation which is fundamental in financial applications [20]*

$$\frac{\partial}{\partial t}v + \frac{s^2}{2}\left(\frac{\partial}{\partial s}\right)^2 v + s\frac{\partial}{\partial s}v - v = 0 \tag{1.0.1}$$

when $v = v(s,t)$ is the price function of stock price s and time t. It has the obvious symmetry of translation in $t : t^* = t + \epsilon$ and scaling in $s : s^* = bs$. The reader can easily verify that these transformations leave the equation (1.0.1) invariant. Clearly, the transformation $\hat{s} = \int \frac{ds}{s} = \log(s) + c_1$, $\hat{t} = t + c_2$ maps the Black Schole's equation to

$$\frac{\partial}{\partial \hat{t}}v + \frac{1}{2}\left(\frac{\partial}{\partial \hat{s}}\right)^2 v - v = 0 \tag{1.0.2}$$

which is a constant coefficient linear equation for which a wide array of solutions methods exist.

In this chapter, we introduce mathematical background and definitions that are important to the thesis. §1.1 gives some fundamental definitions for Lie transformation groups and their associated Lie symmetry algebras. In order to understand these definitions, we present their geometric features as well as their algebraic definitions. §1.2 presents symmetry properties of a differential equation to give insights into mappings between differential equations. Some symmetry properties are demonstrated through the example 1.0.1. Section §1.3 gives a brief overview of differential-elimination algorithms which is used in the construction steps of our mapping algorithm. Some basic background about the numerical algebraic geometry is provided in section §1.4. We end this chapter with brief overview of the contents of each chapter presented in the thesis.

1.1 Symmetry

Roughly speaking, a symmetry of a geometric object is a transformation that preserves the structure of the object.

First recall the definition of group:

Definition 1.1.1 *[21] A group G is a set with binary operation $g \circ h$ for $g, h \in G$ satisfying*

[i] $g \circ h \in G$ for all $g, h \in G$

[ii] Associativity: $g \circ (h \circ k) = (g \circ h) \circ k$ for all $g, h, k \in G$

[iii] Identity: the group have an unique identity element e; $g \circ e = g = e \circ g$

[iv] Inverse: each elements of the group has an inverse that; $g \circ g^{-1} = g^{-1} \circ g = e$.

Example 1.1.1 (Symmetry group) *The set of symmetries of a geometric object forms a group for which the group operation $g \circ h$ is composition (means first do h then do g). It satisfies;*

[i] The composition of two symmetries is a symmetry.

[ii] The identity (do nothing) is always a symmetry.

[iii] The inverse of a symmetry (undo it) is a symmetry.

Now we derive the differential geometric definition of symmetry.

Definition 1.1.2 (Lie group) *([21], §1) A Lie group G is a smooth manifold such that the group multiplication $G \times G \to G$, $(g, h) \to g.h$, and group inverse $G \to G$, $g \to g^{-1}$ are smooth maps.*

If the manifold has dimension r, the group is called a r-parameter Lie group.

Example 1.1.2 *Consider $G = \mathbb{R}^r$ the set of vectors with r real entries with the operation vector addition. Let $g, h \in \mathbb{R}^r$ then $g \circ h = g + h$. Clearly, the identity element is the zero vector, and the inverse of every vector g is $-g$. So, an r- parameter Lie group is the abelian (means commutative) Lie group $\mathbb{R}^r$.*

Naturally, in applications, groups are acting as a family of transformations on a space. In terms of Lie groups, we have the following definition;

Definition 1.1.3 (Lie transformation) *([21], §1) A transformation group acting on a smooth manifold M is determined by a Lie group G and smooth map $\Phi : G \times M \to M$, denoted by $\Phi(g, x) = g(x)$, which satisfies*

$$e(x) = x, \quad g(h(x)) = (g \circ h)(x), \quad \text{for all } x \in M,\ g, h \in G \tag{1.1.1}$$

Condition (1.1.1) implies that the inverse group element g^{-1} determines the inverse to the transformation defined by the group element g, so that each group element g induces a diffeomorphism from M to itself.

Example 1.1.3 (Euclidean group $E(2, \mathbb{R})$) *Transformations in this group map $(x, y) \in \mathbb{R}^2$ to $(\tilde{x}, \tilde{y}) \in \mathbb{R}^2$ by a rotation θ and translation (a, b):*

$$\begin{pmatrix} \tilde{x} \\ \tilde{y} \end{pmatrix} = \begin{pmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{pmatrix} \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix} \tag{1.1.2}$$

This is a 3-dimensional Lie group since $g \in E(2, \mathbb{R})$ depends on group parameters θ, a, b.

A r-dimensional Lie group on n-dimension smooth manifold M has group transformations of the form

$$\tilde{x}^i = \phi_i(x^1, \ldots, x^n, a^1, \cdots, a^r) \tag{1.1.3}$$

with smooth functions $\phi_i, i = 1, \ldots, n$ and group parameters $a^1, \ldots, a^r$. Lie showed that the case of finite r-parameter Lie groups can be understood and analyzed in terms of their one-parameter subgroups

$$\tilde{x} = \phi(x, \epsilon) \tag{1.1.4}$$

where $\tilde{x} = (\tilde{x}^1, \tilde{x}^2, \ldots, \tilde{x}^n)$ and $x = (x^1, x^2, \ldots, x^n)$. If $\tilde{x} = \phi(x, \epsilon)$ and $\tilde{\tilde{x}} = \phi(\tilde{x}, \delta)$ with the group parameters $\epsilon,\ \delta \in \mathbb{R}$ then we have

$$\tilde{\tilde{x}} = \phi(x, \psi(\epsilon, \delta))$$

for some ψ. If the group composition is given by $\psi(\epsilon, \delta) = \epsilon + \delta$ then the identity of the transformation group is $\epsilon = 0$ and the inverse is ϵ^{-1}.

Clearly, these group transformations are nonlinear in their parameters a in (1.1.3), but they are linearizable at their identity. Working with the linear form of Lie transformation group, called infinitesimal transformations, is much easier than working with the nonlinear Lie group.

Definition 1.1.4 (Infinitesimal transformations) *[21] Each $\tilde{x}(\epsilon)$ may be represented as a Taylor series in ϵ (in a neighbourhood of $\epsilon = 0$)*

$$\tilde{x} = x + \epsilon \left.\frac{\partial \phi}{\partial \epsilon}(x, \epsilon)\right|_{\epsilon=0} + \frac{\epsilon^2}{2} \left.\frac{\partial^2 \phi}{\partial \epsilon^2}(x, \epsilon)\right|_{\epsilon=0} + O(\epsilon^2)$$

The infinitesimal form of the one-parameter Lie transformation group (1.1.4) is;

$$\tilde{x} = x + \epsilon\, \xi(x) + O(\epsilon^2) \tag{1.1.5}$$

where $\xi(x) = \left.\frac{\partial \phi}{\partial \epsilon}(x, \epsilon)\right|_{\epsilon=0}$. The components $\xi(x)$ are called the infinitesimals of (1.1.4).

Example 1.1.4 *The infinitesimal transformation of the 1-dimensional rotation group, known as $O(2)$, are obtained by expanding (1.1.2) in the neighborhood of $\theta = 0$;*

$$\begin{aligned} \tilde{x} &= x + \theta \left.\frac{\partial(x\cos\theta - y\sin\theta)}{\partial\theta}\right|_{\theta=0} = x - \theta\, y \\ \tilde{y} &= y + \theta \left.\frac{\partial(x\sin\theta + y\cos\theta)}{\partial\theta}\right|_{\theta=0} = y + \theta\, x \end{aligned} \tag{1.1.6}$$

The infinitesimals of the orthogonal group $O(2)$ are $\xi(x,y) = -y$ and $\eta(x,y) = x$.

Definition 1.1.5 (Vector Field) *[21] The vector field*

$$X \equiv \xi^1(x)\,\frac{\partial}{\partial x^1} + \xi^2(x)\,\frac{\partial}{\partial x^2} + \cdots + \xi^n(x)\,\frac{\partial}{\partial x^n} \tag{1.1.7}$$

is called the infinitesimal generator of the one-parameter Lie transformation group (1.1.4).

Example 1.1.5 *The vector field of the 1-dimensional rotation group $O(2)$ given in (1.1.2), is*

$$X \equiv -y\,\frac{\partial}{\partial x} + x\,\frac{\partial}{\partial y}$$

The components of the vector field $(-y, x)$ are tangent to circles $x^2 + y^2 = r^2$. See Fig. 1.1.

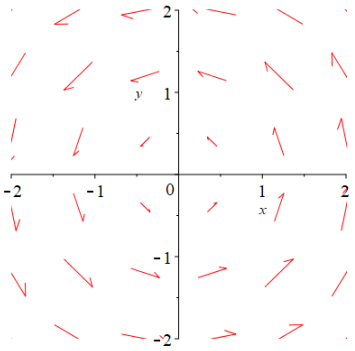


Figure 1.1: The vector field of the rotation group

Such examples motivate the following definition.

Definition 1.1.6 *[22] Suppose R is a system of differential equations involving n independent variables $x = (x^1, \ldots, x^n)$ and m dependent variables $u = (u^1, \ldots, u^m)$. A symmetry group of the system R is a Lie group of transformations G acting on the space $X \times U$ for the system such that G transforms every solution of R to another solution of R.*

In particular, a point transformation

$$\hat{x} = \psi(x,u), \qquad \hat{u} = \phi(x,u), \tag{1.1.8}$$

is a symmetry of the system R, if it maps each solution of R to another solution. Then, a point symmetry group of R is a set of symmetries that forms a transformation group acting on $X \times U$. In such case, the symmetry group has associated vector field:

$$X \equiv \sum_{i=1}^{n} \xi^i \frac{\partial}{\partial x^i} + \sum_{j=1}^{m} \eta^j \frac{\partial}{\partial u^j} \tag{1.1.9}$$

where ξ^i and η^j depend on x and u.

Example 1.1.6 *In the Black Schole's example, the vector fields $\frac{\partial}{\partial t}$ and $s\frac{\partial}{\partial s}$ generate a 2-dimensional Lie group with infinitesimal generator of form*

$$X \equiv \xi^1(s,t,v)\frac{\partial}{\partial s} + \xi^2(s,t,v)\frac{\partial}{\partial t} + \eta^1(s,t,v)\frac{\partial}{\partial v}$$

where $\xi^1 = as,\ \xi^2 = b,\ \eta^1 = 0$.

At this point the reader should be more comfortable with Lie groups of transformation and their associate infinitesimal generators. The following section is provided to get a better understanding of such concepts to demonstrate a mapping algorithm for related differential equations based on point symmetry properties to characterize the map based on exploring the algebraic and geometric properties of Lie Algebras.

1.2 Construction of Mappings

Now, we define an important object that will help us implement the mapping algorithm.

Definition 1.2.1 (Determining equations for infinitesimal symmetries) *[28] The coefficients $\xi^1, \ldots, \xi^n$ of the general vector field of a Lie transformation group, (1.1.7), satisfy a system of linear homogeneous partial differential equations. They are called determining equations or defining equations of the group.*

Suppose a one-parameter Lie group of point transformations

$$\tilde{x}^i = \psi^i(x,u,\epsilon), \quad \tilde{u}^j = \phi^j(x,u,\epsilon) \tag{1.2.1}$$

has corresponding infinitesimal generator

$$X = \sum_{i=1}^{n} \xi^i(x,u)\frac{\partial}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x,u)\frac{\partial}{\partial u^j}. \tag{1.2.2}$$

We determine the coefficients ξ^i and η^j (i.e. the infinitesimals) such that (1.2.2) is a symmetry vector field of R.

Note in where follows we sometimes use the Einstein summation convention, what repeated indices imply summation. For example (1.2.2) can be abbreviated as $X = \xi^i(x,u)\frac{\partial}{\partial x^i} +$

$\eta^j(x,u)\frac{\partial}{\partial u^j}$. See Bluman, Anco and Cheviakhov [3] for these formula using the summation convention.

The infinitesimal transformations need to be extended or prolonged to derivatives. This yields the k-th extended infinitesimal generator (k-th prolongation) of (1.2.2) given by:

$$X^{(k)} = \xi^i(x,u)\frac{\partial}{\partial x^i} + \eta^j(x,u)\frac{\partial}{\partial u^j} + \eta_i^{(1)j}(x,u,\partial u)\frac{\partial}{\partial u_i^j} + \cdots + \eta_{i_1\cdots i_k}^{(k)j}(x,u,\partial u,\cdots,\partial^k u)\frac{\partial}{\partial u_{i_1\cdots i_k}^j} \tag{1.2.3}$$

where the extended infinitesimals are defined by

$$\eta_i^{(1)j} = D_i\eta^j - (D_i\xi^j)u_l^j$$
$$\eta_{i_1\cdots i_k}^{(k)j} = D_{i_k}\eta_{i_1\cdots i_{k-1}}^{(k-1)j} - (D_{i_k}\xi^l)u_{i_1\cdots i_{k-1}l}^j$$

and

$$D_i = \frac{\partial}{\partial x^i} + u_i^j\frac{\partial}{\partial u^j} + u_{ii_1}^j\frac{\partial}{\partial u_{i_1}^j} + u_{ii_1i_2}^j\frac{\partial}{\partial u_{i_1i_2}^j} + \cdots$$

for $i = 1,\ldots,n$, $j = 1,\ldots,m$ and $l = 1,\ldots,k$. In the above equations we have used the Einstein summation convention of summing over repeated indices, to avoid excessive $\sum$ notation.

Definition 1.2.2 *([3]) A one-parameter Lie group of point transformations (1.2.1) leaves the system R invariant if and only if its k-th extension (1.2.3) leaves invariant the solution manifold of R in $(x,u,\partial u,\ldots,\partial^k u)$-space, i.e., it maps any family of solutions $u = u(x)$ of the differential system R into another family of solutions $\tilde{u} = \tilde{u}(\tilde{x})$ of R. In this case, the one-parameter Lie group of point transformations (1.2.1) is called a point symmetry of the system R.*

Theorem 1.2.3 *The transformation (1.2.1) is a point symmetry of the R if and only if*

$$X^{(k)}R(x,u,\partial u,\cdots,\partial^k u)\Big|_{R=0} = 0$$

In summary, the result of $X^{(k)}R\,\Big|_{R=0} = 0$ is a list of linear homogeneous partial differential equations for ξ^i and η^j, called the determining or defining equations for the vector fields (1.2.2) of the system of differential equations [4, 14]. The determining system can be computed automatically by many existing computer algebra implementations [8, 35, 11, 25]. Sometimes ξ^i and η^j can be found by heuristic computer methods, but no general algorithm exists for this task.

We illustrate these steps in the following simple example:

Example 1.2.1 *([14] §3) Consider the second order differential equation,*

$$y_{xx} = 0 \tag{1.2.4}$$

with associated symmetry vector field

$$X \equiv \xi(x,y)\,\frac{\partial}{\partial x} + \eta(x,y)\,\frac{\partial}{\partial y} \tag{1.2.5}$$

The determining equations for Lie point symmetries of (1.2.4) result from

$$X^{(2)}\big|_{y_{xx}=0} = 0. \tag{1.2.6}$$

We construct the determining equations as follows. In step 1, we prolong the vector field (1.2.5) to order 2

$$X^{(2)} = \xi \frac{\partial}{\partial x} + \eta \frac{\partial}{\partial y} + \eta^{(1)} \frac{\partial}{\partial y_x} + \eta^{(2)} \frac{\partial}{\partial y_{x,x}}$$

So, we need to calculate η^k where $k = 1, 2$;

$$\begin{aligned} \eta^{(1)} &= \eta_x + (\eta_y - \xi_x)y_x - \xi_y y_x^2 \\ \eta^{(2)} &= \eta_{xx} + (2\eta_{xy} - \xi_{xx})y_x + (\eta_{yy} - 2\xi_{xy})y_x^2 - \xi_{yy}y_x^3 + (\eta_y - 2\xi_x - 3\xi_y y_x)y_{xx} \end{aligned} \tag{1.2.7}$$

Step 2: Apply the vector field $X^{(2)}$ to the differential equation (1.2.4),

$$X^{(2)} y_{xx} = \eta^{(2)}$$

We substitute equations (1.2.7) into

$$\eta^{(2)} = \xi\omega_x + \eta\omega_y + \eta^{(1)}\omega_{y_x}$$

where $y_{xx} = \omega(x, y, y_x)$. So, we have

$$\begin{aligned} &\eta_{xx} + (2\eta_{xy} - \xi_{xx})y_x + (\eta_{yy} - 2\xi_{xy})y_x^2 - \xi_{yy}y_x^3 + (\eta_y - 2\xi_x - 3\xi_y y_x)\omega \\ &= \xi\omega_x + \eta\omega_y + (\eta_x + (\eta_y - \xi_x)y_x - \xi_y y_x^2)\omega_{y_x} \end{aligned} \tag{1.2.8}$$

Step 3: Restrict $X^{(2)}y_{xx}$ to the subset where $y_{xx} = 0$ in the prolonged space (x, y, y_x, y_{xx}). So, (1.2.4) is

$$\eta^{(2)} = 0 \qquad \text{where} \quad y_{xx} = 0$$

that is

$$\eta_{xx} + (2\eta_{xy} - \xi_{xx})y_x + (\eta_{yy} - 2\xi_{xy})y_x^2 - \xi_{yy}y_x^3 = 0$$

Step 4: Now, split the above equation by powers of y_x. Note $\frac{\partial \eta}{\partial y_x} = 0 = \frac{\partial \xi}{\partial y_x}$ so splitting in powers y_x can be seen by taken derivatives with respect to y_x:

$$\eta_{xx} = 0, \quad 2\eta_{xy} - \xi_{xx} = 0, \quad \eta_{yy} - 2\xi_{xy} = 0, \quad \xi_{yy} = 0 \tag{1.2.9}$$

So, the symmetry condition for (1.2.6) is given by the list of determining equations.
There is no general algorithm to solve such determining systems. However, heuristic computers methods can sometimes solve such systems. The above system can be solved by hand or by one of the available computer methods such as [35] yielding:

$$\begin{aligned} \xi(x, y) &= c_1 + c_3 x + c_5 y + c_7 x^2 + c_8 xy \\ \eta(x, y) &= c_2 + c_4 y + c_6 x + c_7 xy + c_8 y^2 \end{aligned}$$

where $c_1, \ldots, c_8$ are constants. Finally, the most general symmetry vector field of the ordinary differential equation $y_{xx} = 0$ is:

$$X \equiv (c_1 + c_3 x + c_5 y + c_7 x^2 + c_8 xy)\frac{\partial}{\partial x} + (c_2 + c_4 y + c_6 x + c_7 xy + c_8 y^2)\frac{\partial}{\partial y} = \sum_{i=1}^{8} c_i X_i$$

where a basis of the infinitesimal symmetry generators is given by

$$\begin{aligned} X_1 &= \frac{\partial}{\partial x}, \quad X_3 = x\frac{\partial}{\partial x}, \quad X_5 = y\frac{\partial}{\partial x}, \quad X_7 = x^2\frac{\partial}{\partial x} + xy\frac{\partial}{\partial y}, \\ X_2 &= \frac{\partial}{\partial y}, \quad X_4 = y\frac{\partial}{\partial y}, \quad X_6 = x\frac{\partial}{\partial y}, \quad X_8 = xy\frac{\partial}{\partial x} + y^2\frac{\partial}{\partial y} \end{aligned}$$

The corresponding symmetry transformations for these vector fields satisfy an associated ordinary differential equation initial value problem which can sometimes be solved. For example, corresponding to the basis member $\frac{\partial}{\partial x}$, the associated initial value problem is $\frac{d\tilde{x}}{d\epsilon} = 1,\ \tilde{x}(0) = x$ and has solution $\tilde{x} = x + \epsilon$ corresponding to translation in x.

As you can see in the above example, the steps of point symmetry calculation are systematic and tedious and are facilitated by computer algebra packages. [35, 3, 11, 9].

Example 1.2.2 *For the Black-Schole's Equation (1.0.1), the infinitesimal symmetry operator has form*

$$X \equiv \sigma(s,t,v)\frac{\partial}{\partial s} + \tau(s,t,v)\frac{\partial}{\partial t} + \eta(s,t,v)\frac{\partial}{\partial v}.$$

Here comparing with (1.2.14) yields $(\xi^1, \xi^2) = (\sigma, \tau)$ and $\eta^1 = \eta$. The automatically generated un-simplified system of defining equations for infinitesimal symmetries is:

$$\begin{aligned} &\tau_s = 0, \quad \tau_v = 0, \quad \tau_{v,v} = 0, \quad \sigma_{v,v} = 0, \\ &\tau_{s,v}s^2 - 2\,\sigma_v = 0, \quad \eta_{v,v}s - 2\,\sigma_{s,v}s - 4\,\sigma_v = 0, \\ &-2\,\eta_{s,v}s^2 + \sigma_{s,s}s^2 + 2\,\sigma_s s + 6\,\sigma_v v - 2\,\sigma + 2\,\sigma_t = 0, \\ &\tau_{s,s}s^3 - 2\,\tau_s s^2 + 2\,\tau_v sv + 2\,\tau_t s - 4\,\sigma_s s + 4\,\sigma = 0, \\ &-\eta_{s,s}s^3 + 2\,\eta_s s^2 - 2\,\eta_v sv + 4\,\sigma_s sv + 2\,\eta\, s - 2\,\eta_t s - 4\,v\sigma = 0 \end{aligned} \tag{1.2.10}$$

In general, the complexity of expressions arising in the process of setting up and attempting to solve determining equations for symmetries of a given differential system will increase rapidly as the order or numbers of independent and dependent variables in the differential system increases. Differential elimination algorithms are a common tool to address this issue. I explain these algorithms in more detail in the next section §1.3. Generally speaking, the process of finding the determining equations of a differential system has three steps: finding the determining equations, reducing them to get their complete form, and finally solving them. Many computer algebra packages can compute the determining equations. For example see Bluman, Anco and Cheviakhov [3] for references to such packages which are available in all the major computer algebra programs. The differential elimination completion method such as `rifsimp` in `Maple` will improve the computation and increase the success rate of the solving

step. At the end, any PDE solver like `pdsolve` in `Maple` can be used on the completion form of the determining equation provided in the previous step. See section 1.3.

There are several important invariant objects associated with a transformation group, such as, invariant function,vector fields, differential operators and etc. The invariant vector field is the most important one since they serve as the infinitesimal generators of the group action.

Definition 1.2.4 (Lie Algebra [21] §2) *A Lie Algebra $\mathscr{L}$ is a vector space (space of vector fields) over a field $\mathbb{F}$ equipped with a bracket operations $[,] : \mathscr{L} \times \mathscr{L} \longrightarrow \mathscr{L}$ (called the Lie bracket or commutator) which is satisfied the following properties;*

[i] Closure: $\forall\ X, Y \in \mathscr{L} :\ [X, Y] \in \mathscr{L}$

[ii] Bilinear: $\forall \alpha, \beta \in \mathbb{F}$ *and* $X, Y, Z \in \mathscr{L}$*:* $[\alpha X + \beta Y, Z] = \alpha[X, Z] + \beta[Y, Z]$

[iii] Anti-symmetric: $[X, Y] = -[Y, X]$

[iv] Jacobi Identity: $\forall X, Y, Z \in \mathscr{L}$*:* $[X, [Y, Z]] + [Y, [Z, X]] + [Z, [X, Y]] = 0$

Definition 1.2.5 (Structure Constants [21] §2) *If the vector fields $X_1, \ldots, X_r$ be a basis of a Lie Algebra $\mathscr{L}$. By the bracket relations $[,]$:*

$$[X_i, X_j] = X_i X_j - X_j X_i = \sum_{k=1}^{r} C_{ij}^k X_k \tag{1.2.11}$$

where C_{ij}^k are called structure constants. If $[X_i, X_j] = 0$ for all generators in the basis, then the Lie algebra is abelian.

Lie algebra appears in several different branches in applied mathematics and physics. Let us look at a well-known Lie algebra in the following example:

Example 1.2.3 *([14]) The space of vectors in $\mathbb{R}^3$ under the cross product is a Lie algebra. The cross product as commutator*

$$[\mathbf{x}_1, \mathbf{x}_2] = \mathbf{x}_1 \times \mathbf{x}_2$$

is closure, bilinear, anti-symmetry and satisfies the Jacobi identity.
Consider the Cartesian basis for $\mathbb{R}^3$:

$$\mathbf{x}_1 = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix}, \quad \mathbf{x}_2 = \begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix}, \quad \mathbf{x}_3 = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

We have

$$\mathbf{x}_1 \times \mathbf{x}_2 = \mathbf{x}_3, \quad \mathbf{x}_1 \times \mathbf{x}_3 = -\mathbf{x}_2, \quad \mathbf{x}_2 \times \mathbf{x}_3 = \mathbf{x}_1$$

with the following nonzero structure constants:

$$c_{12}^3 = c_{23}^1 = c_{31}^2 = 1, \quad c_{21}^3 = c_{32}^1 = c_{13}^2 = -1.$$

Example 1.2.4 *([14] §5) Consider the vector space of generators of Lie point symmetries of* $y_{x,x,x} = y^{-3}$ *that is spanned by*

$$X_1 = \partial_x, \quad X_2 = x\,\partial_x + \frac{3}{4}\,y\,\partial_y$$

Then the commutator of X_1 *with* X_2 *is:*

$$\begin{aligned}[X_1, X_2] &= (X_1(x) - X_2(1))\,\partial_x + (X_1(\frac{3}{4}\,y) - X_2(0))\,\partial_y \\ &= \partial_x = X_1\end{aligned}$$

and other commutators are:

$$[X_1, X_1] = 0, \quad [X_2, X_2] = 0, \quad [X_2, X_1] = -[X_1, X_2] = -X_1$$

then the structure constant for the basis are:

$$\begin{aligned} c^1_{11} &= 0, \quad c^1_{12} = 1, \quad c^1_{21} = -1, \quad c^1_{22} = 0 \\ c^2_{11} &= 0, \quad c^2_{12} = 0, \quad c^2_{21} = 0, \quad c^2_{22} = 0. \end{aligned}$$

A subalgebra $\mathscr{M}$ is a sub-vector space of $\mathscr{L}$ if it is closed under the commutator, $[\mathscr{M}, \mathscr{M}] \subset \mathscr{M}$. 0 and $\mathscr{L}$ are trivial subalgebras. If $[\mathscr{M}, \mathscr{L}] \subset \mathscr{M}$, then $\mathscr{M}$ is said an ideal of $\mathscr{L}$. An important ideal of Lie algebra $\mathscr{L}$ is derived subalgebra denoted by $\mathscr{L}^{(1)}$. It consists of all commutators of elements of $\mathscr{L}$:

$$\mathscr{L}^{(1)} = [\mathscr{L}, \mathscr{L}]$$

If $\mathscr{L}^{(1)} \neq \mathscr{L}$, the process can continue

$$\mathscr{L}^{(i+1)} = [\mathscr{L}^{(i)}, \mathscr{L}^{(i)}] \quad for\ i \geq 1$$

until obtaining a new subalgebras fails.

Indeed, Lie subalgebras and in particular derived algebras are core objects needed by our mapping algorithms.

For an algorithmic treatment in this thesis, we limit the differential equation systems being differential polynomials with coefficients from a computable sub-field of $\mathbb{C}$ (e.g. $\mathbb{Q}$). Some non-polynomial systems can be converted to differential polynomial form by using the `Maple` command, `dpolyform`.

In this thesis, we consider systems of differential polynomial equations R (source) and $\hat{R}$ (target) with n independent variables and m dependent variables. Suppose a system R has independent variables $x = (x^1, \ldots, x^n)$ and dependent variables $u = (u^1, \ldots, u^m)$ and $\hat{R}$ has independent variables $\hat{x} = (\hat{x}^1, \ldots, \hat{x}^n)$ and dependent variables $\hat{u} = (\hat{u}^1, \ldots, \hat{u}^m)$. We consider local analytic mappings Ψ: $(\hat{x}, \hat{u}) = \Psi(x, u) = (\psi(x, u), \phi(x, u))$, so that R is locally and invertibly mapped to $\hat{R}$:

$$\hat{x}^j = \psi^j(x, u), \qquad \hat{u}^k = \phi^k(x, u), \qquad \text{Det Jac}(\Psi) = \text{Det}\frac{\partial(\psi, \phi)}{\partial(x, u)} \neq 0 \tag{1.2.12}$$

where $j = 1, \ldots, n$ and $k = 1, \ldots, m$. The map is locally invertible so the determinant of the Jacobian of the mapping is nonzero:

$$\text{Det Jac}(\Psi) = \text{Det}\frac{\partial(\psi, \phi)}{\partial(x, u)} \neq 0 \tag{1.2.13}$$

where $\frac{\partial(\psi,\phi)}{\partial(x,u)}$ is the usual Jacobian $(n + m) \times (n + m)$ matrix of first order derivatives of the $(n + m)$ functions (ψ, ϕ) with respect to the $(n + m)$ variables (x, u). Our goal is to determine an invertible map from R to $\hat{R}$ by exploiting the Lie symmetry invariance algebra of the source without integrating its equations.

Suppose source system R has an associated Lie symmetry algebra $\mathscr{L}$ with symmetry defining system S. The infinitesimal Lie point symmetries for R are found by seeking vector fields

$$V = \sum_{i=1}^{n} \xi^i(x, u)\frac{\partial}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x, u)\frac{\partial}{\partial u^j} \tag{1.2.14}$$

whose associated one-parameter group of transformations, (1.1.5);

$$\begin{aligned} \tilde{x} &= x + \xi(x, u)\ \epsilon + O(\epsilon^2) \\ \tilde{u} &= u + \eta(x, u)\ \epsilon + O(\epsilon^2). \end{aligned} \tag{1.2.15}$$

Suppose two vector fields $X = \sum_{i=1}^{m+n} \nu^i \frac{\partial}{\partial z^i}$, $Y = \sum_{i=1}^{m+n} \mu^i \frac{\partial}{\partial z^i}$ in a Lie algebra $\mathscr{L}$ and $z = (x, u)$, then the commutator of them is:

$$[X, Y] = XY - YX = \sum_{i=1}^{m+n} \omega^i \frac{\partial}{\partial z^k} \tag{1.2.16}$$

where $\omega^k = \sum_{i=1}^{m+n} \left(\nu^i \mu^k_{z^i} - \mu^i \nu^k_{z^i}\right)$.

Similarly, we suppose that the *Target*, $\hat{R}$, admits symmetry vector fields;

$$\hat{V} = \sum_{i=1}^{n} \hat{\xi}^i(\hat{x}, \hat{u})\frac{\partial}{\partial \hat{x}^i} + \sum_{j=1}^{m} \hat{\eta}^j(\hat{x}, \hat{u})\frac{\partial}{\partial \hat{u}^j}. \tag{1.2.17}$$

where the *Target* infinitesimals $(\hat{\xi}^i, \hat{\eta}^j)$ satisfy a linear homogeneous defining system $\hat{S}$ generating a Lie algebra $\hat{\mathscr{L}}$.

If an invertible mapping between R and $\hat{R}$ exists, then it establishes an isomorphism between every Lie subalgebra of infinitesimal symmetry generators of the source and target. In fact, restricting to a Lie subalgebra $\mathscr{L}'$ of $\mathscr{L}$ with corresponding Lie subalgebra $\hat{\mathscr{L}}'$ of $\hat{\mathscr{L}}$ still enables the existence of such mapping equations Ψ (Fig. 1.2). Identifying such subalgebra leads simple mapping equations. It is important in reducing the computational difficulty of such mapping methods.

Our interest in mapping is motivated by Lyakhov et al. [17], who used Reid's results [23] on the algorithmic determination of structure of Lie symmetry algebras of differential equations to implement their mapping algorithm for ODE. Our approach also has been influenced by the methods of Bluman and Kumei [12, 3] for exploiting the Lie symmetries of a system in the determination of mappings between differential equations. Their method depends on extracting subalgebra by explicit non-algorithmic integration, whereas we use algorithmic differential algebra.

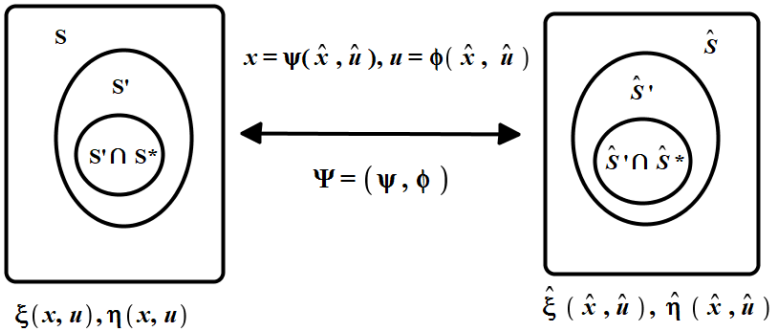


Figure 1.2: The Figure represents the relationship between various solution sets: solutions of the symmetry defining system S generating Lie algebra $\mathscr{L}$ for R and $\hat{S}$ its image under the mapping $\Psi = (\psi, \phi)$. S^* is the symmetry defining system for the $\hat{R}$ in (x, u) coordinates and $\hat{S^*}$ is its image under the mapping Ψ (note $S^* \setminus S$ is usually non-empty). $S^{'}$ is the symmetry defining system for a Lie subalgebra of $\mathscr{L}^{'}$ with the property that $S^{'} \cap S^* = S \cap S^*$. Also $\hat{S}^{'}$ is its image under the mapping Ψ.

1.3 Symbolic differential elimination completion algorithms

Differential elimination completion DEC algorithms apply a finite number of differentiation and linear eliminations to symmetry determining systems. Before we discuss a systematic DEC algorithm, we invite the reader to try simplifying the BS determining system (1.2.10) by such operations. Indeed the first equations of (1.2.10), $\sigma_v = 0$ implies that $\tau_{s,v} = 0$ by differentiation, and substitution in $s^2\tau_{s,v} - 2\sigma_v = 0$ gives $\sigma_v = 0$, which then reduce the $\eta_{v,v} = 0$. So by a few differentiations and eliminations, the first two equations of (1.2.10) become $\tau_s = 0,\ \tau_v = 0,\ \sigma_v = 0,\ \eta_{v,v} = 0$.

Example 1.3.1 *Application of a differential-elimination algorithm to (1.2.10) system augmented with $\eta_v = \eta/v$ yields:*

$$\eta_s = \frac{(3\,\tau_t s + 4\,\sigma_t)\,v}{4s^2}, \eta_t = \frac{(17\,\tau_t s - 2\,\tau_{t,t} s + 12\,\sigma_t)\,v}{8s}, \eta_v = \frac{\eta}{v},$$
$$\tau_{t,t,t} = 0, \sigma_{t,t} = 0, \sigma_s = \frac{s\,\tau_t + 2\,\sigma}{2s}, \tau_s = 0, \sigma_v = 0, \tau_v = 0 \tag{1.3.1}$$

Generally speaking, DEC algorithms reduce the order of differential systems, expose simple equations and decouple equations to get an equivalent system where structural information about solutions is visible. Much progress has been made in computer implementation of exact differential elimination methods. See especially: Schwarz [29, 30], the `differential Gröbner basis` by Mansfield [18, 19], the `DifferentialThomas` Package [33, 24], the `DifferentialAlgebra` package [5, 6, 15] and the `rifsimp` algorithm in the `PDETools` package [26]. Differential elimination algorithms require a ranking which plays a similar role to that used in Guassian elimination.

First, we introduce the crucial aspect in all DEC algorithms known as ranking or ordering of the derivatives.

Definition 1.3.1 (Ranking [34]) *The relation $\preceq$ is ranking on the set $S \subseteq \{u, \underset{1}{u}, \underset{2}{u}, \dots\}$ if*

- *$\preceq$ is a reflexive, transitive relation on S*
- *$\preceq$ is a total order of S, that is for $a, b \in S$ exactly one of the three conditions holds: $a \preceq b$, or $b \preceq a$, or $a = b$*
- *$\preceq$ is positive, that is for any $a, \partial_{x^i} a \in S$ and all $1 \leq i \leq n$ we have $a \prec \partial_{x^i} a$*
- *$\preceq$ is invariant under differentiation, that is for any $a, b, \partial_{x^i} a, \partial_{x^i} b \in S$ and all $1 \leq i \leq n$, $a \prec b \Rightarrow \partial_{x^i} a \prec \partial_{x^i} b$*

where $a \prec b$ is defined in the expected way, namely $a \prec b \Rightarrow a \preceq b, a \neq b$.

A complete theory and description of such rankings is given by Rust [26].

Example 1.3.2 *Consider the determining system related to the Black-Schole's example 1.3.1 with independent variables (s, t) and dependent variables σ, τ, η, an admissible ranking for the system (1.2.10) is;*

$$\tau \prec \sigma \prec \eta \prec \tau_t \prec \sigma_t \prec \eta_t \prec \tau_s \prec \sigma_s \prec \eta_s \prec \tau_{tt} \prec \sigma_{tt} \prec \eta_{tt} \prec \tau_{ts} \prec \sigma_{ts} \prec \eta_{ts} \prec \cdots$$

Consider a polynomially nonlinear system of differential equations $R = \{R^1, \dots, R^\ell\} = 0$ with independent variable $x = (x^1, x^2, \dots, x^n)$ and dependent variables $u = (u^1, \dots, u^m)$ over $\mathbb{C}$ (or $\mathbb{R}$) with coefficients from some computable extension of $\mathbb{Q}$. Replacing solutions and derivatives by formal (Jet) variables allows manipulation of equations without first assuming that solutions exist [7, 27, 31]. In particular, denoting the ith order derivatives of u by $\underset{i}{u}$, then the total derivative order q, where $R^s : J^q \to \mathbb{C}$, $J^q = \mathbb{C}^{N(n,m,q)}$ is;

$$D_{x^j} = \frac{\partial}{\partial x^j} + \sum_{\ell=1}^{m} u^\ell_{x^j} \frac{\partial}{\partial u^\ell} + \cdots$$

Here $N(n, m, q)$ is the number of jet variables of order less than or equal to q:

$$N(n, m, q) = n + m \left(\begin{array}{c} n+q \\ q \end{array} \right).$$

The jet variety of the system in the jet space $J^q(\mathbb{R}^n, \mathbb{R}^m)$ is;

$$\mathscr{V}(R) = \left\{ (x, \underset{0}{u}, \underset{1}{u}, \dots, \underset{q}{u}) \in J^q : R(x, \underset{0}{u}, \underset{1}{u}, \dots, \underset{q}{u}) = 0 \right\}.$$

As in [27, 5, 31], studying differential equations by their Jet variety allows us to manipulate equations without knowing if their solutions exist. By differentiation, the graphs of solutions can be embedded as curves lying in the jet variety. The union of these prolonged graphs yields the solution variety $\mathscr{V}_{\text{sol}}(R)$. Then $\mathscr{V}_{\text{sol}}(R) \subseteq \mathscr{V}(R)$. Whenever $\mathscr{V}_{\text{sol}}(R) = \mathscr{V}(R)$ the differential equation is said to be locally solvable. Generally, a differential equation is locally solvable at a point of its variety if at least one prolonged graph of a smooth solution passes through that point [26].

In order to understand differential elimination algorithms, we need to recall the two fundamental operations in Differential Algebra called projection and prolongation. A single prolongation of the Jet variety of the qth order system denoted by $D(R)$ is the list of first order total derivatives of all equations of R with respect to all the independent variables.

$$D(R) = \left\{ (x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q+1}{u}) \in J^{q+1} : R = 0, \;\; D_{x^i} R^j = 0, \; i = 1, \ldots, n, \;\; j = 1, \ldots, \ell \right\}$$

It extends the locus of a differential system from lower order to higher order of Jet space. A single projection is;

$$\pi(R) = \left\{ (x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q-1}{u}) \in J^{q} : R(x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q}{u}) = 0 \right\}.$$

which is the mapping from higher order to lower order Jet space for example, from J^q to J^{q-1}.

Many differential systems arising in applications are over determined systems such as the determining systems arising in the process of analyzing symmetry. Such systems require finite number of prolongations and projections to complete them to a form that includes all their integrability conditions. This form is known as the involutive form where an existence uniqueness theorem is available.

The full method to complete partial differential equations is the Cartan-Kuranishi Algorithm [10, 13]. Algorithm 1 is a simplified Cartan-Kuranishi Algorithm. This method prolong the system to order $q + 1$, then project to order q to test for the existence of new constraints. This continued until no new constraints are found.

Algorithm 1

Input: System $R = (R^1, \ldots, R^\ell) = 0$
Output: Involutive form of R
While $R \neq (\pi o D)R$ **repeat** $R := (\pi o D)R$

The above algorithm succeeds in making the involutive form for many ordinary and partial differential equations. It has been improved and made more widely applicable by the concept of symbol matrix. The properties of the symbol are used to determine the integrability conditions of partial differential equations.

Definition 1.3.2 (Symbol) *The Jacobian of $R = (R^1, \ldots, R^\ell) = 0$ with respect to the highest derivatives of the system is denoted symbol where q is the order of the system.*

$$Symbol\ R := \frac{\partial R}{\partial \underset{q}{u}}$$

The symbol is the coefficient of the highest derivatives in the prolongations equations DR, D^2R, $\ldots$ of R.

The symbol of a qth order system R is involutive if

$$\text{rank Symbol}(DR) := \sum_{j=1}^{n} j\beta_j^{(q)} \tag{1.3.2}$$

and the $\beta_j^{(q)}$ are the dimensions of certain subspaces of the Null Space of the Symbol of R. See [31] for more details.

Algorithm 2 Full Cartan-Kuranishi

Input: System $R = (R^1, \ldots, R^\ell) = 0$
Output: Involutive form of R
repeat
  While $R \neq (\pi o D)R$ **repeat** $R := (\pi o D)R$
  While Symbol R is not involutive **repeat** $R := DR$
until Symbol R is involutive and $R := (\pi o D)R$

In the Full Cartan-Kuranishi algorithm, if the symbol is involutive, then the algorithm has terminated and return the involutive form of the system. Otherwise, the system is prolonged until the symbol of the resulting q- order system becomes involutive. The involutivity test by the symbol is the more technical part of the Cartan-Kuranishi algorithm [10, 13, 31].

1.4 Numerical algebraic geometry

Ordered Gaussian-elimination is generally unstable when applied to linear systems with approximate coefficients. Similarly, symbolic differential elimination algorithms such as, `rifsimp`, are less suitable in approximate case since the dependency on the ordering of the variables in such methods leads numerical instability. This instability has motivated the development of a new generation of symbolic-numeric completion methods for systems of differential equations [38, 37]. These methods such as representing the variety via certain points, give a coordinate independent description of properties of systems in Jet space. Alternatively, such coordinate independent methods have greater numerical stability. In this section, we give an overview of such methods.

Sommese et al. introduced a numerical algebraic geometry method that represents the irreducible components of an algebraic variety by certain points (witness points). Witness points are computed as the intersection of irreducible components of a variety with a random linear system. In particular, the method yields 0-dimensional system by appending a random linear space to a given algebraic system [32].

For example, suppose a non-constant polynomial $f(u, v) = 0$ in $\mathbb{C}[u, v]$ intersected by a random line $au + bv + c = 0$. The roots of $\{f(u, v) = 0,\ au + bv + c = 0\}$ will be called witness points. Based on the Fundamental Theorem of Algebra this system has at least one complex root (a witness point). Obviously, this idea will fail in the real case. For instance, let a circle $f(u, v) = u^2 + v^2 - 1 = 0$ in $\mathbb{R}[u, v]$. Then, a random real line $au + bv + c = 0$ will not intersect the circle with high probability. There has been remarkable progress in developing more appropriate witness point methods for the real case such as the critical point method [1, 36]. Generally speaking, critical point methods extremize the distance of a random point (or plane) to a given variety.

Definition 1.4.1 *[32, 2] Consider f as a system of k polynomials $f = (f^1, f^2, \ldots, f^k) = 0$ from $\mathbb{C}[x^1, \ldots, x^n]$. Geometrically, the complex variety $\mathcal{V}(f)$ can be decomposed into irreducible components which are represented by witness points obtained by slicing the variety*

with random planes of equal co-dimension.

$$\mathscr{V}(f) = \bigcup_{i=0}^{dim\mathscr{V}(f)} \mathscr{V}_i = \bigcup_{i=0}^{dim\mathscr{V}(f)} \bigcup_{j\in\Lambda_j} \mathscr{V}_{ij}$$

where $\mathscr{V}_i$ is the pure i-dimensional component of $\mathscr{V}(f)$ consisting of all i-dimensional irreducible components $\mathscr{V}_{ij}$ and Λ_j is a finite indexing set for the i-dimensional irreducible components.

A witness set for a i-dimensional solution component consists of i random hyperplanes and isolated solutions out by the intersection of the component with those hyperplanes. In particular, a numerical irreducible decomposition of $\mathscr{V}(f)$ has the form

$$\bigcup_{i=0}^{dim\mathscr{V}(f)} \bigcup_{j\in\Lambda_j} \mathscr{W}_{ij}$$

where $\mathscr{W}_{ij}$ is a witness set for a distinct i-dimensional irreducible component of $\mathscr{V}(f)$.

Example 1.4.1 *[2] Consider the algebraic system*

$$\begin{aligned} f = \{&(y-x^2)(x^2+y^2+z^2-1)(x-2) = 0,\\ &(z-x^3)(x^2+y^2+z^2-1)(y-2) = 0,\\ &(z-x^3)(y-x^2)(x^2+y^2+z^2-1)(z-2) = 0\} \end{aligned}$$

The variety (solution set) of f can be decomposed as;

$$\mathscr{V}(f) = \mathscr{V}_2 \cup \mathscr{V}_1 \cup \mathscr{V}_0$$

Where $\mathscr{V}_2 = \mathscr{V}_{2,1}$ is a surface (sphere), $\mathscr{V}_1 = \mathscr{V}_{1,1} \cup \mathscr{V}_{1,2} \cup \mathscr{V}_{1,3} \cup \mathscr{V}_{1,4}$ where $\mathscr{V}_{1,1}$ is the cubic curve and other components are the three lines and $\mathscr{V}_0 = \mathscr{V}_{0,1}$ is just the isolated point $(2, 2, 2)$.

Applying directly such point-based methods to differential equations regarded as algebraic equations in Jet space is not geometrically correct. Wu, Reid, and Golubitsky [38] extend these algebraic concepts to differential equations. Finding such witness points in a variety of a differential system R lies on the below property;

$$\pi^r \mathscr{V}(D^k(R_{invol})) = \mathscr{V}(R_{invol}) \subseteq \mathscr{V}(R);\ \ for\ any\ r, k \in \mathbb{N}$$

where R_{invol} is an involutive form of R. More precisely, since R_{invol} contains all integrability conditions so there is a power series solution of system R order by order at witness points in $\mathscr{V}(R_{invol})$. Some progress has been made towards numeric involutive form based on witness points (witness Jet points) in complex variety [38, 37].

Example 1.4.2 *Consider a simple system of differential equations R which involves one independent variable t and one dependent variable u.*

$$R := \{(u_t^2 + u^2 + t^2 - 1)(u-2) = 0, \quad (u_t^2 + u^2 + t^2 - 1)(u_t - 2) = 0\} \tag{1.4.1}$$

Replace $\{u \to X, u_t \to Y\}$ *yields an algebraic system with a variety*

$$\mathscr{V}_{\mathbb{R}}(R) := \{(X,Y,t) \in \mathbb{R}^3;\ Y^2+X^2+t^2-1=0,\ X-2=0,\ Y^2+X^2+t^2-1=0,\ Y-2=0\} \quad (1.4.2)$$

Decomposition of $\mathscr{V}(R)$ *yields two connected components (manifolds): a sphere and a line,* $\mathscr{V}(R) = S \cup L$ *in* $\mathbb{R}^3 \simeq J^1$.
Consider the two cases $Y^2 + X^2 + t^2 - 1 \neq 0$ *and* $Y^2 + X^2 + t^2 - 1 = 0$. *Lets look at L, where we might anticipate* $L \simeq$ *is a solution of* $\mathbb{R}$. *However* $L = \{(X,Y,t);\ X-2=0, Y-2=0\}$ *in* $\mathscr{V}(R)$ *and* $\{(u_t,u,t);\ u-2=0, u_t-2=0\}$ *in R. But if* $u = 0$ *on some open connected interval* $I \subseteq \mathbb{R} \ \Rightarrow u_t = 0$. *No such solution exits on any such* $I \neq \emptyset$. *So L does not correspond to solution. Similarly consider S. To compare solution to S, we must take graphs of solutions* $\{(u,t);\ u_t^2 + u^2 + t^2 - 1 = 0,\ t \in I\}$ *and extend (prolong) them into 3-dimensional space* $\{(u_t,u,t);\ t \in I\} \subseteq \mathbb{R}^3$.

This idea motivates us to generalize differential-elimination completion algorithms to a more frequently applicable cases involving approximate real coefficients that appears in chapter 4 of this thesis.

1.5 Thesis Outline

1.5.1 Contents of Chapter 2: "Introduction of the MapDE Algorithm for Mappings Relating Differential Equations"

Symmetry analysis of differential equation systems is a major tool for studying the behavior of their solutions. Applications include finding exact solutions, mapping solutions of a differential equation to other differential equation solutions, and solving mapping problems. This thesis aims to develop an algorithm that determines whether a given system can be mapped into an equivalent system of interest by exploiting symmetry.

In this chapter, we introduce and demonstrate the core of our mapping algorithm called `MapDE`, which takes a source differential equation and the desired target as input and returns the reduced involutive form of the mapping system if such a mapping exists. For algorithmic implementation, we restrict our treatment to differential polynomial systems DPS, which are polynomially nonlinear functions of their derivatives and dependent variables with coefficients from some computable field (e.g., $\mathbb{Q}$). In addition, we give an algorithm for invertible mapping a linear differential polynomial equation to linear differential polynomial equation with constant coefficients by exploiting symmetry (in particular the existence of on derived Lie algebra).

We are motivated by some aspects of the Bluman-Kumei mapping approach for PDE and remarkable work by Lyakhov and collaborators who developed a linearization algorithm for ODE by using Reid's result [23] on their algorithmic determination of the structure of Lie group of symmetries of ODE.

1.5.2 Contents of Chapter 3: "Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDS to linear PDS"

This chapter is a sequel to the previous chapter. We extend `MapDE` to determine if a nonlinear DPS is linearizable by an invertible mapping.

`MapDE` algorithm involves several tests to exclude non-linearizable systems. The most important necessary condition for Linearizability is based on a differential Hilbert Function to test the equality of the size of the solution space of the given system with the size of the solution space of the symmetry subgroup for linear superposition. This feature is a crucial hypothesis that seems to be missed from Bluman et al. [3]. We compare the result of `MapDE` algorithm with Lyakhov et al. algorithm through a set of test examples.

1.5.3 Contents of Chapter 4: "A Completion algorithm for real differential polynomial systems to Involutive Form"

Most differential systems arising in applications contain approximate coefficients. The dependency of the `MapDE` on exact differential elimination completion algorithms that are unstable on differential systems with floating-point coefficients motivate us to improve the existing symbolic-numeric completion algorithms.

This chapter is a sequel to the work of Reid and his collaborators on Numerical Jet Geometry in the complex case. The goal of that work is to extend concepts from Numerical Algebraic Geometry to differential polynomial systems in Jet space. Geometrically, the components of differential polynomial systems are represented by approximate complex witness points, which are the intersection of random linear spaces with the components. Homotopy-based numerical solver can compute such witness points efficiently, but this idea fails in the real case.

This chapter is influenced by remarkable work by Wu et al. [36] for the real case. In that case, real witness points extremize the distance of a random point (or plane) to the variety and are called critical points. The goal is to find at least one critical point on each connected component. We have improved the symbolic-numeric completion algorithm based on this idea. We implement our algorithm in `Maple` with an interface to `Bertini` which is a homotopy-based solver [2].

We conclude with a discussion of our results and future work in chapter 5. Examples of our `MapDETools` package are given in Appendix A.

Bibliography

[1] G.M. Besana, S. Di Rocco, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler. Cell decomposition of almost smooth real algebraic surfaces. Num. Algorithms, DOI:10.1007/s11075-012-9646-y, published online Sept 28, 2012. 15, 73

[2] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler, Numerically Solving Polynomial Systems with the Software Package Bertini. In preparation. To be published by SIAM, 2013. 15, 16, 18, 73, 81, 84, 94

[3] G.W. Bluman, A.F. Cheviakov, and S.C. Anco. Applications of Symmetry Methods to Partial Differential Equations, Springer, 2010. 1, 6, 8, 11, 18, 23, 24, 27, 30, 31, 32, 50, 51, 62, 63, 64, 93, 94, 95

[4] Bluman and Kumei. Symmetries and Differential Equations. SpringerVerlag, 1989. 6, 93

[5] F. Boulier, D. Lazard, F. Ollivier, and M. Petitot. Representation for the radical of a finitely generated differential ideal. Proc. ISSAC '95, 158–166, 1995. 12, 13, 42, 44, 49, 57

[6] F. Boulier, D. Lazard, F. Ollivier, M. Petitot. Computing representations for radicals of finitely generated differential ideals. Journal of AAECC, 2009. 12, 42, 44

[7] F. Boulier, D. Lazard, F. Ollivier, and M. Petitot. Representation for the radical of a finitely generated differential ideal. Proc. ISSAC 1995. ACM Press, 158–166, 1995. 13, 73

[8] J. Carminati and K. Vu. Symbolic computation and differential equations: Lie symmetries. Journal of Symbolic Computation, 29, 95–116, 2000. 6, 24, 46, 87

[9] K.T. Vu, G.F. Jefferson and J. Carminati. Finding generalised symmetries of differential equations using the MAPLE package DESOLVII. Comput. Phys. Commun., 183, 1044–1054, 2012. 8

[10] Cartan. Systèmes différentiels, théories d'équivalence. Oeuvres complèes, Part 2, Vol. 2: Groupes finis, 1953. 14, 15, 82, 93

[11] A.F. Cheviakov. GeM software package for computation of symmetries and conservation laws of differential equations. Computer Physics Communications, 176(1), 48–61, 2007. https://doi.org/10.1016/j.cpc.2006.08.001 6, 8, 24, 46, 87

[12] S. Kumei and G.W. Bluman. When nonlinear differential equations are equivalent to linear differential equations. SIAM Journal of Applied Mathematics, 42, 11571173, 1982. 11, 23, 24, 43

[13] M. Kuranishi. On E. Cartans Prolongation Theorem of Exterior Differential Systems. Amer.J. of Math., 79, 1–47, 1957. 14, 15, 82

[14] P. Hydon. Symmetry Methods for Differential Equations: A beginners guide. Cambridge Univ. Press, Cambridge, 2000. 1, 6, 9, 10

[15] F. Lemaire. Contribution à l'algorithmique en algèbre différentielle, Université des Sciences et Technologie de Lille - Lille I, 2002. 12, 44

[16] I.G. Lisle, and G.J. Reid. Geometry and Structure of Lie Pseudogroups from Infinitesimal Defining Systems. J. Symb. Comput., 26(3), 355–379, 1998. 29, 43, 47, 51

[17] D.A. Lyakhov, V.P. Gerdt, and D.L. Michels. Algorithmic Verification of Linearizability for Ordinary Differential Equations. In Proc. ISSAC '17. ACM, 285–292, 2017. 11, 23, 28, 35, 43, 48, 49, 52, 56, 57, 60, 61, 63, 64, 93, 94

[18] E. Mansfield. Differential Gr"obner bases. PhD thesis, University of Sydney, 1991. 12

[19] E.L. Mansfield and P.A. Clarkson. Applications of the differential algebra package diffgrob2 to classical symmetries of differential equations. J. Symb. Comput., 23, 517–533, 1997. 12

[20] T. Masebe and J. Manale. New symmetries of Black-Scholes equation. In Proc. AMCM '13., 221–231, 2013. 1, 25, 35

[21] P. Olver. Equivalence, invariance, and symmetry. Cambridge University Press, 1995. 2, 3, 4, 9, 23, 27, 42, 48, 50, 52, 65, 93

[22] P. Olver. Application of Lie groups to differential equations. Springer-Verlag, 2nd edition, 1993. 4, 25, 30, 44, 73, 86

[23] G.J. Reid. Finding abstract Lie symmetry algebras of differential equations without integrating determining equations. European Journal of Applied Mathematics 2, 319–340, 1991. 11, 17, 23, 43

[24] D. Robertz. Formal Algorithmic Elimination for PDEs. Lect. Notes Math. Springer, 2121, 2014. 12, 42, 44, 49, 57, 63

[25] T. Rocha Filho and A. Figueiredo. SADE: A Maple package for the symmetry analysis of differential equations. Computer Physics Communications, 182(2), 467–476, 2011. 6, 24, 46, 87

[26] C.J. Rust, G.J. Reid, and A.D. Wittkopf. Existence and uniqueness theorems for formal power series solutions of analytic differential systems. Proc. ISSAC '99, 105–112, 1999. 12, 13, 29, 42, 44, 49, 57

[27] C.J. Rust and G.J. Reid. Rankings of partial derivatives. In Proc. ISSAC '97, 9–16, 1997. 13, 29, 44

[28] F. Schwarz. Algorithmic Lie theory for solving ordinary differential equations. Chapman and Hall / CRC Press, 2008. 5

[29] F. Schwarz. An algorithm for determining the size of symmetry groups. Computing, 49, 95–115, 1992. 12

[30] F. Schwarz. Reduction and completion algorithms for partial differential equations, Proc ISSAC92, New York, ACM, 49–56, 1992. 12

[31] W.M. Seiler. Involution: The formal theory of differential equations and its applications in computer algebra. Algorithms and Computation in Mathematics, Vol. 24, Springer, 2010. https://doi.org/10.1007/978-3-642-01287-7 13, 15, 25, 28, 42, 44, 73, 75, 81

[32] A.J. Sommese and C.W. Wampler. *The Numerical solution of systems of polynomials arising in engineering and science*. World Scientific Press, 2005. 15, 73, 78

[33] J.M. Thomas. Differential Systems. AMS Colloquium Publications, XXI, 1937. 12, 44, 49, 63

[34] A. Wittkopf. Algorithms and Implementations for Differential Elimination. *Ph.D. Thesis*, Simon Fraser University, 2004. 13, 76

[35] T. Wolf. Investigating differential equations with crack, liepde, applsymm and conlaw. Handbook of Computer Algebra, Foundations, Applications, Systems, 37, 465–468, 2002. 6, 7, 8, 35, 43, 65, 87

[36] W. Wu, C. Chen and G. Reid. Penalty Function Based Critical Point Approach to Compute Real Witness Solution Points of Polynomial System *Computer Algebra in Scientific Computing CASC*, 377–391, 2017. 15, 18, 72, 73, 78, 79, 80, 81, 88, 94, 95

[37] W. Wu and G. Reid. Application of Numerical Algebraic Geometry and Numerical Linear Algebra to PDE. Proc. of ISSAC '06, 345–352, ACM, 2006. 15, 16, 78

[38] W. Wu, G. Reid and O. Golubitsky. Towards Geometric Completion of Differential Systems by Points. Approximate Commutative Algebra. Texts and Monographs in Symbolic Computation (A Series of the Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria). Springer, Vienna, 79–97, 2009. 15, 16, 78, 82, 88

Chapter 2

Introduction of the MapDE Algorithm for Mappings Relating Differential Equations

This paper is the first of a series in which we develop exact and approximate algorithms for mappings of systems of differential equations. Here we introduce the `MapDE` algorithm and its implementation in Maple, for mappings relating differential equations. We consider the problem of how to algorithmically characterize, and then to compute mappings of less tractable (*Source*) systems R to more tractable (*Target*) systems $\hat{R}$ by exploiting the Lie algebra of vector fields leaving R invariant. Suppose that R is a (*Source*) system of (partial or ordinary) differential equations with independent variables $x = (x^1, x^2, \ldots, x^n) \in \mathbb{C}^n$ and dependent variables $u = (u^1, \ldots, u^m) \in \mathbb{C}^m$. Similarly suppose $\hat{R}$ is a (*Target*) system in the variables $(\hat{x}, \hat{u}) \in \mathbb{C}^{n+m}$. For systems of exact differential polynomials R, $\hat{R}$ our algorithm `MapDE` can decide, under certain assumptions, if there exists a local invertible mapping $\Psi(x, u) = (\hat{x}, \hat{u})$ that maps the *Source* system R to the *Target* $\hat{R}$. We use a result of Bluman and Kumei who have shown that the mapping Ψ satisfies infinitesimal (linearized) mapping equations that map the infinitesimals of the Lie invariance algebra for R to those for $\hat{R}$.

`MapDE` involves applying the differential-elimination algorithm to the defining systems for infinitesimal symmetries of R, $\hat{R}$, and also to the nonlinear mapping equations (including the Bluman-Kumei mapping subsystem); returning them in a form which includes its integrability conditions and for which an existence uniqueness theorem is available. Once existence is established, a second stage can determine features of the map, and some times by integration, explicit forms of the mapping. Examples are given to illustrate the algorithm.

Algorithm `MapDE` also allows users to enter broad target classes instead of a specific system $\hat{R}$. For example avoiding the integrations of the Bluman-Kumei approach `MapDE` can determine if a linear differential equation R can be mapped to a linear constant coefficient differential equation.

2.1 Introduction

This paper is the first of a series in which we explore algorithmic aspects of mappings of differential equation systems that transform differential equations (DEs) to DEs. Naturally this exploration includes symmetry transformations – transformations of a DE to itself, and also

equivalence transformations where one member of a class of DEs is mapped to another member of the class. We introduce the algorithm `MapDE` for characterizing mappings between DEs. For algorithmic implementation we restrict our treatment to differential polynomial systems (DPS), systems which are polynomially nonlinear functions of their derivatives and dependent variables; with coefficients from some computable field (e.g. $\mathbb{Q}$).

In earlier work we developed approximate methods for determination of approximate Lie symmetry algebra of DEs [9, 19]. A key motivation for our current work is how to practically use such approximate methods. We see determination of approximate mappings of DPS, to be explored later in this series, as a practical way in which to exploit such approximate symmetry information. Our interest in mappings was also motivated by recent work [21], which used Reid [28] on the algorithmic determination of structure of Lie algebras of symmetries of DE, to give an algorithm to determine the existence of mappings exactly linearizable ODE. We give an algorithmic implementation of the methods of Bluman and Kumei [8, 17] for exploiting the Lie symmetries of a system in the determination of mappings between DEs.

In particular in this paper we introduce an algorithm for such mappings in the presence of symmetry. The algorithm `MapDE` is implemented as part of Huang and Lisle's LAVF object-oriented Maple package [18]. We give examples to illustrate the algorithm and compare it with the approach of Bluman and Kumei. We extend the algorithm, to determine the existence of a mapping from linear DE, to linear constant coefficient DE, avoiding the heuristic integrations of Bluman and Kumei's approach.

We consider systems of (partial or ordinary) differential equations with n independent variables and m dependent variables. Suppose R has independent variables $x = (x^1, \ldots, x^n)$ and dependent variables $u = (u^1, \ldots, u^m)$ and $\hat{R}$ has independent variables $\hat{x} = (\hat{x}^1, \ldots, \hat{x}^n)$ and dependent variables $\hat{u} = (\hat{u}^1, \ldots, \hat{u}^m)$. In particular we consider local analytic point mappings Ψ: $(\hat{x}, \hat{u}) = \Psi(x, u) = (\psi(x, u), \phi(x, u))$, so that R is locally and invertibly mapped to $\hat{R}$:

$$\hat{x}^j = \psi^j(x, u), \qquad \hat{u}^k = \phi^k(x, u) \tag{2.1.1}$$

where $j = 1, \ldots, n$ and $k = 1, \ldots, m$. The mapping is locally invertible so the determinant of the Jacobian of the mapping is nonzero:

$$\text{Det Jac}(\Psi) = \text{Det}\frac{\partial(\psi, \phi)}{\partial(x, u)} \neq 0, \tag{2.1.2}$$

where $\frac{\partial(\psi,\phi)}{\partial(x,u)}$ is the usual Jacobian $(n+m)\times(n+m)$ matrix of first order derivatives of the $(n+m)$ functions (ψ, ϕ) with respect to the $(n + m)$ variables (x, u). Note throughout this paper, we will call $\hat{R}$ the *Target* system of the mapping, which will generally have some more desirable features than R, which we call the *Source* system.

Algorithms for existence of such mappings, and methods for their explicit construction, is the topic of this paper. A very general approach to such problems, Cartan's famous Method of Equivalence [27], finds invariants, that label the classes of systems, equivalent under the pseudogroup of such mappings. The fundamental importance of such equivalence questions, and the associated demanding computations has attracted attention from symbolic computation researchers. For example, Neut, Petitot and Dridi [25], implemented Cartan's method for ODE and certain classes of PDE of finite type (i.e. with finite dimensional solution space). Olver and collaborators developed a new version of Cartan's moving frames [12]. Valiquette [34] applied

this method to equivalence problems and further results are given by Arnaldson [3]. Also see [2] which introduces the `DifferentialGeometry` package, available in `Maple` and has been applied to equivalence problems. Also see [15, 22] for approaches to the non-commutative calculus that results in calculations. Underlying these calculations, is that overdetermined PDE systems, with some non-linearity, are required to be reduced to forms that enable the statement of a local existence and uniqueness theorem (such include passive and involutive forms). See [14] for estimates of complexity of such methods, which indicate their difficulty.

Our initial approach, is fairly direct, and exploits the linearity of the Bluman-Kumei mapping equations. It also is motivated by in the longer term, we wish to include invariant differential operators and using the newly developed methods of Numerical Jet Geometry, to investigate approximate equivalence.

In contrast, Bluman and Kumei [17] consider a narrower class of mapping problems, which is focused on the case where the *Target* system is uniquely characterized in terms of its Lie symmetry invariance algebra (See [8, 17]). In this article we will implement an algorithm based on the Bluman and Kumei approach.

Suppose that a *Source* system, has an associated Lie symmetry algebra, together with its defining system. Such infinitesimal Lie *point* symmetries for R are found by seeking vector fields

$$V = \sum_{i=1}^{n} \xi^i(x,u)\frac{\partial}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x,u)\frac{\partial}{\partial u^j} \tag{2.1.3}$$

whose associated one-parameter group of transformations

$$\begin{aligned} x^* &= x + \xi(x,u)\epsilon + O(\epsilon^2) \\ u^* &= u + \eta(x,u)\epsilon + O(\epsilon^2) \end{aligned} \tag{2.1.4}$$

which away from exceptional points preserves the jet locus of such systems - mapping solutions to solutions. See [6, 7] for applications. For a vector field (2.1.3) acting on the space with coordinates (x,u), the operator $\frac{\partial}{\partial x^i}$ denotes partial derivative with respect to x^i, holding x^j, $j \neq i$ constant, and u constant; with $\frac{\partial}{\partial u^k}$ similarly defined. The *infinitesimals* (ξ^i, η^j) of a symmetry vector field (2.1.3) for a system of DEs are found by solving an associated system of linear homogeneous defining equations (or determining equations) for the infinitesimals. The defining system is derived by an explicit algorithm, for which numerous computer implementations are available [10, 11, 30]. Similarly we suppose that the *Target* admits symmetry vector fields

$$\hat{V} = \sum_{i=1}^{n} \hat{\xi}^i(\hat{x},\hat{u})\frac{\partial}{\partial \hat{x}^i} + \sum_{j=1}^{m} \hat{\eta}^j(\hat{x},\hat{u})\frac{\partial}{\partial \hat{u}^j} \tag{2.1.5}$$

in the *Target* infinitesimals $(\hat{\xi}, \hat{\eta})$. Computations with defining systems of both systems will be essential in our approach. We have implemented our algorithms in Huang and Lisle's powerful object oriented LAVF, `Maple` package [18].

In §2.2 we give preliminaries and the Bluman-Kumei Mapping equations, together with a simple illustrative example. In §2.3 we describe our core algorithm `MapDE`, which takes R and $\hat{R}$ as input, and returns the reduced involutive form RIF-form for Ψ, establishing existence, non-existence of the mapping. Once existence of the mapping is established, a further phase, is to try to obtain an explicit form for the mapping by integrating the mapping equations.

We treat two cases: one in which R and $\hat{R}$ are specified and another where `TargetClass = ConstantCoeffDE`. In §2.4 we give examples of application `MapDE` and conclude with a discussion in §2.5.

2.2 Preliminaries & Mapping Equations

For an algorithmic treatment, we limit the systems considered to being differential polynomials, with coefficients from a computable subfield of $\mathbb{C}$ (e.g. $\mathbb{Q}$). Some non-polynomial systems can be converted to differential polynomial form by the use of the Maple command, `dpolyform`.

The geometric approach to DEs centers around the jet locus, where the derivatives are regarded as formal variables and a map to polynomials, in our case, where the tools of algebraic geometry can be used. In general systems of polynomial equations and inequations must be considered (differences of varieties). The union of prolonged graphs of local solutions is a subset of the jet locus in J^q, the jet space of order q. For details concerning the Jet geometry of DEs see [26, 33].

Example 2.2.1 *Consider the famous Black Schole's equation which is fundamental in financial applications [23], we will use as an introductory simple example:*

$$\frac{\partial}{\partial t}v + \frac{s^2}{2}\left(\frac{\partial}{\partial s}\right)^2 v + s\frac{\partial}{\partial s}v - v = 0 \tag{2.2.1}$$

By inspection this equation has the obvious symmetry of translation in $t : t^* = t + \epsilon$ and scaling in $s : s^* = bs$. Moreover the infinitesimal form of these symmetries (2.1.4) is generated by the operators (2.1.3) given by $\frac{\partial}{\partial t}$ and $s\frac{\partial}{\partial s}$. Since these vector fields obviously commute, it is natural to map to new coordinates in which:

$$\frac{\partial}{\partial t} = \frac{\partial}{\partial \hat{t}}, \quad s\frac{\partial}{\partial s} = \frac{\partial}{\partial \hat{s}} \tag{2.2.2}$$

So by trivial integration the transformation $\hat{s} = \int \frac{ds}{s} = \log(s) + c_1$, $\hat{t} = t + c_2$ should map the Black Schole's Equation into an equation invariant under two commuting translations, i.e. to a constant coefficient equation. Indeed by inspection we find:

$$\frac{\partial}{\partial \hat{t}}v + \frac{1}{2}\left(\frac{\partial}{\partial \hat{s}}\right)^2 v - v = 0 \tag{2.2.3}$$

which is the famous Black-Schole's transformation of (2.2.1) to the backwards heat equation. This example simply illustrates that there can be a strong connection between symmetries admitted by an equation and mappings of the equation to convenient forms.

Indeed this illustrates the key idea of Bluman-Kumei's method for determining when a linear differential equation (DE) in n independent variables can be mapped to a (*Target*) linear constant coefficient DE: that the *Target* admits n commuting translations. Geometrically the *Source* must correspondingly admit a subalgebra of its Lie symmetry algebra consisting of n commuting symmetries (that act transitively on the space of its independent variables).

One can try to devise an algorithm for determining such symmetries explicitly. In general this involves integrating systems of overdetermined PDE, and, though advantageous in many applications, no general algorithm is known for this task. In our paper we describe algorithms using a finite number of differentiations and eliminations, and no integrations, that guarantees the algorithmic determination of the existence of such transformations. The computer algebra system `Maple` has several excellent such differential elimination algorithms, and also excellent algorithms for generating the linearized equations for symmetries.

Example 2.2.2 *For the Black-Schole's Equation* (2.2.1)*, the infinitesimal symmetry operator has form* $\sigma(s,t,v)\frac{\partial}{\partial s} + \tau(s,t,v)\frac{\partial}{\partial t} + \eta(s,t,v)\frac{\partial}{\partial v}$*. Here comparing with* (2.1.5) *yields* $(\xi^1, \xi^2) = (\sigma, \tau)$ *and* $\eta^1 = \eta$*. The automatically generated unsimplified system of defining equations for infinitesimal symmetries is:*

$$\begin{aligned}
&\tau_s = 0, \tau_v = 0, \tau_{v,v} = 0, \sigma_{v,v} = 0, \tau_{s,v}s^2 - 2\,\sigma_v = 0,\\
&\eta_{v,v}s - 2\,\sigma_{s,v}s - 4\,\sigma_v = 0,\\
&-2\,\eta_{s,v}s^2 + \sigma_{s,s}s^2 + 2\,\sigma_s s + 6\,\sigma_v v - 2\,\sigma + 2\,\sigma_t = 0,\\
&\tau_{s,s}s^3 - 2\,\tau_s s^2 + 2\,\tau_v sv + 2\,\tau_t s - 4\,\sigma_s s + 4\,\sigma = 0,\\
&-\eta_{s,s}s^3 + 2\,\eta_s s^2 - 2\,\eta_v sv + 4\,\sigma_s sv + 2\,\eta\, s - 2\,\eta_t s - 4\,v\sigma = 0
\end{aligned} \tag{2.2.4}$$

Application of a differential-elimination algorithm to this system augmented with $\eta_v = \eta/v$ *yields:*

$$\begin{aligned}
&\eta_s = \frac{v\,(3\,\tau_t s + 4\,\sigma_t)}{4s^2}, \eta_t = \frac{v\,(17\,\tau_t s - 2\,\tau_{t,t}s + 12\,\sigma_t)}{8s}, \eta_v = \frac{\eta}{v},\\
&\tau_{t,t,t} = 0, \sigma_{t,t} = 0, \sigma_s = \frac{\tau_t s + 2\,\sigma}{2s}, \tau_s = 0, \sigma_v = 0, \tau_v = 0
\end{aligned} \tag{2.2.5}$$

Application of RIF*'s initial data algorithm yields:*

$$\begin{aligned}
&\eta(s_0,t_0,v_0) = c_1, \quad \tau(s_0,t_0,v_0) = c_2, \quad \tau_t(s_0,t_0,v_0) = c_3,\\
&\tau_{tt}(s_0,t_0,v_0) = c_4, \quad \sigma(s_0,t_0,v_0) = c_5, \quad \sigma_t(s_0,t_0,v_0) = c_6
\end{aligned} \tag{2.2.6}$$

The key aspect relevant for our paper is that (2.2.5) *and* (2.2.6) *are obtained with algorithmic operations and in particular without integration. In addition further algorithms from the* LAVF *package can determine the structure of its Lie Algebra. Indeed we find a two dimensional abelian subalgebra from that output, a necessary condition for the existence of a map of the Black-Schole's equation to a constant coefficient equation.*

2.2.1 Mapping Equations

Assuming existence of a local analytic invertible map $\Psi = (\psi, \phi)$ between the *Source* system R and the *Target* system $\hat{R}$ and applying it to the infinitesimals $(\hat{\xi}, \hat{\eta})$ yields what we will call the *Bluman-Kumei (BK) mapping equations*:

$$\begin{aligned}
&\sum_{i=1}^{n} \xi^i(x,u)\frac{\partial \psi^k}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x,u)\frac{\partial \psi^k}{\partial u^j} = \hat{\xi}^k(\hat{x},\hat{u})\\
&\sum_{i=1}^{n} \xi^i(x,u)\frac{\partial \phi^\ell}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x,u)\frac{\partial \phi^\ell}{\partial u^j} = \hat{\eta}^\ell(\hat{x},\hat{u})
\end{aligned} \tag{2.2.7}$$

where $1 \leq k \leq n$ and $1 \leq \ell \leq m$. See Bluman and Kumei [5, 8] for details and generalizations (e.g. to contact transformations). Note that all quantities on the LHS of the BK mapping equations (2.2.7) are functions of (x, u) including ϕ and ψ.

Example 2.2.3 *We informally illustrate the BK mapping equations on Example 2.2.1. Here we follow an approach based on heuristic integration of the symmetry defining system. Indeed the defining system is easily integrated to find the full 6 dimensional Lie symmetry algebra. And among the basis of symmetries the reader can easily find the two operators previously by inspection:*

$$\frac{\partial}{\partial t}, \quad s\frac{\partial}{\partial s} \tag{2.2.8}$$

which implies that mapping has form: $\hat{s} = \psi^1(s,t)$, $\hat{t} = \psi^2(s,t)$, $\hat{v} = \phi(s,t) = v$. When the corresponding coefficients of (2.2.8) are substituted into the BK system (2.2.7) we get:

$$\begin{aligned} s\frac{\partial}{\partial s}\psi^1(s,t) = 1, \quad s\frac{\partial}{\partial s}\psi^2(s,t) = 0 \\ \frac{\partial}{\partial t}\psi^1(s,t) = 0, \quad \frac{\partial}{\partial t}\psi^2(s,t) = 1 \end{aligned} \tag{2.2.9}$$

which yields by simple integration the same result as before for the mapping of the Black-Schole's to constant coefficient:

$$\hat{s} = \psi^1(s,t) = \log(s) + c_1, \quad \hat{t} = \psi^2(s,t) = t + c_2 \tag{2.2.10}$$

Indeed this integrating and breaking down into a basis, is the method used by Bluman and Kumei. However it does not yield an algorithm, since it depends on heuristic integration.

Finally we mention, that we are not opposed to integration, and in fact, a combination of integration and the algorithmic methods of this article, are probably a preferable way to proceed in practice.

Let S, $\hat{S}$ denote the symmetry defining systems for the *Source* system R and the *Target* system $\hat{R}$ respectively, with corresponding Lie symmetry algebras $\mathscr{L}$ and $\hat{\mathscr{L}}$. If an invertible map Ψ exists mapping R to $\hat{R}$ then it most generally depends on $\dim(\mathscr{L}) = \dim(\hat{\mathscr{L}})$ parameters. But we only need one such Ψ. So reducing the number of such parameters, e.g. by restricting to a Lie subalgebra $\mathscr{L}'$ of $\mathscr{L}$ with corresponding Lie subalgebra $\hat{\mathscr{L}}'$ of $\hat{\mathscr{L}}$ that still enables the existence of such Ψ, is important in reducing the computational difficulty of such methods. We will use the notation $S', \hat{S}'$ denote the symmetry defining systems of Lie sub-algebras $\mathscr{L}'$, $\hat{\mathscr{L}}'$ respectively. See [8, 27] discussion on this matter.

Example 2.2.4 *The mapping of the linear Black-Schole's equation (2.2.1) to a constant coefficient linear equation; we exploited the existence of a two dimensional abelian subalgebra. Indeed if we are lucky enough to identify this subalgebra immediately, then it gives very simple mapping equations with only two parameters. In the general algorithm for mapping linear equations to constant coefficient equations we described later, we can first bring such equations to homogeneous form. Restricting to symmetries, and mappings that retain the homogeneous form, can be imposed by restricting to symmetries with $\{\eta_v = \eta/v, \sigma_v = 0, \tau_v = 0\}$, which has a finite 6 parameter Lie group of symmetries. Thus we rejected the unhelpful infinite super-position subgroup as we did in the Black-Schole's example earlier. For more details see Bluman et al. [8].*

2.2.2 Algorithms EquivDetSys and DimEquivTest

With the *Source* system R, the *Target* system $\hat{R}$ and the mapping Ψ, the algorithm `MapEqs`$(R, \hat{R}, \Psi)$ is to return the full non-nonlinear defining equations for mappings from R to $\hat{R}$ which are invertible (i.e. Det Jac$(\Psi) \neq 0$). As preparation for the description of this algorithm we introduce the following algorithm.

EquivDetSys$(R, \hat{R})$: This is `Maple` implementation of returning the nonlinear DPS for invertible mappings Ψ from R to $\hat{R}$. Our implementation currently requires that $R, \hat{R}$ are in solved form for their leading derivatives with respect to a ranking graded by total differential order; though this could be weakened in the future. Then `Maple`'s general purpose routine for changing variables `dchange` is applied, yielding expressions in the parametric derivatives of R. Setting coefficients of independent powers of the parametric derivatives to zero, together with Det Jac$(\Psi) \neq 0$ simplified with respect to Ψ yields the nonlinear determining system for Ψ. This construction is well-known (indeed it is used in [21] in the special case of mappings linearizing ODE). However the nonlinear overdetermined systems are challenging to compute due to the expansion of determinants as the number of variables and differential order of $R, \hat{R}$ increase.

Our approach in this paper, is to take advantage of such linearized infinitesimal information, available from Lie symmetries and in particular via the BK equations, which are linear in the mapping variables. Then if necessary, at the end of `MapDE` apply `EquivDetSys`, which can be much simplified by the earlier computed conditions in Ψ.

Also we employ a number of efficient preliminary tests that can some times quickly determine if R and $\hat{R}$ are not equivalent via Ψ.

DimEquivTest$(R, \hat{R})$: Differential-elimination algorithms such as those in the packages `RIF`, `DifferentialAlgebra` and `DifferentialThomas` allow a determination of a coordinate dependent description of initial data, and using that the determination of the coordinate independent quantities dim(R), dim$(\hat{R})$. Thus a quick first test applied by `DimEquivTest` is dim(R) = dim$(\hat{R})$.

If the input ranking is graded first by total derivative order, then further dimension invariants can be derived from that initial data: which are the number of parametric derivatives at each derivative order n (determining the Differential Hilbert Series). `DimEquivTest` tests the equality of these invariants up to the maximum involutivity order for $R, \hat{R}$. One further invariant is the number of arbitrary functions of the maximum number of independent variables appearing in the initial data. For background information see [33].

2.3 MapDE Algorithm

In §2.3.1 we describe `MapDE` for a specific *Source* system R and specific *Target* system $\hat{R}$. In §2.3.2 we give a description of `MapDE` for a linear input equation and a class of *Target* systems (where the *Target* is constant coefficient linear equation).

2.3.1 The MapDE Agorithm for specific R and $\hat{R}$

The algorithm `MapDE`$(R, \hat{R}, \Psi)$ returns the system of mapping equations in RIF-form, and, if their integration is successful, an explicit form of the transformations to map the system to

the *TargetClass*. It is described in the `MapDE` Algorithm 3 provided next. In that algorithm we suppose that $\mathscr{L}, \hat{\mathscr{L}}$ are respectively the Lie algebras of symmetries of R, $\hat{R}$, with defining systems S, $\hat{S}$.

For mathematical properties of the algorithms, including finiteness, see the following references. For LAVF see [18], for RIF's existence and uniqueness theory see [32], for the classification of differential rankings see [31]. For the algorithmic determination of structure of transitive Lie pseudogroups see Lisle and Reid [20].

Algorithm 3 `MapDE`

`MapDE`($Source, Target, Map$)
Input:
Source: a DPS system R, $[x, u]$, $[\xi, \eta]$, Opt
Target: a DPS system $\hat{R}$, $[\hat{x}, \hat{u}]$, $[\hat{\xi}, \hat{\eta}]$, Opt
Map: Ψ, Opt
Output: $\emptyset$ if no consistent RIF-form cases computed, otherwise a consistent RIF-form case Q for Ψ and `pdsolve` (Q)

1: Compute
$R :=$ RIF(R), ID(R), dim(R)
$\hat{R} :=$ RIF$(\hat{R})$, ID$(\hat{R})$, dim$(\hat{R})$
2: **if** `DimEquivTest` $(R, \hat{R}) \neq$ true **then return** $\Psi = \emptyset$ **end if**
3: Compute
$S =$ RIF(DetSys(R)), $\hat{S} =$ RIF(DetSys$(\hat{R})$)
ID(S), ID$(\hat{S})$, dim(S), dim$(\hat{S})$
4: **if** `DimEquivTest` $(S, \hat{S}) \neq$ true **then return** $\Psi = \emptyset$ **end if**
5: Compute StrucCons$(\mathscr{L})$, StrucCons$(\hat{\mathscr{L}})$
where $d =$ dim$(\mathscr{L}) =$ dim$(\hat{\mathscr{L}})$.
6: **if** $\mathscr{L} \neq \hat{\mathscr{L}}$ **then return** $\Psi = \emptyset$ **end if**
7: Set $M_{\mathrm{BK}} =$(2.2.7) and obtain the mapping system:
$M := S \cup \hat{S}|_\Psi \cup M_{\mathrm{BK}}(\mathscr{L}, \hat{\mathscr{L}}) \cup \{\mathrm{DetJac}(\Psi) \neq 0\}$
where $\hat{S}|_\Psi$ is $\hat{S}$ evaluated in terms of (x, u) and (ξ, η, ψ, ϕ) as functions of (x, u) via Ψ.
8: Compute $M_{\mathrm{RIF}} :=$ RIF$(M, \prec, casesplit, mindim = d)$
9: **if** $M_{\mathrm{RIF}} = \emptyset$ **then return** $\Psi = \emptyset$ **end if**
10: **if** $\exists M_{\mathrm{RIF}}[\ell] \in M_{\mathrm{RIF}}$ with $d < \infty$ dimensional ID for Ψ
then return $Q := M_{\mathrm{RIF}}[\ell]$ and `pdsolve`(Q)
end if
11: Compute Sys(Ψ) := `EquivDetSys`$(R, \hat{R})$
Q := $\emptyset$
12: **while** Q = $\emptyset$ for each consistent sys $M_{\mathrm{RIF}}[k] \in M_{\mathrm{RIF}}$ **do**
$Q :=$ SelSys(RIF(Sys$(\Psi) \cup M_{\mathrm{RIF}}[k], casesplit, mindim = d)$)
end do
13: **if** $Q \neq \emptyset$ **return** Q and `pdsolve` (Q) **else return** $\Psi = \emptyset$ **end if**

Notes for the MapDE Algorithm

Input: The input *Source R* consists of differential polynomial system (DPS) of differential polynomials with coefficients in some computable field (e.g. $\mathbb{Q}$); Opts are additional Options such as input rankings if not default.

Output: `pdsolve` is `Maple` general purpose exact PDE solver: the application of `Maple`'s `pdsolve` which can not guarantee successful integration of DE.

Step 1: Here and throughout RIF and ID refer to `Maple`'s `DEtools` package commands `rifsimp` and `initialdata` commands. Alternatively one could use other Maple packages such as `diffalg` or `DifferentialThomas`.

Step 2: As introduced in §2.2.2, `DimEquivTest`$(R, \hat{R})$ is a simple algorithm for checking some necessary conditions for the existence of a mapping: the simplest being $\dim(R) = \dim(\hat{R})$, and include others corresponding to coefficients of the Differential Hilbert Series for R and $\hat{R}$.

Steps 3, 4: Restriction to a subalgebra is also possible and can improve efficiency. Similarly to Step 2, invariant dimension information can lead to early rejection of existence of a mapping: the first being that $\dim(S) = \dim(\hat{S})$.

Steps 5, 6: LAVF command `StructureCoefficients` algorithmically determines the structure constants of the algebras for $d < \infty$. `Maple`'s `LieAlgebras` and `DifferentialGeometry` packages, are then used to generate the polynomial system for $b_{i,j}$ in a change of basis matrix $B = [b_{i,j}]$ which is then analyzed by the solver `Triangularize`.

Step 7: The change of coordinates to compute $\hat{S}|_\Psi$ is accomplished by applying the Maple command `dchange` and using the transformation properties of Lie vector fields [8, 26].

Step 8: Differential elimination with casesplitting is applied and useless computations on branches with ID < mindim= d wrt $(\xi, \eta, \hat{\xi}, \hat{\eta})$ avoided. The ranking $\prec$ ranks the map variables Ψ less than any derivative of the infinitesimals $(\xi, \eta, \hat{\xi}, \hat{\eta})$ yielding an uncoupled system in Ψ whose ID is then examined and cases with less than d dimensional data rejected. This ranking means that the linearity in $(\xi, \eta, \hat{\xi}, \hat{\eta})$ is maintained in computations.

Step 11: See §2.2.2.

Step 12: `SelSys` (M) selects a consistent system from the output of RIF $(M$, casesplit, $mindim = d)$

2.3.2 MapDE for mapping Linear Homogeneous DE to Constant Coefficient Linear DE

Here we consider how to map a linear homogeneous source DE to a constant coefficient linear homogeneous DE, with an algorithm which results from straightforward changes to Algorithm 3.

The idea introduced in Bluman et al. [8, §2.5] for this problem is to introduce a chain of Lie subalgebras whose purpose is to focus on the *Target*: $\hat{\mathscr{L}} \supset \hat{\mathscr{L}}' \supset \hat{\mathscr{L}}^*$ and via Ψ also a chain $\mathscr{L} \supset \mathscr{L}' \supset \mathscr{L}^*$.

Now $\hat{R} = \sum_{i\in I} a_i K(i) = 0$ where $K = \{K(i) : i \in I\}$ is the set of derivatives of $\hat{u}$ of order $\leq$ differential order of $\hat{R}$. The unspecified constants a_i are the coefficients of the target. It is natural to restrict to transformations that preserve the linearity and homogeneity of the input DE and result from eliminating the superposition symmetry: $\hat{x} = f(x)$ and $\hat{u} = g(x)\,u$ where $x = (x^1, \ldots, x^n)$ (see Bluman et al. [8]). Correspondingly it is natural to consider a subalgebra $\mathscr{L}'$ that results by appending the equations $\xi^j_u = 0$, $j = 1, \ldots, n$ and $\eta_u = \eta/u$ to S to form S':

$$S' := \{\eta_u = \eta/u, \xi^j_u = 0 : j = 1, \ldots, n\} \cup S \tag{2.3.1}$$

and similarly for $\hat{\mathscr{L}}'$. To avoid the early calculations that involve $\hat{R}$, we focus like Bluman et al, on accessible infinitesimal information encoded in a Lie algebra $\hat{\mathscr{L}}^*$. In this case $\hat{\mathscr{L}}^*$ corresponds to n commuting translations in the independent variables $\hat{x}^1, \ldots, \hat{x}^n$, i.e. n translations with generators $\frac{\partial}{\partial \hat{x}^j}$. The corresponding differential system for $\hat{S}^*$ and $\hat{\mathscr{L}}^*$ is

$$\hat{S}^* = \{\hat{\eta} = 0, \hat{\xi}^j_{\hat{u}} = 0, \hat{\xi}^j_{\hat{x}^k} = 0 : 1 \leq j, k \leq n\} \tag{2.3.2}$$

Algorithm 4 `MapDE` with `TargetClass` =`ConstantCoeffDE`

`MapDE`(*Source*, *Target*, *Map*)
Input:
Source: A single linear homogeneous DPS R, $[x, u]$, $[\xi, \eta]$, Opt
Target: `TargetClass` =`ConstantCoeffDE` , Opt
Map: Ψ, Opt
Output: $\emptyset$ if no consistent RIF-form cases computed, otherwise
a consistent RIF-form case Q for Ψ and `pdsolve` (Q)

1: Compute
   $R :=$ RIF(R), ID(R), dim(R)
   $\hat{R} := \sum_{i\in I} a_i K(i) = 0$
2: Compute $S =$ RIF(S'), ID(S'), dim(S), dim$(\hat{S})$
3: Compute StrucCons$(\mathscr{L}')$
4: $M := S' \cup \hat{S}^*|_\Psi \cup M_{\mathrm{BK}}(\mathscr{L}', \hat{\mathscr{L}}^*) \cup \{\mathrm{DetJac}(\Psi) \neq 0\}$
5: Compute $M_{\mathrm{RIF}} :=$ RIF$(M, \prec, casesplit, mindim = n)$
6: **if** $M_{\mathrm{RIF}} = \emptyset$ **then return** $\Psi = \emptyset$ **end if**
7: **if** $\exists M_{\mathrm{RIF}}[\ell] \in M_{\mathrm{RIF}}$ with $d = n < \infty$ dimensional ID for Ψ
   then return $Q := M_{\mathrm{RIF}}[\ell]$ and `pdsolve`(Q)
end if

2.4 Examples

2.4.1 Equivalence

Example 2.4.1 *Bluman et al. [8, §2.3.2, pg 133-137] apply their mapping method based on explicit integrations to determine an invertible mapping by a point transformation of the cylin-*

drical KdV equation R to the KdV equation $\hat{R}$ that first appeared in the work of Korobeinikov [16]:

$$R := \left\{u_{x,x,x} = -uu_x - u_t - \frac{u}{2t}\right\} \tag{2.4.1}$$

$$\hat{R} := \{\hat{u}_{\hat{x},\hat{x},\hat{x}} = -\hat{u}\,\hat{u}_{\hat{x}} - \hat{u}_{\hat{t}}\} \tag{2.4.2}$$

They give details of their calculations and for illustration we apply the `MapDE` *algorithm 3 to the same example. Here we seek transformations* $\hat{x} = \psi(x,t,u), \hat{t} = \phi(x,t,u), \hat{u} = \Upsilon(x,t,u)$.

Steps 1, 2: Both R and $\hat{R}$ *are already in* RIF*-form with respect to any orderly ranking. The initial data for the R and* $\hat{R}$ *are*

$$\{u(x_0,t) = F_1(t), u_x(x_0,t) = F_2(t), u_{x,x}(x_0,t) = F_3(t)\}$$

$$\{\hat{u}(\hat{x}_0,\hat{t}) = F_1(\hat{t}), \hat{u}_{\hat{x}}(\hat{x}_0,\hat{t}) = F_2(\hat{t}), \hat{u}_{\hat{x},\hat{x}}(\hat{x}_0,\hat{t}) = F_3(\hat{t})\}$$

Here there are arbitrary functions in the initial data, so $\dim R = \dim \hat{R} = \infty$. *Their Hilbert Series obviously are equal, and up to the order of involutivity:* $HS(s) = 1 + 2s + 3s^2 + 0(s^3)$ *where the coefficient of* s^n *is the number of parametric derivatives of order n. So* `DimEquivTest`$(R,\hat{R})$ = *true in Step 2.*

Step 3, 4: The RIF*-form systems* $S, \hat{S}$ *are*

$$\begin{aligned} S = [&\eta_{u,u} = 0, \xi_x = -\frac{1}{2}\eta_u, \beta_x = 0, \eta_x = \frac{-3\,t\eta_u - 2\beta}{4t^2}, \beta_t = -\frac{3}{2}\eta_u, \\ &\eta_t = \frac{4\,t\eta_u u + 2\beta\,u - \eta\,t}{2t^2}, \xi_u = 0, \beta_u = 0, \xi_t = -\eta_u u + \eta] \end{aligned} \tag{2.4.3}$$

$$\begin{aligned} \hat{S} = [&\hat{\eta}_{\hat{u},\hat{u}} = 0, \hat{\xi}_{\hat{x}} = -\frac{1}{2}\hat{\eta}_{\hat{u}}, \hat{\beta}_{\hat{x}} = 0, \hat{\eta}_{\hat{x}} = 0, \\ &\hat{\xi}_{\hat{t}} = -\hat{\eta}_{\hat{u}}\hat{u} + \hat{\eta}, \hat{\beta}_{\hat{t}} = -3/2\,\hat{\eta}_{\hat{u}}, \hat{\eta}_{\hat{t}} = 0, \hat{\xi}_{\hat{u}} = 0, \hat{\beta}_{\hat{u}} = 0] \end{aligned} \tag{2.4.4}$$

and yield ID giving $\dim(S) = \dim(\hat{S}) = 5$. *Also* `DimEquivTest`$(S,\hat{S})$ = *true in Step 4.*

Step 5: Here `MapDE` *uses the* LAVF *command* `StructureConstants` *to compute the structure of the* 4 *dimensional Lie algebras for R and* $\hat{R}$ *obtaining:*

$$\begin{aligned} \mathscr{L} :[&[Y_1, Y_4] = -1/2\,Y_1, [Y_2, Y_3] = Y_1, [Y_2, Y_4] = -3/2\,Y_1 + Y_2, \\ &[Y_3, Y_4] = -3/2\,Y_3, [Y_1, Y_3] = 0, [Y_1, Y_2] = 0] \end{aligned} \tag{2.4.5}$$

$$\begin{aligned} \hat{\mathscr{L}} :[&[\hat{Y}_1, \hat{Y}_4] = -1/2\,\hat{Y}_1, [\hat{Y}_2, \hat{Y}_3] = \hat{Y}_1, [\hat{Y}_2, \hat{Y}_4] = -3/2\,\hat{Y}_2, \\ &[\hat{Y}_3, \hat{Y}_4] = \hat{Y}_3, [\hat{Y}_1, \hat{Y}_2] = 0, [\hat{Y}_1, \hat{Y}_3] = 0] \end{aligned} \tag{2.4.6}$$

Steps 5, 6: We obtain $\mathscr{L} \simeq \hat{\mathscr{L}}$ *and the explicit isomorphism:*

$$\hat{Y}_1 = Y_1, \quad \hat{Y}_2 = Y_3, \quad \hat{Y}_3 = Y_1 - Y_2, \quad \hat{Y}_4 = Y_4 \tag{2.4.7}$$

Bluman et al. [8, Eqs (2.39), (2.40), pg 134]obtain the structure and an isomorphism by explicitly integrating the defining systems, whereas we avoid this. This isomorphism is a necessary but not sufficient condition for the existence of a local analytic invertible map to $\hat{R}$.

Steps 7, 8: *The* RIF*-form of the mapping system results in one consistent case with $d = 4$-dim ID in the infinitesimals for $S, \hat{S}$. The* RIF*-form of the Ψ system is:*

$$[\phi_{t,t} = -3/2\,\frac{\phi_t}{t}, \Upsilon_{u,u} = 0, \Upsilon_x = -1/2\,\frac{\Upsilon_u}{t}, \psi_x = \phi_t \Upsilon_u,$$
$$\phi_x = 0, \Upsilon_t = \frac{\Upsilon_u u}{t}, \psi_t = -u\Upsilon_u \phi_t + \Upsilon\,\phi_t, \psi_u = 0, \phi_u = 0] \tag{2.4.8}$$

Steps 9, 10: *Step 9 does not apply. The above* RIF*-form for the Ψ system has ID for $z_0 = (x_0, t_0, u_0)$:*

$$\Upsilon(z_0) = c_1, \Upsilon_u(z_0) = c_2, \phi(z_0) = c_3, \phi_t(z_0) = c_4, \psi(z_0) = c_5$$

Geometrically, since the ID has dimension $5 > d = 4$, there is a class of systems with the same d dimensional invariance group, that possibly includes the Target system. So Step 10 does not apply. Thus we have to apply `EquivDetSys` *to find missing condition(s). After explicit integration Bluman et al also find that they don't uniquely specify the target, and essentially they substitute the transformations to obtain the parameter values to specify the target.*

Steps 11, 12: *Applying* RIF *to the combined system* {`EquivDetSys`$(R, \hat{R})$, (2.4.8)} *yields a single case:*

$$M_{\text{RIF}} = [\psi_{t,t} = -3/2\,\frac{\psi_t}{t}, \Upsilon_{u,u} = 0, \Upsilon_x = -1/2\,\frac{\Upsilon_u}{t}, \phi_x = 0,$$
$$\psi_x = \frac{\Upsilon_u \psi_t}{-u\Upsilon_u + \Upsilon}, \Upsilon_t = \frac{u\Upsilon_u}{t}, \phi_t = \frac{\psi_t}{-u\Upsilon_u + \Upsilon}, \phi_u = 0, \psi_u = 0]$$

where the constraint is $-{\psi_t}^2 {\Upsilon_u}^3 + {\Upsilon_u}^2 u^2 - 2\,\Upsilon\, u \Upsilon_u + \Upsilon^2 = 0$, and the inequation $\frac{{\psi_t}^2 {\Upsilon_u}^2}{(-u\Upsilon_u + \Upsilon)^2} \neq 0$. The ID shows we now have $4 = d$ parameters, confirming the existence of the transformations, without integration.

Step 13 *Applying* `pdsolve` *yields the solution for the transformation below:*

$$\{\hat{x} = \frac{c_3\, x}{\sqrt{t}} - 2\,\frac{c_3\, c_2}{\sqrt{t} c_1} + c_4, \quad \hat{t} = -2\,\frac{c_3}{\sqrt{t} c_1} + c_5, \quad \hat{u} = 1/2\,(2\,tu - x)\,c_1 + c_2\}$$

where

$$\left(-1/4\,{c_1}^3 {c_3}^2 + 1/4\,{c_1}^2\right) x^2 + \left({c_1}^2 c_2\, {c_3}^2 - c_1\, c_2\right) x - c_1\, {c_2}^2 {c_3}^2 + {c_2}^2 = 0.$$

subject to the determinental condition. The last condition implies $c_1 c_3^2 = 1$. Specializing the values of the c_j give the transformations obtained also in Bluman et al.

2.4.2 Mapping to constant coefficient DE

Example 2.4.2 *The harmonic-oscillator Schrödinger Equation is $i\hbar\varphi_t = -\frac{\hbar^2}{2m}\varphi_{xx} + \frac{1}{2}m\omega^2 x^2 \varphi$ which in normalized* RIF*-form becomes:*

$$R := \{u_{x,x} = -x^2 u + u_t\} \tag{2.4.9}$$

Applying the `MapDE` *algorithm using the option* `Target = ConstantCoeffDE` *shows that (2.4.9) maps to a constant coefficient linear DE:*

$$\hat{R} := \{a_1\hat{u}_{\hat{x},\hat{x}} + a_2\,\hat{u}_{\hat{x},\hat{t}} + a_3\hat{u}_{\hat{t},\hat{t}} + a_4\hat{u}_{\hat{x}} + a_5\hat{u}_{\hat{t}} + a_6\hat{u} = 0\} \tag{2.4.10}$$

Existence of such a mapping is given algorithmically and the output includes the system for Ψ *with* $\dim(\Psi) = 6$*:*

$$\begin{aligned}\Psi = [&\psi_u = 0, \Upsilon_u = 0, \Upsilon_{x,x} = 0, \phi_u = \frac{\phi}{u}, \psi_x = \Upsilon_x, \psi_t = -{\Upsilon_x}^2 + \Upsilon_t,\\ &\phi_x = -\frac{\phi\left(-{\Upsilon_x}^2 + \Upsilon_t\right)}{2\Upsilon_x}, \Upsilon_{t,t,t} = -\frac{12\,\Upsilon_{t,t}x\Upsilon_x - 16\,{\Upsilon_t}^2 - 3\,{\Upsilon_{t,t}}^2}{2\Upsilon_t},\\ &\Upsilon_{x,t} = -\frac{\Upsilon_x\left(4\,x\Upsilon_x - \Upsilon_{t,t}\right)}{2\Upsilon_t},\\ &\phi_t = \frac{\phi\left(4\,{\Upsilon_x}^3x - 4\,{\Upsilon_x}^2\Upsilon_t x^2 - {\Upsilon_x}^4\Upsilon_t + 2\,{\Upsilon_x}^2{\Upsilon_t}^2 - {\Upsilon_t}^3 - {\Upsilon_x}^2\Upsilon_{t,t}\right)}{4{\Upsilon_x}^2\Upsilon_t}]\end{aligned}$$

Integrating the system and specializing the 6 *constants gives:*

$$\begin{aligned}\hat{t} = \psi(x,t,u) &= \frac{2\,x + \cos(2\,t)}{\sin(2\,t) - \cos(2\,t)}\\ \hat{x} = \Upsilon(x,t,u) &= \frac{2\,x + 3\,\cos(2\,t)}{\sin(2\,t) - \cos(2\,t)}\\ \hat{u} = \phi(x,t,u) &= u(x,t)\sqrt{\sin(2\,t) - \cos(2\,t)}\exp(h(x,t))\\ h(x,t) &= \frac{4\,(\sin(2t) + \cos(2\,t))\,x^2 + 12\,x + 9\,\cos(2\,t)}{8\,(\sin(2t) - \cos(2\,t))}\end{aligned}$$

where $\hat{R}$ *is* $\hat{u}_{\hat{x},\hat{x}} + 2\,\hat{u}_{\hat{x},\hat{t}} + \hat{u}_{\hat{t},\hat{t}} - \hat{u}_{\hat{t}} = 0$*.*

When the mapping system $M := S' \cup \hat{S}^*|_\Psi \cup M_{BK}(\mathscr{L}', \hat{\mathscr{L}}^*) \cup \{DetJac(\Psi) \neq 0\}$ *is reduced to* RIF*-form as in Step 4, of Algorithm 4 it yields a consistent system for* Ψ *with* 10 *arbitrary constants in its initial data, hence establishing existence of a mapping to a constant coefficient DE. Since* $\dim(\mathscr{L}) = 6$ *this means that there is a* 4 *dimensional target class of constant coefficient linear DE. Two possible approaches to proceed are now described. The first is to attempt to (heuristically) integrate this system; and this successfully yielded the solution of the system with 10 parameters. The second approach is to further simplify the problem using algorithmic differential elimination before integrating. In particular we inserted an optional step in the algorithm, that executes a general change of coordinates on the TargetDE which has* 5 *arbitrary constants. Applying differential-elimination ranking those constants highest in the ordering yielded the subsystem:*

$$\begin{aligned}&a_2 = 2\,\frac{\Upsilon_x}{\psi_x},\ a_3 = \frac{{\Upsilon_x}^2}{{\psi_x}^2},\ a_4 = \frac{-\phi\,\psi_t - 2\,\psi_x\phi_x}{{\psi_x}^2\phi},\ a_5 = -\frac{\phi\,\Upsilon_t + 2\,\Upsilon_x\phi_x}{{\psi_x}^2\phi},\\ &a_6 = 1/4\,\frac{4\,\phi^2x^2\Upsilon_t - 4\,\phi^2x\Upsilon_x + \phi^2\Upsilon_{t,t} + 4\,\phi\,\Upsilon_t\phi_t + 4\,\Upsilon_t{\phi_x}^2}{\Upsilon_t{\psi_x}^2\phi^2}\end{aligned}$$

The system M_{RIF} is updated by appending the above equations for the constants, which amount to integrations, that reduce the dimension of the system for Ψ by specializing the values of the constants. This yielded 6 dimensional system for Ψ. Note that it was reduced by 4 dimensions (after eliminating the relation $a_2^2 = 4a_3$).

Example 2.4.3 *Consider the DE arises in financial models known as Black Schole's[23] as the Source system R,*

$$v_t + \frac{s^2 v_{s,s}}{2} + s v_s - v = 0$$

Using our algorithm `MapDE` *with* `TargetClass = ConstantCoeffDE`*, automatically yields the mapping Ψ:*

$$\hat{s} = \ln(s)\,,\ \hat{t} = t,\ \hat{v} = v$$

and the Target system $\hat{R}$ is $\hat{v}_{\hat{t}} + 1/2\,\hat{v}_{\hat{s},\hat{s}} - \hat{v} = 0$.

2.5 Discussion

Mappings of mathematical models are a fundamental tool of mathematics and its applications. This fact and the notorious difficulty of their computation motivates us to explore the approach we presented in this article. This resulted in our algorithm, `MapDE`, which is given algorithmic realization for two cases, involving point mappings. The first is where the input systems R and $\hat{R}$ are specified as polynomially nonlinear DEs and `MapDE` returns a reduced involutive RIF-form for the mapping equations. The second is where R is a linear homogeneous DE and the *TargetClass* is a constant coefficient linear homogeneous DE (`Target = ConstantCoeffDE`). Our approach exploits the linearity of the Bluman-Kumei mapping equations that arise in the presence of symmetry, and postpone, simplify and even avoid direct computations with the full nonlinear determining equations for the mappings. We also implement some fast preliminary tests for equivalence under mappings.

The works of Anco, Bluman and Wolf [1] and also Wolf [35], consider a computer program for computing linearization mappings. It exploits Wolf's program ConLaw's strong facilities for integrating systems of PDE exactly in addition to the BK mapping equations, as well as an embedding technique involving multipliers and conservation laws. They also mention that the problem of full algorithmization using differential algebra as an important open problem. In another paper [24], we give various extensions to `MapDE`. One of these involves extending it to determining existence of exact linearization mappings of DE. In so doing we provide algorithm and combining aspects of the approach of Bluman, Anco and Wolf [1, 35] and also of Gerdt et al [21]. Building in invariant properties into the completion process is also a possibility; borrowing aspects of the more geometrical approaches. Also we plan to extend our methods beyond point mappings to also include contact and nonlocal transformations. An important open problem is the characterization of equivalence classes of linear PDE. Maximally invariant linear PDE, constitute one class, as do minimally invariant PDE. Our work gives clues to such characterizations, as does the more complete results known for ODE.

On the longer term we are particularly interested in exploring approximate mappings and approximate equivalence. Indeed LAVF already has the first available algorithm for determining the structure of approximate symmetry of DE theoretically first described in Lisle, Huang and

Reid [19, 29]. For example, consider Poisson's equation for a gravitational potential $u(x, y, z)$: $\nabla^2 u = f(x, y, z)$ and an interstellar gas with density proportional to $f(x, y, z) = \frac{1}{2}(G(x, y, z-a) + G(x, y, z+a))$ where $G(x, y, z) = \exp(-x^2 - y^2 - z^2)$ and $a = 10^{-3}$:

$$u_{xx} + u_{yy} + u_{zz} = f(x, y, z) = \frac{1}{2}\left(G(x, y, z + a) + G(x, y, z - a)\right)$$

Applying Lie's standard method where

$$L = \xi(x, y, z, u)\frac{\partial}{\partial x} + \eta(x, y, z, u)\frac{\partial}{\partial y} + \psi(x, y, z, u)\frac{\partial}{\partial z} + \phi(x, y, z, u)\frac{\partial}{\partial u},$$

discarding the superposition via $\phi_u = \phi/u$ and performing an exact symmetry analysis yields only a 1 dimensional rotation group about the z axis; throughout all space no matter how small a is. However as $a \to 0$ or as we move further away from $(x, y, z) = (0, 0, 0)$ we expect additional approximate symmetries. Symbolic (exact) approaches to finding such approximate symmetries, based on identifying a small parameter in the DE, and using perturbation series expansion of the solution and symmetry, are given by [4, 13]. Instead we apply the method of Lisle, Huang and Reid [19] that uses stable tools of numerical linear algebra to find the approximate Lie algebra of symmetries of Poisson's Equation near a point (x_0, y_0, z_0). Choosing $(x_0, y_0, z_0) = (0, 3.2, 0)$ and $a = 10^{-3}$ we find:

$$\begin{aligned}
[L_1, L_2] &= -1.182 \times 10^{-13}\, L_1 - 2.724 \times 10^{-9}\, L_2 - 0.707\, L_3 \\
[L_1, L_3] &= -1.446 \times 10^{-7}\, L_1 + 0.236\, L_2 + 2.724 \times 10^{-9}\, L_3 \\
[L_2, L_3] &= -0.707\, L_1 + 1.446 \times 10^{-7}\, L_2 - 6.042 \times 10^{-14}\, L_3
\end{aligned}$$

which we can recognize as

$$[L_1, L_2] = -\frac{1}{\sqrt{2}}L_3, \quad [L_1, L_3] = \frac{1}{3\sqrt{2}}L_2, \quad [L_2, L_3] = -\frac{1}{\sqrt{2}}L_1$$

or after the basis change $L_1 = \frac{Y_1}{\sqrt{6}}, L_2 = -\frac{Y_2}{\sqrt{2}}, L_3 = \frac{Y_3}{\sqrt{6}}$ is $so(3)$:

$$[Y_1, Y_2] = Y_3, \quad [Y_2, Y_3] = Y_1, \quad [Y_3, Y_1] = Y_2$$

Choosing a grid of points (x_0, y_0, z_0) we get different regions with different approximate Lie algebras, plus transition bands. Potentially and intuitively the model can be mapped to various forms depending on the region, a topic for future research.

Acknowledgements

GJR acknowledges the support of an NSERC Discovery Grant from the government of Canada. GJR acknowledge the contribution of his colleague and departed friend Ian Lisle, whose inspiration lies behind this work in its spirit and many details. We thank one of the referees, in particular, for their helpful comments.

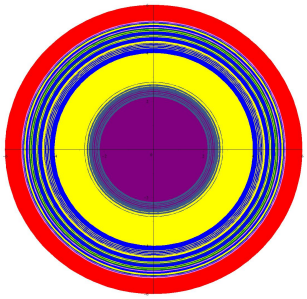


Figure 2.1: Regions of approximate symmetry in the $x - y$ plane for $\nabla^2 u = f(x, y, z)$. Purple Region: $\dim \mathscr{L} = 1$, $\mathscr{L} \approx$so(2); Yellow Region $\dim \mathscr{L} = 3$,$\mathscr{L} \approx$so(3); Red Region $\dim \mathscr{L} = 11$. Between the red, yellow and purple regions are transition bands in which the approximate Lie algebra was not stably computed.

Bibliography

[1] S. Anco, G.W. Bluman, and T. Wolf. Invertible Mappings of Nonlinear PDEs to Linear PDEs Through Admitted Conservation Laws. *Acta Applicandae Mathematica*, 101, 21–38, 2008. 35, 43, 64, 86, 94

[2] I.M. Anderson and C.G. Torre. Newsymbolic tools for differential geometry, gravitation, and field theory. *J. Math. Phys*, 53(1), Article 013511, 2012. https://doi.org/10.1063/1.3676296 24, 42, 93

[3] O. Arnaldsson. Involutive Moving Frames. Ph.D. Dissertation. Department of Mathematics, University of Minnesota, 2017. 24, 42, 65

[4] V.A. Baikov, R.K. Gazizov, and N.H. Ibragimov. Approximate symmetries. Mathematics of the USSR, Sbornik, 64, 427–441, 1989. 36

[5] G.W. Bluman and S. Kumei. Symmetries and Differential Equations, Springer, 1989. 27, 50, 63

[6] G.W. Bluman and S. Kumei. Symmetry based algorithms to relate partial differential equations: I. Local symmetries. Eur. J. Appl. Math. 1, 189–216, 1990. 24, 46

[7] G.W. Bluman and S. Kumei. Symmetry based algorithms to relate partial differential equations: II. Linearization by nonlocal symmetries. Eur. J. Appl. Math. 1, 217–223, 1990. 24, 46, 52

[8] G.W. Bluman, A.F. Cheviakov, and S.C. Anco. Applications of Symmetry Methods to Partial Differential Equations, Springer, 2010. 1, 6, 8, 11, 18, 23, 24, 27, 30, 31, 32, 50, 51, 62, 63, 64, 93, 94, 95

[9] J. Bonasia, F. Lemaire, G.J. Reid, R. Scott, and L. Zhi. Determination of approximate symmetries of differential equations. In Group Theory and Numerical Analysis (CRM Proceedings and Lecture Notes), P. Winternitz et al. (Eds.), Vol. 39. AMS/CRM, 233–249, 2005. 23

[10] J. Carminati and K. Vu. Symbolic computation and differential equations: Lie symmetries. Journal of Symbolic Computation 29, 95–116, 2000. 6, 24, 46, 87

[11] A.F. Cheviakov. GeM software package for computation of symmetries and conservation laws of differential equations. Computer Physics Communications 176(1), 48–61, 2007. https://doi.org/10.1016/j.cpc.2006.08.001 6, 8, 24, 46, 87

[12] M. Fels and P. Olver. Moving Coframes II. Regularization and theoretical foundations. Acta Applicandae Mathematicae 55, 127–208, 1999. 23, 42, 65

[13] W.I. Fushchich and W.M. Shtelen. On approximate symmetry and approximate solutions of the non-linear wave equation with a small parameter. Journal of Physics A 22, L887–L890, 1989. 36

[14] O. Golubitsky, M. Kondratieva, A. Ovchinnikov, and A. Szanto. A bound for orders in differential Nullstellensatz. Journal of Algebra 322(11), 3852–3877, 2009. https://doi.org/10.1016/j.jalgebra.2009.05.032 Computational Algebra. 24, 42

[15] E. Hubert. Differential invariants of a Lie group action: Syzygies on a generating set. Journal of Symbolic Computation 44(4), 382–416, 2009. 24, 42, 65

[16] V.P. Korobeinikov. Certain types of solutions of Korteweg-de Vries-Burgers equations for plane, cylindrical, and spherical waves. Proceedings IUTAM Symposium on Nonlinear Wave Deformations,Tallinn, in Russian, 1982. 32

[17] S. Kumei and G.W. Bluman. When nonlinear differential equations are equivalent to linear differential equations. SIAM Journal of Applied Mathematics 42, 11571173, 1982. 11, 23, 24, 43

[18] I.G. Lisle and S.-L.T. Huang. Algorithms calculus for Lie Determining Systems. Journal of Symbolic Computation, 482–498, 2017. 23, 24, 29, 43, 47, 51, 94

[19] I.G. Lisle, S.-L.T. Huang, and G.J. Reid. Structure of Symmetry of PDE: Exploiting Partially Integrated Systems. Proceedings of the 2014 Symposium on Symbolic-Numeric Computation, 61–69, 2014. 23, 36, 81, 87, 88

[20] I.G. Lisle and G.J. Reid. Geometry and Structure of Lie Pseudogroups from Infinitesimal Defining Systems. J. Symb. Comput. 26(3), 355–379, 1998. https://doi.org/10.1006/jsco.1998.0218 29, 43, 47, 51

[21] D.A. Lyakhov, V.P. Gerdt, and D.L. Michels. Algorithmic Verification of Linearizability for Ordinary Differential Equations. In Proc. ISSAC 17. ACM, 285–292, 2017. 11, 23, 28, 35, 43, 48, 49, 52, 56, 57, 60, 61, 63, 64, 93, 94

[22] E.L. Mansfield. A Practical Guide to the Invariant Calculus. Cambridge Univ. Press. 2010. 24, 42, 65

[23] T. Masebe and J. Manale. New symmetries of Black-Scholes equation. In Proc. AMCM 13. 221–231, 2013. 1, 25, 35

[24] Z. Mohammadi, G. Reid, and S.-L.T. Huang. Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDS to linear PDS. Technical Report arXiv:1903.03727v1 [math.AP]. ArXiV. 2019. 35, 83, 94

[25] S. Neut, M. Petitot, and R. Dridi. Élie Cartan's geometrical vision or how to avoid expression swell. Journal of Symbolic Computation 44(3), 261–270, 2009. https://doi.org/10.1016/j.jsc.2007.04.006 Polynomial System Solving in honor of Daniel Lazard. 23, 42

[26] P.J. Olver. Application of Lie groups to differential equations (2nd ed.). Springer-Verlag, 1993. 4, 25, 30, 44, 73, 86

[27] P.J. Olver. Equivalence, invariance, and symmetry. Cambridge University Press, 1995. 2, 3, 4, 9, 23, 27, 42, 48, 50, 52, 65, 93

[28] G.J. Reid. Finding abstract Lie symmetry algebras of differential equations without integrating determining equations. European Journal of Applied Mathematics 2, 319–340, 1991. 11, 17, 23, 43

[29] G.J. Reid. Symbolic and Numeric Algorithms for Approximate Symmetry of PDE. https://doi.org/10.13140/RG.2.2.11483.41761 Workshop Talk at CASC 2017. 36

[30] T.M. Rocha Filho and A. Figueiredo. SADE: A Maple package for the symmetry analysis of differential equations. Computer Physics Communications 182(2), 467–476, 2011. https://doi.org/10.1016/j.cpc.2010.09.021 6, 24, 46, 87

[31] C.J. Rust and G.J. Reid. Rankings of partial derivatives. In Proc. ISSAC 97. 9–16, 1997. 13, 29, 44

[32] C.J. Rust, G.J. Reid, and A.D. Wittkopf. Existence and uniqueness theorems for formal power series solutions of analytic differential systems. In Proc. ISSAC99. ACM, 105–112, 1999. 12, 13, 29, 42, 44, 49, 57

[33] W.M. Seiler. Involution: The formal theory of differential equations and its applications in computer algebra. Algorithms and Computation in Mathematics, Vol. 24, Springer, 2010. https://doi.org/10.1007/978-3-642-01287-7 13, 15, 25, 28, 42, 44, 73, 75, 81

[34] F. Valiquette. Solving local equivalence problems with the equivariant moving frame method. SIGMA: Symmetry Integrability Geom. Methods Appl. 9, 2013. https://doi.org/10.3842/SIGMA.2013.029 23, 42

[35] T. Wolf. Investigating differential equations with CRACK, LiePDE, applsymm and ConLaw. Handbook of Computer Algebra, Foundations, Applications, Systems, 37, 465–468, 2002. 6, 7, 8, 35, 43, 65, 87

Chapter 3

Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE

This paper is a sequel to our previous work where we introduced the MapDE algorithm to determine the existence of analytic invertible mappings of an input (source) differential polynomial system (DPS) to a specific target DPS, and sometimes by heuristic integration an explicit form of the mapping. A particular feature was to exploit the Lie symmetry invariance algebra of the source without integrating its equations, to facilitate MapDE, making algorithmic an approach initiated by Bluman and Kumei. In applications, however, the explicit form of a target DPS is not available, and a more important question is, can the source be mapped to a more tractable class?

We extend MapDE to determine if a source nonlinear DPS can be mapped to a linear differential system. MapDE applies differential-elimination completion algorithms to the various over-determined DPS by applying a finite number of differentiations and eliminations to complete them to a form for which an existence-uniqueness theorem is available, enabling the existence of the linearization to be determined among other applications. The methods combine aspects of the Bluman-Kumei mapping approach with techniques introduced by Lyakhov, Gerdt and Michels for the determination of exact linearizations of ODE. The Bluman-Kumei approach for PDE focuses on the fact that such linearizable systems must admit a usually infinite Lie sub-pseudogroup corresponding to the linear superposition of solutions in the target. In contrast, Lyakhov et al. focus on ODE and properties of the so-called derived sub-algebra of the (finite) dimensional Lie algebra of symmetries of the ODE. Examples are given to illustrate the approach, and a heuristic integration method sometimes gives explicit forms of the maps. We also illustrate the powerful maximal symmetry groups facility as a natural tool to be used in conjunction with MapDE.

3.1 Introduction

This paper is a sequel to [29] and is part of a series in which we explore algorithmic aspects of exact and approximate mappings of differential equations. We are interested in mapping less

tractable differential equations into more tractable ones, in particular in this article focusing on mapping nonlinear systems to linear systems. This builds on progress in [29] where we considered mappings from a specific differential system to a specific target system and mappings from a linear to a linear constant coefficient differential equation.

As in [29] we consider systems of (partial or ordinary) differential equations with n independent variables and m dependent variables which are local analytic functions of their arguments. Suppose R has independent variables $x = (x^1, \ldots, x^n)$ and dependent variables $u = (u^1, \ldots, u^m)$ and $\hat{R}$ has independent variables $\hat{x} = (\hat{x}^1, \ldots, \hat{x}^n)$ and dependent variables $\hat{u} = (\hat{u}^1, \ldots, \hat{u}^m)$. In particular, we consider local analytic mappings Ψ: $(\hat{x}, \hat{u}) = \Psi(x, u) = (\psi(x, u), \phi(x, u))$ so that R is locally and invertibly mapped to $\hat{R}$:

$$\hat{x}^j = \psi^j(x, u), \qquad \hat{u}^k = \phi^k(x, u) \tag{3.1.1}$$

where $j = 1, \ldots, n$ and $k = 1, \ldots, m$. The mapping is locally invertible, so the determinant of the Jacobian of the mapping is nonzero:

$$\mathrm{DetJac}(\Psi) = \mathrm{Det}\frac{\partial(\psi, \phi)}{\partial(x, u)} \neq 0, \tag{3.1.2}$$

where $\frac{\partial(\psi,\phi)}{\partial(x,u)}$ is the usual Jacobian $(n + m) \times (n + m)$ matrix of first order derivatives of the $(n + m)$ functions (ψ, ϕ) with respect to the $(n + m)$ variables (x, u). Note throughout this paper: we will call $\hat{R}$ the *Target* system of the mapping, which will generally have some more desirable features than R, which we call the *Source* system. A well-known direct approach to forming the equations satisfied by Ψ is roughly to substitute the general change of variables into $\hat{R}$, evaluate the result modulo R, appending equations that express the independence of Ψ on derivative jet variables (or equivalently decomposing in independent expressions in the jet variables). The resulting equations for Ψ are generally nonlinear overdetermined systems. Algorithmic manipulation of these and other over-determined systems of PDE are at the core of the algorithms used in this paper. We make prolific use of differential-elimination completion (DEC) algorithms, which apply a finite number of differentiations and eliminations to complete such over-determined systems to a form including their integrability conditions, for which an existence uniqueness theorem is available. `Maple` is fortunate to have several such differential elimination packages. Currently we use the RIF algorithm via the `Maple` command `rifsimp` [40] in our implementation, but other `Maple` packages could be used [37, 10]. To be algorithmic we restrict to systems R and $\hat{R}$ that are polynomially nonlinear (i.e. differential polynomial systems, DPS). In this paper DEC refers to a Differential Elimination Completion algorithm to emphasize that a number of algorithms are available.

A very general approach to such problems, concerning maps Ψ from R to $\hat{R}$, is Cartan's famous Method of Equivalence which finds invariants that label the classes of systems equivalent under the pseudogroup of such mappings. See especially texts [32] and [27]. The fundamental importance and computational difficulty of such equivalence questions has attracted attention from the symbolic computation community [30]. For recent developments and extensions of Cartan's moving frames for equivalence problems see [14], [43] and [3]. The `DifferentialGeometry` package [2] is available in `Maple` and has been applied to equivalence problems [18]. Underlying these calculations is that overdetermined PDE systems with some non-linearity must be reduced to forms that enable the statement of a local existence and uniqueness theorem [17, 15, 41, 40, 9].

Our methods here and in [29] are based on the mapping approach initiated by Bluman and Kumei [19] which focuses on the interaction between such mappings and Lie symmetries via their infinitesimal form on the source and target. In particular, let $\mathscr{G}$ be the Lie group of transformations leaving R invariant. Also, let $\hat{\mathscr{G}}$ be the Lie group of transformations leaving $\hat{R}$ invariant. Locally, such Lie groups are characterized by their linearizations in a neighborhood of their identity, that is by their Lie algebras $\mathscr{L}$, $\hat{\mathscr{L}}$. If an invertible map Ψ exists then $\mathscr{G} \simeq \hat{\mathscr{G}}$ and $\mathscr{L} \simeq \hat{\mathscr{L}}$. This yields a subsystem of linear equations for Ψ which we call the Bluman-Kumei equations. It is a significant challenge to translate the methods of Bluman and Kumei into procedures that are algorithmic (i.e., guaranteed to succeed on a defined class of inputs in finitely many steps). Please see [1, 7, 44, 45] for progress in their approach and some (heuristic) integration-based computer implemented methods. Our methods are also inspired by remarkable recent progress on this question for ODE by Lyakhov et al.[25] who presented an algorithm for determining when an ODE is linearizable. It was also stimulated by their use of an early method by one of us (see Reid [33]), which has been dramatically improved and extended [34, 23] with the latest improvements in the LAVF package [22, 16].

In our previous work [29] we provided an algorithm to determine the existence of a mapping of a linear differential equation to the class of constant coefficient linear homogeneous differential equations. Key for this application was the exploitation of a commutative sub-algebra of symmetries of $\hat{\mathscr{L}}$ corresponding to translations of the independent variables in the target. The main contribution of this paper is to present an algorithmic method for determining the mapping of a nonlinear system to a linear system when it exists. Using a technique of Bluman and Kumei, we exploit the fact that $\hat{R}$ must admit a sub-pseudo group corresponding to the superposition property that linear systems by definition must satisfy. Once existence is established, a second stage can determine features of the map and sometimes by integration, explicit forms of the mapping. For an algorithmic treatment using differential elimination (differential algebra), we limit our treatment to systems of differential polynomials, with coefficients from $\mathbb{Q}$ or some computable extension of $\mathbb{Q}$ in $\mathbb{C}$. Thus our input system R should be a system of DPS. Some non-polynomial systems can be converted to differential polynomial form by using the Maple command, `dpolyform`.

In §3.2 we provide some introductory material on differential-elimination algorithms, initial data and Hilbert dimensions. In §3.3 we give an introduction to symmetries and mapping equations. In §3.4, we introduce the `MapDE` algorithm. Examples of application `MapDE` are given in §3.6, and we conclude with a discussion in §3.7. Our `MapDE` program and a demo file are publicly available on GitHub at: https://github.com/GregGitHub57/MapDETools.

3.2 Differential-elimination algorithms, initial data and Hilbert functions

The geometric approach to DPS centers on the jet locus, the solution set of the equations obtained by replacing derivatives with formal variables, yielding systems of polynomial equations and inequations and differences of varieties (solution sets of polynomial equations). In this way the algorithmic tools of algebraic geometry can be applied to systems of DPS. The union of prolonged graphs of local solutions of a DPS is a subset of the jet locus in $J(\mathbb{C}^n, \mathbb{C}^m)$, the jet

space, with n independent variables, and m dependent variables. For details concerning Jet geometry see [31, 41].

Throughout this paper we make prolific use of differential-elimination algorithms which apply a finite number of differentiations and eliminations to an input DPS to yield in a form that yields information about its properties and solutions. For example, consider

$$u_{xyy} - u_{yy} = 0, \quad u_{xx} + u_{xy} - u_x - u_y = 0, \quad u_{xx} - u_{xy} - u_x + u_y = 0 \tag{3.2.1}$$

Simply eliminating using the ordering $u_{xx} \succ u_{xy} \succ u_{yy} \succ u_x \succ u_y$ gives the equivalent system $u_{xyy} = u_{yy}, u_{xx} = u_x, u_{xy} = u_y$. The first equation can be omitted since it is a derivative of the third, yielding

$$u_{xx} = u_x, \quad u_{xy} = u_y \tag{3.2.2}$$

The operations to reduce the example above mirror those to reduce a related polynomial system via $\frac{\partial}{\partial x} \leftrightarrow X$, $\frac{\partial}{\partial y} \leftrightarrow Y$, $XY^2 - Y^2 = 0, X^2 + XY - X - Y = 0, X^2 - XY - X + Y = 0$ to a Gröbner Basis. Indeed a natural generalization of Gröbner Bases exists for linear homogeneous PDE. However DPS are much tougher theoretically and computationally, with straightforward generalizations yielding infinite bases, and undecidable problems. Currently we use the RIF algorithm via the `Maple` command `rifsimp` [40] in our implementation, but other `Maple` packages could be used such as `DifferentialThomas` Package [42, 37], and the `DifferentialAlgebra` package[9, 10, 21] or `casesplit` which offers a uniform interface to such packages.

A key aspect of these DEC packages for DPS is that they split on cases where certain leading polynomial quantities are zero or nonzero. This leads to systems of differential polynomial equations and inequations. In particular a system R of equations $\{p_1 = 0, p_2 = 0, \cdots, p_b = 0\}$ and inequations $\{q_1 \neq 0, \cdots, q_c \neq 0\}$ has solution locus

$$Z(R) = V^{=}(R) \setminus V^{\neq}(R) \tag{3.2.3}$$

where $V^{=}(R)$ are the solutions satisfying $\{p_1 = 0, p_2 = 0, \cdots, p_c = 0\}$ and $V^{\neq}(R)$ is the set of solutions of $\Pi_{i=1}^{i=c} q_i = 0$.

Moreover, a central input in such algorithms are rankings of derivatives [39]. Indeed let $\Omega(R)$ be all the derivatives of dependent variables for R. Throughout this paper the set of derivatives also includes 0-order derivatives (i.e. dependent variables). A ranking on $\Omega(R)$ is a total order $\prec$ that satisfies the axioms in [39]. Given a ranking and algorithms in [40] determine initial data and the existence and uniqueness of formal power series solutions. Additionally, if the ranking is orderly and of Riquier type (i.e. ordered first by total order of derivative, with a ranking specified by a Riquier ranking matrix) analytic initial data yield local analytic solutions. See [36] for a proof of this result. We will need some block elimination rankings that eliminate groups of dependent variables in favor of others. Enforcing the block order via the first row of the Riquier Matrix and then enforcing total order of the derivative as the next criterion for each block enables analytic data to yield analytic solutions that is sufficient for this paper.

The differential-elimination algorithms used in this article enable the algorithmic posing of initial data for the determination of unique formal power series of differential systems. We will exploit a powerful measure of solution dimension information given by Differential Hilbert Series and its related Differential Hilbert Function [28, 20]. Indeed given a ranking algorithms

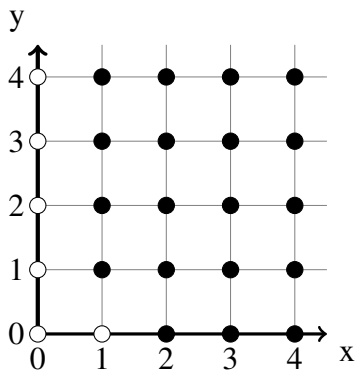


Figure 3.1: In the figure, the empty circles ∘ correspond to parametric derivatives, and solid circles • corresponding to derivatives of leaders. The Hilbert Series is obtained by diagonal counting of the empty circles.

such as the RIF algorithm, Rosenfeld Groebner and Thomas Decomposition partition the set of derivatives of the unknown function at a regular point x_0 into a set of parametric derivatives $\mathscr{P}$ and a set of principal derivatives. Principal derivatives are derivatives of the leading derivatives and parametric derivatives are its complement, the ones that can be ascribed arbitrary values at x_0. Then the Hilbert Series is defined as:

$$\mathrm{HS}(R, x_0, s) := \sum_{G \in \mathscr{P}} s^{\mathrm{dord}(G)} = \sum_{n=0}^{\infty} a_n s^n \tag{3.2.4}$$

where dord(G) is the differential order of G, a_n is the number of parametric derivatives of differential order n. To algorithmically compute such a series exploits the fact that the parametric data can be partitioned into subsets.

For example consider $R = \{u_{xx} = u_x, u_{xy} = u_y\}$, which is already in RIF-form with respect to an orderly ranking. Then $\mathscr{P} = \{u_x\} \cup \{u, u_y, u_{yy}, u_{yyy}, \cdots\}$, and the associated set of initial data is $\{u_x(x_0, y_0) = c_0\} \cup \{u(x_0, y_0) = c_1, u_y(x_0, y_0) = c_2, u_{yy}(x_0, y_0) = c_3, \cdots\}$. In what follows it is helpful to associate these derivatives with corresponding points in $\mathbb{N}^2$ via $\frac{\partial^{i+j}}{\partial x^i \partial y^j} u \leftrightarrow (i, j) \in \mathbb{N}^2$. See Fig. 3.1 for a graphical depiction of $\mathscr{P}$.

Then

$$\mathrm{HS}(R, (x_0, y_0), s) = 1 + 2s + s^2 + s^3 + s^4 + \cdots \tag{3.2.5}$$

A crucial condition to check in our algorithms will be that two Hilbert series are the same, which is complicated since no finite algorithm exists for checking equality of series. For our example, however, the series can be expressed finitely $\mathrm{HS}(R, (x_0, y_0)) = 1+2s+s^2+s^3+s^4+\cdots = s + \frac{1}{1-s}$. This collapsing of the series into a rational function can be accomplished in general. In our implementation, we exploited the output of the `initialdata` algorithm in the RIF package. For example this returns data as a partition of two sets: a finite set of initial data $\mathscr{F}$ and an set that represents an infinite set of data $\mathscr{I}$ by compressing them into arbitrary functions. For our example this compression is:

$$\mathscr{F} \cup \mathscr{I} = \{u_x(x_0, y_0) = c_0\} \cup \{u(x_0, y) = f(y)\} \tag{3.2.6}$$

This approach is easily extended to yield the Differential Hilbert Function by applying a function that acts on each piece of initial data in $\mathscr{F}$ and $\mathscr{I}$

$$\mathrm{HF}(\mathscr{F}, \mathscr{I}) := \sum_{G \in \mathscr{F}} s^{\mathrm{dord}(G)} + \sum_{G \in \mathscr{I}} \frac{s^{\mathrm{dord}(G)}}{(\mathrm{free}(G) - 1)!} \left(\frac{d}{ds}\right)^{\mathrm{free}(G)-1} (1 - s)^{-1} \tag{3.2.7}$$

where we have suppressed the dependence on x_0 and s for notational brevity. In the above formula free(G) is the number of free variables in the right hand side of the infinite data set $\mathscr{I}$. So for our example with $G = u(x_0, y) = f(y)$ we get free($u(x_0, y) = f(y)$) = 1 and dord(G) = 0. Then the formula yields as before

$$\mathrm{HF}(\mathscr{F}, \mathscr{I}) := s + \frac{1}{1-s} \tag{3.2.8}$$

For leading linear DPS, in RIF-form with respect to an orderly Riquier ranking, the Differential Hilbert Series gives coordinate independent dimension information. If a ranking is not orderly then the Differential Hilbert Series is no longer invariant. For example just consider the initial data for $v_{xx} = v_t$ in an orderly ranking compared to $v_t = v_{xx}$ in a non-orderly ranking.

The output of the initial data which partitions the parametric derivatives into disjoint cones of various dimensions $\leq n$, can express this in the rational function form $\mathrm{HS}(s) = \frac{P(s)}{(1-s)^{\mathbf{d}}}$ where $\mathbf{d} = \mathbf{d}(R)$ is the differential dimension of R. It corresponds to the maximum number of free independent variables appearing in the functions for the initial data. For further information on differential Hilbert Series see [28]. The algorithms are simple modifications of those for Gröbner bases for modules.

3.3 Symmetries & Mapping Equations

3.3.1 Symmetries

Infinitesimal Lie *point* symmetries for R are found by seeking vector fields

$$V = \sum_{i=1}^{n} \xi^i(x,u)\frac{\partial}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x,u)\frac{\partial}{\partial u^j} \tag{3.3.1}$$

whose associated one-parameter group of transformations

$$\begin{aligned} x^* &= x + \xi(x,u)\epsilon + O(\epsilon^2) \\ u^* &= u + \eta(x,u)\epsilon + O(\epsilon^2) \end{aligned} \tag{3.3.2}$$

away from exceptional points preserve the jet locus of such systems, mapping solutions to solutions. See [5, 6] for applications. The *infinitesimals* (ξ^i, η^j) of a symmetry vector field (3.3.1) for a system of DEs are found by solving an associated system of linear homogeneous defining equations S (or determining equations) for the infinitesimals. The defining system S is derived by a prolongation formula for which numerous computer implementations exist [12, 13, 38]. Lie's classical theory of groups and their algebras requires local analyticity in its defining equations. Such local analyticity will be a key assumption throughout our paper.

The resulting vector space of vector fields is closed under its commutator. The commutator of two vector fields for vector fields $X = \sum_{i=1}^{m+n} \nu^i \frac{\partial}{\partial z^i}$, $Y = \sum_{i=1}^{m+n} \mu^i \frac{\partial}{\partial z^i}$ in a Lie algebra $\mathscr{L}$ and $z = (x, u)$, is:

$$[X, Y] = XY - YX = \sum_{i=1}^{m+n} \omega^i \frac{\partial}{\partial z^k} \tag{3.3.3}$$

where $\omega^k = \sum_{i=1}^{m+n} \left(\nu^i \mu^k_{z^i} - \mu^i \nu^k_{z^i}\right)$.

Similarly, we suppose that the *Target* admits symmetry vector fields

$$\hat{V} = \sum_{i=1}^{n} \hat{\xi}^i(\hat{x}, \hat{u}) \frac{\partial}{\partial \hat{x}^i} + \sum_{j=1}^{m} \hat{\eta}^j(\hat{x}, \hat{u}) \frac{\partial}{\partial \hat{u}^j} \tag{3.3.4}$$

in the *Target* infinitesimals $(\hat{\xi}, \hat{\eta})$ that satisfies a linear homogeneous defining system $\hat{S}$ generating a Lie algebra $\hat{\mathscr{L}}$. Computations with defining systems will be essential in our approach and are implemented using Huang and Lisle's powerful object oriented LAVF, `Maple` package [22].

Example 3.3.1 *Consider as a simple example the third order nonlinear ODE which is in* RIF-*form with respect to an orderly ranking*

$$u_{xxx} = \frac{3\left(uu_{xx} + {u_x}^2 + 1\right)^2}{u(uu_x + x)} - \frac{3u_x u_{xx}}{u} + \frac{8x\left(uu_x + x\right)^4 \left(u^2 + x^2 + 1\right)}{u(u^2 + x^2)} \tag{3.3.5}$$

at points $u \neq 0, uu_x + x \neq 0, u^2 + x^2 \neq 0$. *When* `Maple`*'s* `dsolve` *is applied to (3.3.5) it yields no result. Later in this section, we will discover important information about (3.3.5) using symmetry aided mappings. It will be used as a simple running example to illustrate the techniques of the article.*

The defining system for Lie point symmetries of form $\xi(x,u)\frac{\partial}{\partial x} + \eta(x,u)\frac{\partial}{\partial u}$ *of (3.3.5) has* RIF *form with respect to an orderly ranking given by:*

$$\begin{aligned} S = [\xi = -\frac{\eta\, u}{x}, \quad \eta_{x,u} &= \frac{(u-x)(u+x)\eta}{u^3 x} + \frac{\left(u^2+x^2\right)\eta_u}{xu^2} - \frac{\eta_x}{u} + \frac{x\eta_{u,u}}{u}, \\ \eta_{x,x} &= -\frac{\left(2\,u^4 - x^2u^2 + x^4\right)\eta}{u^4x^2} + \frac{\left(u^2+x^2\right)\eta_u}{u^3} + 2\,\frac{\eta_x}{x} + \frac{x^2\eta_{u,u}}{u^2}, \\ \eta_{u,u,u} &= -\frac{\left(16\,u^8 + 24\,u^6x^2 + 8\,u^4x^4 + 16\,u^6 + 8\,u^4x^2 + 3\,u^2 + 3\,x^2\right)\eta}{\left(u^2+x^2\right)u^3} \\ &\quad - \frac{\left(8\,u^6x^2 + 8\,u^4x^4 + 8\,u^4x^2 - 3\,u^2 - 3\,x^2\right)\eta_u}{u^2\left(u^2+x^2\right)} + 8\,\frac{u^3x\left(u^2+x^2+1\right)\eta_x}{u^2+x^2}] \end{aligned} \tag{3.3.6}$$

Its corresponding initial data is

$$\mathrm{ID}(S) = [\eta(x_0,u_0) = c_1, \eta_x(x_0,u_0) = c_2, \eta_u(x_0,u_0) = c_3, \eta_{u,u}(x_0,u_0) = c_4] \tag{3.3.7}$$

There are 4 *arbitrary constants in the initial data at regular points* (x_0, u_0)*, so (3.3.5) has a* 4 *dimensional local Lie algebra of symmetries* $\mathscr{L}$ *in a neighborhood of such points:* $\dim \mathscr{L} = 4$. *The structure of* $\mathscr{L}$ *of (3.3.6) can be algorithmically determined without integrating the defining system [34, 23, 22, 16]:*

$$\begin{aligned} &[Y_1, Y_2] = -Y_1 - 2\,Y_2, \quad [Y_1, Y_3] = Y_1 - 2\,Y_3, \quad [Y_1, Y_4] = -2\,Y_4, \\ &[Y_2, Y_3] = Y_2 + Y_3, \quad [Y_2, Y_4] = Y_4, \quad [Y_3, Y_4] = -Y_4 \end{aligned} \tag{3.3.8}$$

where a regular point $(x_0 = 1, u_0 = 1)$ *was substituted into the relations.*

But what can such symmetry information tell us about nonlinear systems R such as the above ODE using mappings? In particular, in this paper we focus on the question of when a system R can be mapped to a linear system $\hat{R}$. Throughout this paper we maintain blanket local analyticity assumptions. So in the case of a single linear differential equation $\hat{R}$ has the form $\mathscr{H}\hat{u} = f(\hat{x})$ where $\mathscr{H}$ is a linear differential operator with coefficients that are analytic functions of $\hat{x}$ and f is also analytic. Lewy's famous counterexample of a single linear differential equation in 3 variables, of order 1, where $\mathscr{H}$ is analytic with smooth inhomogeneous term, without smooth solutions, provides a counterexample in the smooth case. Then supposing we have the existence of a local analytic solution $\tilde{u}$ in a neighborhood of $\hat{x}_0$, in this neighborhood the point transformation $\hat{u} \to \hat{u} - \tilde{u}$ implies that without loss we can consider $\hat{R}$ to be a homogeneous linear differential equation $\mathscr{H}\hat{u} = 0$ where $\hat{u} \in \mathscr{A}(\hat{x}_0, \delta)$, the set of analytic functions on some sufficiently small disk $|\hat{x} - \hat{x}_0| < \delta$. Then solutions of $\hat{R}$ satisfy the superposition property $\mathscr{H}(\hat{v} + \hat{w}) = \mathscr{H}\hat{v} + \mathscr{H}\hat{w} = 0$. This corresponds to point symmetries generated by the Lie algebra of vector fields

$$\hat{\mathscr{L}}^* := \left\{ \hat{v}(\hat{x}) \frac{\partial}{\partial \hat{u}} : \mathscr{H}\hat{v}(\hat{x}) = 0 \text{ and } \hat{v} \in \mathscr{A}(\hat{x}_0, \delta) \right\} \tag{3.3.9}$$

Consequently, assuming the existence of a local analytic map, $\dim \hat{\mathscr{L}}^* = \dim \hat{R} = \dim R$. If R is an ODE of order $d \geq 2$ then $\dim \hat{\mathscr{L}}^* = \dim \hat{R} = \dim R = d$. Similarly the superposition property $\mathscr{H}(c\hat{v}) = c\mathscr{H}\hat{v} = 0$ corresponds to a 1 parameter family of scalings with symmetry vectorfield $\hat{u}\frac{\partial}{\partial \hat{u}}$. So we get the well-known and obvious result that an ODE of order d that can be mapped to a linear ODE must have $\dim \mathscr{L} = \dim \hat{\mathscr{L}} \geq d + 1$. Similarly if R is linearizable and $\dim R = \infty$ then $\dim \mathscr{L} = \dim \hat{\mathscr{L}} = \infty$ with similar properties for systems. The Lie sub-algebra $\hat{\mathscr{L}}^*$ is easily shown to be 'abelian' by direct computation of commutator [32] in both the finite and infinite dimensional case. Indeed consider the so-called derived algebra $\mathscr{L}' = \text{DerivedAlgebra}(\mathscr{L})$, which is the Lie subalgebra of $\mathscr{L}$ generated by commutators of members of $\mathscr{L}$ and similarly for $\hat{\mathscr{L}}'$. By direct computation of commutators $\hat{\mathscr{L}}^*$ is a subalgebra of $\hat{\mathscr{L}}'$ (e.g. $[\hat{v}(\hat{x})\frac{\partial}{\partial \hat{u}}, \hat{u}\frac{\partial}{\partial \hat{u}}] = \hat{v}(\hat{x})\frac{\partial}{\partial \hat{u}} \in \hat{\mathscr{L}}$). Thus, a necessary condition for the existence of a map Ψ to a linear target is that $\hat{\mathscr{L}}'$ has a d dimensional abelian subalgebra in the finite and infinite dimensional cases (see Olver [32]). In the preceding paragraph we considered the case of a single dependent variable, which is easily extended to the multivariate case. For example in equation (3.3.9) the symmetry generator $\hat{v}(\hat{x})\frac{\partial}{\partial \hat{u}}$ can be replaced by $\sum_{i=1}^{m} \hat{v}^i(\hat{x})\frac{\partial}{\partial \hat{u}^i}$ for the case of a system.

Lyakhov, Gerdt and Michels [25] use this to implement a remarkable algorithm to determine the existence of a linearization for a single ODE of order d. See Algorithm 5 and [25, 26] for further background. There are two main cases. The first is when the nonlinear ODE has a Lie symmetry algebra of maximal dimension, as shown in Step 5 of Algorithm 5. Such maximal cases are always linearizable. These occur for $d = 1, 2$ where the maximal dimensions of $\mathscr{L}$ are ∞ and 8 respectively, and for $d > 2$ where the maximal dimension of $\mathscr{L}$ is $d + 4$. The second main case is sub-maximal and occurs for $d > 2$ when $\dim \mathscr{L} = d + 1$ or $\dim \mathscr{L} = d + 2$.

Example 3.3.2 *We illustrate the above discussion and Algorithm 5 by a continuation of Example 3.3.1. For that example* $d = 3$ *and* $\dim \mathscr{L} = 4$. *Then from the commutation relations the*

Algorithm 5 LGMLinTest(R)

Input: a leading linear ODE R solved for its highest derivative of order ≥ 1
Output: Lin = true if R linearizable otherwise Lin = false
1: Lin:= false
2: Compute S := ThomasDecomposition(DetSys(R))
3: Find dim $\mathscr{L}$:= dim S, d := difforder(R)
4: ComRels := Structure(S)
5: **if** $d = 1$ or ($d = 2$ and dim $\mathscr{L} = 8$) or ($d > 2$ and dim $\mathscr{L} = d + 4$) **then** Lin := true
else if $d > 2$ and (dim $\mathscr{L} = d + 1$ or dim $\mathscr{L} = d + 2$) **then**
$\mathscr{L}'$:= DerivedAlgebra(ComRels)
if IsAbelian($\mathscr{L}'$) and d = dim($\mathscr{L}'$) **then** Lin := true
end if
end if
6: **return** Lin

derived algebra $\mathscr{L}'$ is generated by

$$[Z_1 = Y_1 - 2\,Y_3,\ Z_2 = Y_2 + Y_3,\ Z_3 = Y_4] \tag{3.3.10}$$

Thus dim *DerivedAlgebra*($\mathscr{L}$) = dim $\mathscr{L}'$ = 3. *Also its structure is easily found as*

$$[Z_1, Z_2] = [Z_1, Z_3] = [Z_2, Z_3] = 0 \tag{3.3.11}$$

so $\mathscr{L}'$ is abelian and by Algorithm 5, (3.3.5) is exactly linearizable.

The Algorithms introduced by Lyakhov et al. [25] have two stages: the first given above is to determine whether the system is lineaizable. The second stage is to attempt to construct an explicit form for the mapping by integration. A fundamental algorithmic tool for both stages is the `ThomasDecomposition` algorithm which is a differential elimination algorithm which outputs a disjoint decomposition of a DPS finer than that of [9] or [40]. It is based on the work of Thomas [42, 37]. The algorithm is available in distributed Maple 18 and later versions. We note that the construction step involves heuristic integration. The algorithm that they use to construct a system for the mapping to a linear ODE, before it is reduced using `ThomasDecomposition`, is related to the algorithm `EquivDetSys` given in our introductory paper [29]. It is expensive as we illustrate later with examples. One of the contributions of our paper is to find a potentially more efficient algorithm that avoids the application of the full nonlinear equivalence equations generated by `EquivDetSys` in [29].

3.3.2 Bluman-Kumei Mapping Equations

Assume the existence of a local analytic invertible map $\Psi = (\psi, \phi)$ between the *Source* system R and the *Target* system $\hat{R}$, with Lie symmetry algebras $\mathscr{L}$, $\hat{\mathscr{L}}$ respectively. Applying Ψ to the infinitesimals $(\hat{\xi}, \hat{\eta})$ of a vectorfield in $\hat{\mathscr{L}}$ yields what we will call the *Bluman-Kumei (BK) mapping equations*:

$$M_{\mathrm{BK}}(\mathscr{L}, \hat{\mathscr{L}}) = \begin{cases} \hat{\xi}^k(\hat{x}, \hat{u}) &= \sum_{i=1}^{n} \xi^i(x,u) \frac{\partial \psi^k}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x,u) \frac{\partial \psi^k}{\partial u^j} \\ \hat{\eta}^\ell(\hat{x}, \hat{u}) &= \sum_{i=1}^{n} \xi^i(x,u) \frac{\partial \phi^\ell}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x,u) \frac{\partial \phi^\ell}{\partial u^j} \end{cases} \tag{3.3.12}$$

where $1 \le k \le n$ and $1 \le \ell \le m$, and (ξ, η) are infinitesimals of Lie symmetry vectorfields in $\mathscr{L}$. See Bluman and Kumei [4, 8] for details and generalizations (e.g. to contact transformations). Note that all quantities on the RHS of the BK mapping equations (3.3.12) are functions of (x, u) including ϕ and ψ. See [29, Example 1] for an introductory example of mappings and the examples in [8].

Remark 1 *When considered together with $\hat{x} = \psi(x,u), \hat{u} = \phi(x,u)$ the BK mapping equations (3.3.12) are a change of variables from (x,u) to $(\hat{x},\hat{u})$ coordinates. Simply interchanging target and source variables then yields the inverse of the BK mapping equations below. Considered together with $x = \hat{\psi}(\hat{x},\hat{u}), u = \hat{\phi}(\hat{x},\hat{u})$ these are a change of variables from $(\hat{x},\hat{u})$ to (x,u) coordinates.*

$$M_{\mathrm{BK}}(\hat{\mathscr{L}}, \mathscr{L}) = \left\{ \begin{array}{lcl} \xi^k(x,u) & = & \sum_{i=1}^n \hat{\xi}^i(\hat{x},\hat{u}) \frac{\partial \hat{\psi}^k}{\partial \hat{x}^i} + \sum_{j=1}^m \hat{\eta}^j(\hat{x},\hat{u}) \frac{\partial \hat{\psi}^k}{\partial \hat{u}^j} \\ \\ \eta^\ell(x,u) & = & \sum_{i=1}^n \hat{\xi}^i(\hat{x},\hat{u}) \frac{\partial \hat{\phi}^\ell}{\partial \hat{x}^i} + \sum_{j=1}^m \hat{\eta}^j(\hat{x},\hat{u}) \frac{\partial \hat{\phi}^\ell}{\partial \hat{u}^j} \end{array} \right. \tag{3.3.13}$$

We note the following relation between Jacobians in (x,u) and $(\hat{x},\hat{u})$ coordinate systems

$$\frac{\partial(\hat{\psi},\hat{\phi})}{\partial(\hat{x},\hat{u})} = \left[\frac{\partial(\psi,\phi)}{\partial(x,u)}\right]^{-1} \tag{3.3.14}$$

If an invertible map Ψ exists mapping R to $\hat{R}$ then it most generally depends on $\dim(\mathscr{L}) = \dim(\hat{\mathscr{L}})$ parameters. But we only need one such Ψ. In other words, if $\Psi : R \to \hat{R}$ is an inverible mapping then $\Psi \circ g : R \to \hat{R}$ is also an invertible mapping for any $g \in \mathscr{G}$. So reducing the number of such parameters, e.g., by restricting to a Lie subalgebra $\mathscr{L}'$ of $\mathscr{L}$ with corresponding Lie subalgebra $\hat{\mathscr{L}}'$ of $\hat{\mathscr{L}}$ that still enables the existence of such a Ψ, is important in reducing the computational difficulty of such methods. We will use the notation $S', \hat{S}'$ to denote the symmetry defining systems of Lie sub-algebras $\mathscr{L}'$, $\hat{\mathscr{L}}'$ respectively. See [8, 32] for discussion on this matter.

For mapping from nonlinear to linear systems, a natural candidate for $\mathscr{L}'$ is the DerivedAlgebra($\mathscr{L}$), and the natural Lie symmetry algebra for target is $\hat{\mathscr{L}}^*$ corresponding to the superposition defined in (3.3.9).

Example 3.3.3 *This is a continuation of Examples 3.3.1 and 3.3.2 concerning (3.3.5). Here we will use the BK mapping equations (3.3.12) where $\mathscr{L}'$ = DerivedAlgebra($\mathscr{L}$).*
For the construction of Ψ we actually need differential equations for $\mathscr{L}'$, in addition its structure which are provided by the algorithm `DerivedAlgebra` *in the* LAVF *package which for (3.3.5) yields its* RIF*-form:*

$$\begin{aligned} S' = [\xi = -\frac{\eta\, u}{x}, \eta_x = \frac{\left(u^2+x^2\right)\eta}{x u^2} + \frac{\eta_u x}{u}, \\ \eta_{u,u,u} = -\frac{\left(8\,u^8 + 8\,u^6 x^2 + 8\,u^6 + 3\,u^2 + 3\,x^2\right)\eta}{\left(u^2+x^2\right)u^3} + 3\,\frac{\eta_u}{u^2}] \end{aligned} \tag{3.3.15}$$

The derived algebra is then shown by LAVF *commands to be both 3 dimensional and abelian. Moreover, its determining system (3.3.15) is much simpler than the determining system of $\mathscr{L}$*

given in (3.3.6). Crucially it means we can exploit this determining system using the BK mapping equations. Since the target infinitesimal generator is $\hat{\xi}\frac{\partial}{\partial \hat{x}} + \hat{\eta}\frac{\partial}{\partial \hat{u}} = 0 \cdot \frac{\partial}{\partial \hat{x}} + \hat{\eta}(\hat{x})\frac{\partial}{\partial \hat{u}}$ *where* $\mathscr{H}\hat{u} = 0$. *So* $\hat{\xi} = 0$ *and* $\mathscr{H}\hat{u} = 0$ *and the BK equations are:*

$$M_{\mathrm{BK}}(\mathscr{L}', \hat{\mathscr{L}}^*) = \begin{cases} 0 &= \xi(x,u)\frac{\partial \psi}{\partial x} + \eta(x,u)\frac{\partial \psi}{\partial u} \\ \hat{\eta}(\hat{x},\hat{u}) &= \xi(x,u)\frac{\partial \phi}{\partial x} + \eta(x,u)\frac{\partial \phi}{\partial u} \end{cases} \tag{3.3.16}$$

where $\hat{\eta}_{\hat{u}} = 0$ *and* $\mathscr{H}\hat{u} = 0$. *These equations are an important necessary condition for the linearization of (3.3.5) and this will be exploited in Section §3.4 in the computation of the mapping.*

3.4 Algorithms and Preliminaries for the MapDE Algorithm

The `MapDE` algorithm introduced in [29] is extended to determine if there exists a mapping of a nonlinear source R to some linear target $\hat{R}$, using the target input option *Target* = `LinearDE`.

3.4.1 Symmetries of the linear target and the derived algebra

We summarize and generalize some aspects of the discussion in §3.1 and §3.3. The following theorem is a straightforward consequence of the necessary conditions in [8] where we have also required that the target system is in RIF-form.

Theorem 3.4.1 (Superposition symmetry for linearizable systems) *Suppose that the analytic system R is exactly linearizable by a local holomorphic diffeomorphism* $\hat{x} = \psi(x,u), \hat{u} = \phi(x,u)$ *to yield a linear target system. Then* $\hat{R}$ *locally takes the form*

$$\hat{R} : \mathscr{H}\hat{u}(\hat{x}) = 0 \tag{3.4.1}$$

where $\mathscr{H}$ *is a vector partial differential operator, with coefficients that are local analytic functions of* $\hat{x}$ *and the system (3.4.1) is in* RIF*-form with respect to an orderly ranking. Moreover* $\hat{R}$ *admits the symmetry vector field* $\sum_{j=1}^{m} \hat{\eta}^j(\hat{x})\frac{\partial}{\partial \hat{u}^j}$*:*

$$\hat{S}^* := \left\{ \hat{\xi}^i = 0,\ \mathscr{H}\hat{\eta} = 0,\ \hat{\eta}^j_{\hat{u}^k} = 0 : 1 \le i \le n,\ 1 \le j,k \le m \right\}. \tag{3.4.2}$$

From the previous discussion, computation of determining systems for derived algebras is important in both the finite and infinite cases. In the Remark below we sketch what appears to be the first algorithm to compute such systems in the infinite case.

Remark 2 (Algorithm for computation of infinite Derived Algebras) *A simple consequence of the commutator formula (3.3.3) is that the commutators generate a Lie algebra which is called the derived algebra. Lisle and Huang [22] implement efficient algorithm in the* LAVF *package to compute the determining system for the derived algebra for finite dimensional Lie algebras of vectorfields. We have made a first implementation in the infinite dimensional case, together with the Lie pseudogroup structure relations [34, 23]. First each of the* ν, μ, ω *in*

the commutation relations (3.3.3) must satisfy the determining system of $\mathscr{L}$ so we enter three copies of those determining systems. We then reduce the combined system using a block elimination ranking which ranks any derivative of ω strictly less than those of μ, ν. The resulting block elimination system for ω generates the derived algebra in the infinite case.

The commutator between any superposition generator and the scaling symmetry admitted by linear systems yields

$$\left[\sum_{i=1}^{m} \hat{v}^i(\hat{x})\frac{\partial}{\partial \hat{u}^i} \ , \ \sum_{i=1}^{m} \hat{u}^i \frac{\partial}{\partial \hat{u}^i}\right] = \sum_{i=1}^{m} \hat{v}^i(\hat{x})\frac{\partial}{\partial \hat{u}^i} \tag{3.4.3}$$

So we have the following result as an easy consequence (See Olver [32] for related discussion in both the finite and infinite case).

Theorem 3.4.2 *Suppose that the analytic system R is exactly linearizable by a local holomorphic diffeomorphism $\hat{x} = \psi(x,u), \hat{u} = \phi(x,u)$, to yield a linear target system (3.4.2) and $\mathscr{L}$, $\mathscr{L}'$ are the Lie symmetry algebra and its derived algebra for R. Also, let $\hat{\mathscr{L}}$, $\hat{\mathscr{L}}'$ be the corresponding algebras for $\hat{R}$. Let $\mathscr{L}^*$, $\hat{\mathscr{L}}^*$ be the superposition algebras under Ψ. Then $\hat{\mathscr{L}}^*$ is a subalgebra of $\hat{\mathscr{L}}'$ and $\mathscr{L}^*$ is a subalgebra of $\mathscr{L}'$. Moreover $\mathscr{L}^*$ and $\hat{\mathscr{L}}^*$ are abelian.*

We wish to determine if a system R is linearizable and if so, characterize the target $\hat{R}$, i.e $\mathscr{H}\hat{\eta} = 0$. But initially we don't know $\mathscr{H}$. One approach is to write a general form for this system that specifies $\hat{S}^*$ with undetermined coefficient functions whose form is established in further computation. See for example, [25] use this approach in the case of a single ODE, but don't consider the Bluman-Kumei mapping system. We will apply our method to a test set of ODE (3.6.9) given in [25]. Instead, we only include $\xi^i = 0, \hat{\eta}^j_{\hat{u}^k} = 0$ and don't include $\mathscr{H}\hat{\eta} = 0$. Thus, we only include a subset $\hat{S}^\star$ of $\hat{S}^*$, denoting the truncated system as

$$\hat{S}^\star := \left\{\hat{\xi}^i = 0, \hat{\eta}^j_{\hat{u}^k} = 0 : 1 \le i \le n,\ 1 \le j,k \le m\right\} \tag{3.4.4}$$

and allow $\mathscr{H}\hat{\eta} = 0$ to be found naturally later in the algorithm. We note that $\hat{S}^\star$ are the defining equations of a (usually infinite) Lie pseudogroup.

3.4.2 Algorithm PreEquivTest for excluding obvious nonlinearizable cases

As discussed in §3.3, R being linearizable implies that the superposition is in its Lie symmetry algebra and is a coordinate change of R. This implies some fairly well-known efficient tests for screening out obvious non-linearizable cases. For linearization necessarily $\dim S \ge d+1$ and $\dim S' \ge d$ for finite d. For $d = \infty$, necessarily $\dim S = \infty = \dim S'$ and in terms of differential dimensions $\mathbf{d}(S) \ge \mathbf{d}(S') \ge \mathbf{d}(R)$. Note that Algorithm 6 returns null, if all its tests are true. The most well-known of the above tests occur when $d = \infty$ and $\dim S = \infty$ and are given for example, in Bluman and Kumei [6]. Also see Theorem 6.46 in Chapter 6 of Olver [32].

Algorithm 6 `PreEquivTest`

`PreEquivTest`(R, IDR, IDS, IDS′)

Input: R is a leading linear DPS system in DEC-form R with no leaders of order 0 with respect to an orderly Riquier ranking

IDR, IDS, IDS′ are respectively the initial data for R, S and S' where S and S' are the symmetry determining systems for $\mathscr{L}$ and $\mathscr{L}'$

Output: [IsLinearizable, DimInfo]

IsLinearizable = false if one of the necessary conditions T_j tests false

DimInfo is the dimension info (dimension, differential dimension and Differential Hilbert Function, computed for each of R, S, S'.

1: Set IsLinearizable := null. Apply DifferentialHilbertFunction to IDR, IDS, IDS′ to get

2: DimInfo := [dim R, $\mathbf{d}(R)$, HF(R), dim S, $\mathbf{d}(S)$, HF(S) , dim S', $\mathbf{d}(S')$, HF(S')]

3: **if** $d = \infty$ **then**

T_1 := evalb(dim S $< \infty$)

T_2 := evalb(dim S' $< \infty$)

T_3 := evalb($\mathbf{d}(S) < \mathbf{d}(R)$)

T_4 := evalb($\mathbf{d}(S') < \mathbf{d}(R)$)

else if $d < \infty$ **then**

T_5 := evalb(dim S $< d + 1$)

T_6 := evalb(dim S' $< d$)

end if

4: **if** $\wedge_{i=1}^{i=6} T_i$ = false **then** IsLinearizable := false; **end if**

return [IsLinearizable, DimInfo]

3.4.3 Algorithm ExtractTarget for extracting the linear target system

When a system R is determined to be linearizable by Algorithm 7, the conditions for linearizability will yield a list of cases $\bigcup_{c\in C} Q_c$ where each Q_c is in RIF-form. To implicitly determine the target linear system $\hat{R}$ for a case Q_c, Algorithm ExtractTarget is applied to Q_c. It first selects from Q_c the linear homogeneous differential sub-system R_c^* in $\xi(x,u)$, $\eta(x,u)$ with coefficients depending on (x,u,ψ,ϕ) in the (x,u) coordinates. Algorithm ExtractTarget then applies the inverse BK transformations (3.3.13) to convert the system R_c^* to $(\hat{x},\hat{u})$ coordinates, after which $\hat{\xi} = 0$ and also $\hat{\eta}(\hat{x},\hat{u}) = \hat{\eta}(\hat{x})$ is imposed. This yields R_c^* as a system $\hat{R}_c^*$ which is a linear homogeneous differential system in $\hat{\eta}(\hat{x})$ with coefficients in $\hat{x},\hat{u},\hat{\psi},\hat{\phi}$. Though R_c^* is in RIF-form, $\hat{R}_c^*$ is not usually in RIF-form so another application of RIF is applied, to yield $\hat{R}_c^*$ in RIF-form for $\hat{\eta}(\hat{x})$; case splitting is not required here. As shown in the proof of Algorithm 7, the coefficients of $\hat{R}_c^*$ depend only on $\hat{x},\hat{\psi},\hat{\phi}$ and not on $\hat{u}$. The proof of Algorithm 7 also shows that $\hat{\eta}(\hat{x})$ can be replaced in $\hat{R}_c^*$ with $\hat{u}(\hat{x})$ yielding the target linear homogeneous differential equation $\hat{R}_c$.

3.4.4 Heuristic integration for the Mapping functions in MapDE using PDSolve

This routine is still in the early stages of its development. The heuristic integration routine `PDSolve`, basically a simple interface to `Maple`'s `pdsolve` which is applied to the Ψ sub-system (the sub-system with highest derivatives in ψ, ϕ) together with its inequations in Q_c to attempt to find an explicit form of the mapping Ψ. We naturally use a block elimination ranking in Ψ sub-system where all derivatives of ϕ are higher than all derivatives of ψ. Then we attempt to solve uncoupled subsystem for ψ using the LAVF routine Invariants, which depends on integration, and subsequently solve the substituted system for ϕ by `pdsolve`.

The geometric idea is that in the $(\hat{x}, \hat{u})$ coordinates the Lie symmetry generator corresponding to linear superposition has the form $\sum_k \hat{\eta}^k(\hat{x})\frac{\partial}{\partial \hat{u}^k}$ and generates an abelian Lie algebra with obvious invariants $\hat{x}$. So the independent variables for the target linear equation are invariants of this vector field acting on the base space of variables $(\hat{x}, \hat{u})$, and thus on (x, u) space via the map Ψ. The process of integrating the mapping equations first starts with the determination of these invariants in terms of (x, u) using the LAVF command `Invariants`. If the integration is successful this yields $\hat{x}^j = \psi^j = I^j(x, u)$, for $j = 1, \cdots, n$. Then substitution into the Ψ sub-system, yields a system with dependence only on the $\hat{\phi}$ mapping functions, which we attempt to integrate using Maple's `pdsolve`.

3.5 The MapDE Algorithm

The main subject here is Algorithm 7 which makes heavy use of differential-elimination completion (DEC) algorithms, which in our current implementation is the RIF algorithm accessed via `Maple`'s `rifsimp`. Other DEC algorithms could be used such as `ThomasDecomposition`, `RosenfeldGroebner` or `casesplit`.

In §3.5.1 we will describe pseudo-code for `MapDE`. In §3.5.2 we will give notes about the steps of `MapDE` and in §3.5.3 we will a proof of correctness of `MapDE`.

3.5.1 Pseudo-code for the MapDE algorithm

Here we describe the pseudo-code for `MapDE`.

Algorithm 7 `MapDE` with `Target =LinearDE`

`MapDE`(*Source*, *Target*, *Map*, *Options*)

Input:

Source: A leading linear DPS system in DEC-form R with no leaders of order 0 with respect to an orderly Riquier ranking; vars $[x, u]$, $[\xi, \eta]$

Target: `Target =LinearDE`

Map: Ψ

Options: Additional options (for strategies, outputs, etc).

Output:

IsLinearizable = false if there $\nexists$ an invertible local linearization Ψ

IsLinearizable = true if there $\exists$ an invertible local linearization Ψ and

— Collection of cases in RIF-form yielding such linearizations

— Implicit form of the target linear equation for $\hat{R}$ (see 3.4.3)

— Explicit form of Ψ if the heuristic method `PDSsolve` is successful (see §3.4.4)

1: Set IsLinearizable := null. Compute IDR := ID(R)

2: Let $\mathscr{L}'$ = `DerivedAlgebra`($\mathscr{L}$) and compute:
S := DEC(DetSys($\mathscr{L}$)), S' := DEC(DetSys($\mathscr{L}'$))
IDS := ID(S), IDS′ := ID(S')

3: [IsLinearizable, DimInfo] := `PreEquivTest`(R, IDR, IDS, IDS′)
if IsLinearizable = false **then return** [IsLinearizable, DimInfo] **end if**
When R is an ODE also calculate LGMLin := LGMLinTest(R).

4: Set $\hat{S}^\star := \left\{\hat{\xi}^i = 0, \hat{\eta}^j_{\hat{u}^k} = 0 : 1 \le i \le n,\ 1 \le j, k \le m\right\}$
$M := S' \cup \hat{S}^\star|_\Psi \cup M_{BK}(\mathscr{L}', \hat{\mathscr{L}}^\star) \cup \{\text{DetJac}(\Psi) \neq 0\}$

5: Compute list of consistent cases $P = [P_1, \cdots, P_{\text{nc}}]$ with dim $\ge d$:
P := DEC($M, \prec, casesplit, mindim = d$)

6: **if** $P = \emptyset$ **then** IsLinearizable := false **return** [IsLinearizable, DimInfo] **end if**

7: Q := [ ]

8: **for** $k = 1$ **to** nc **do**

9: **if** HF(R) = HF(P_k) **then** $Q := Q \cup P_k$ **end if**
end do

10: **if** Q = [ ] **then** IsLinearizable := false **return** [IsLinearizable, DimInfo]
else if $Q \neq$ [ ] **then** IsLinearizable := true
end if

11: **if** CaseSelect $\notin$ Options **then** C := [1] **else** Assign C using Options **end if**

12: **for** $c \in C$ **do** $\hat{R}_c$:= ExtractTarget(Q_c) **end do** (See 3.4.3)

13: **for** $c \in C$ **do** Attempt heuristic integration Ψ^c_{sol} := `PDSolve`(Q_c) **end do** (See 3.4.4)

14: **return** $\bigcup_{c\in C} [Q_c, \hat{R}_c, \hat{R}_c|_{\Psi^c_{\text{sol}}}, \Psi^c_{\text{sol}}]$

Abbreviations used above: DEC: Differential Elimination Completion, M_{BK} BK system, ID: InitialData

3.5.2 Notes on the MapDE Algorithm with Target = LinearDE

We briefly list some main aspects of Algorithm 7.

Input: Due to current limitations of `Maple`'s `DeterminingPDE` we restrict to input a single system R, in DEC (i.e. RIF-form) leading linear equations with leading derivatives of differential order ≥ 1, together with inequations and no leading nonlinear equations. This form is more general than CauchyKowalevski form, and includes over and under-determined systems, but not systems with 0 order (algebraic) constraints.

`Options` refers to additional options for strategies and outputs. For example including OutputDetails in Options yields more detailed outputs.

Step 2: See Remark 2, where we briefly describe our new algorithm for computing determining systems for infinite dimensional derived algebras.

Step 3: As discussed in §3.3, R being linearizable means that linear superposition generates a symmetry sub-algebra of $\mathscr{L}$, yielding some fairly well-known efficient tests for rejecting many non-linearizable systems. See Algorithm 6 for details. We also apply Algorithm 5 for the LGMLinTest [25] when R is an ODE in order to compare and test our Hilbert linearization test which occurs later in the algorithm.

Step 4: $\hat{S}^\star|_\Psi$ is $\hat{S}^\star$ evaluated in (x,u) coordinates via Ψ using differential reduction. $\hat{S}^\star := \left\{\hat{\xi}^i = 0, \hat{\eta}^j_{\hat{u}^k} = 0 : 1 \leq i \leq n,\ 1 \leq j,k \leq m\right\}$

Step 5: Here `mindim` $= \dim(R) = d$, as computed by the `Maple` command `initialdata` of the DEC form of R. The mindim option avoids computing cases of dimension $< d$ by monitoring an upper bound based on initial data of such cases. The block elimination ranking $\prec$ ranks all infinitesimals and their derivatives for the first block $[\xi, \eta, \hat{\xi}, \hat{\eta}]$ strictly greater than the second block of the ϕ map variables, which are strictly greater than all derivatives of the third block of ψ variables. This maintains linearity in the variables $[\xi, \eta, \hat{\xi}, \hat{\eta}]$. The mindim dimension is computed with respect to these variables, and not the degrees of freedom in the map variables (ψ, ϕ). The block structure also facilitates the later integration phase. Each case P_k consists of equations and inequations.

Step 9: The Hilbert Functions of R and P_k, disregarding the equations that don't involve infinitesimals, should be equal if the system is linearizable.

Step 11: Note that Q can consist of several systems. If *CaseSelect* = *all* is included in Options, then all cases leading to linearization are returned. By default, `MapDE` returns only one such case: $C = [1]$.

3.5.3 Proof of correctness of the MapDE Algorithm

Theorem 3.5.1 *Let R be a single input system in* RIF*-form consisting of leading linear equations with leaders of differential order ≥ 1, inequations, and with no leading nonlinear equations. Then Algorithm 7 converges in finitely many steps, and determines whether there exists a local holomorphic diffeomorphism $\hat{x} = \psi(x,u), \hat{u} = \phi(x,u)$ transforming R to a linear homogeneous target system*

$$\hat{R} : \mathscr{H}\hat{u}(\hat{x}) = 0 \tag{3.5.1}$$

In the case of existence the output RIF*-form consists of* DPS *of equations and inequations including those for the mapping function (ψ, ϕ).*

Proof We first note that Algorithm 7 converges in finitely many steps due to finiteness of each of the sub-algorithms used [40, 25, 9, 37]. To complete the proof we need to establish correctness of the two possible outcomes:
Case I: IsLinearizable = true and *Case II:* IsLinearizable = false.

Case I: IsLinearizable = true
Our task here is to show that given consistent input R, IsLinearizable = true and output Q then there exists a local holomorphc diffeomorphism Ψ to some linear system $\hat{R}$. To do this we build initial data for a solution of R, and initial data for solutions of Q. A complication is that these spaces have different independent variables x and (x, u). The assumption that all leaders for the RIF-form of R are of order ≥ 1 enables us to regard (x, u) as independent variables for Q. The inequations for Q include those for R together with the invertibility condition $\mathrm{DetJac}(\Psi) \neq 0$.

Suppose that the input RIF-form of R has differential order d_R and consists of equations and inequations with associated varieties $V^{=}(R)$, $V^{\neq}(R)$ in Jet space $J^{d_R}(\mathbb{C}^m, \mathbb{C}^n)$. So any point on the jet locus satisfies $(x, u^{(\leq d_R)}) \in V^{=}(R) \setminus V^{\neq}(R)$ in jet space $J^{d_R}(\mathbb{C}^m, \mathbb{C}^n)$ where $u^{(\leq d_R)}$ denotes the jet variables of total differential order $\leq d_R$. Let $\pi_0^{d_R} : J^{d_R} \to X$ be the projection of points in J^{d_R}, the jet space of order d_R, to the base space of independent variables $X \simeq \mathbb{C}^m$ where $x \in X$. The assumption that all leaders for R are of order ≥ 1 implies that $\pi_0^{d_R}(V^{=}(R) \setminus V^{\neq}(R)) = \mathbb{C}^m \setminus \pi_0^{d_R}(V^{\neq}(R))$.

When IsLinearizable = true there will be several systems P_k in the list of systems Q at Step 11 of Algorithm 7. We consider the case where there is only one such system, and without loss denote it by Q. For the case of several systems in Q we simply repeat the argument below for each such system. Suppose the system has differential order d_Q, and consists of equations and inequations for $v = (\xi, \eta, \psi, \phi, \hat{\eta})$ with associated varieties $V^{=}(Q)$, $V^{\neq}(Q)$ in $J^{d_Q}(\mathbb{C}^{m+n}, \mathbb{C}^{(2m+3n)})$, so that $((x, u), v^{(\leq d_Q)}) \in V^{=}(Q) \setminus V^{\neq}(Q)$ in $J^{d_Q}(\mathbb{C}^{m+n}, \mathbb{C}^{(2m+3n)})$. Then $\pi_0^{d_Q}(V^{=}(Q) \setminus V^{\neq}(Q)) = \mathbb{C}^{m+n} \setminus \pi_0^{d_Q}(V^{\neq}(Q))$.

Consider points $x_0 \in \mathbb{C}^m \setminus \pi_0^{d_R}(V^{\neq}(R))$ and $(x_0, u_0) \in \mathbb{C}^{m+n} \setminus \pi_0^{d_Q}(V^{\neq}(Q))$ belonging to the projections of R and Q onto their base spaces $X \simeq \mathbb{C}^m$ and $X \times U \simeq \mathbb{C}^{m+n}$. Then a family of initial data corresponding to all local analytic solutions u in a neighborhood of x_0 exists, and similarly for v. For R there exists a neighborhood $\mathscr{N}(x_0, u_0^{(\leq d_R)}) \subseteq V^{=}(R) \setminus V^{\neq}(R)$ in J^{d_R} and from Q there exists a neighborhood $\mathscr{N}((x_0, u_0), v_0^{(\leq d_Q)}) \subseteq V^{=}(Q) \setminus V^{\neq}(Q)$ in J^{d_Q}. The existence and uniqueness Theorems associated with RIF-form implies that for such analytic initial data there corresponds unique local analytic solutions and implies that there exists a local holomorphic diffeomorphism Ψ between neighborhoods mapping R to $\hat{R}$, and similarly between neighborhoods mapping Q to to $\hat{Q}$. Under this diffeomorphism the images of $\mathscr{N}(x_0, u_0^{(\leq d_R)})$ and $\mathscr{N}((x_0, u_0), v_0^{(\leq d_Q)})$ are $\hat{\mathscr{N}}(\hat{x}_0, \hat{u}_0^{(\leq d_R)})$ and $\hat{\mathscr{N}}((\hat{x}_0, \hat{u}_0), \hat{v}_0^{(\leq d_Q)})$ respectively.

To show that $\hat{R}$ is linear we consider the subsystem of Q for $\hat{\eta}$:

$$\mathscr{L}^{**} = \left\{ \sum_{\ell=1}^{m} \hat{\eta}^{\ell}(\hat{x}) \frac{\partial}{\partial \hat{u}^{\ell}} : \hat{\eta}^{j}_{\hat{u}^k} = 0, \mathscr{H}(\hat{\eta}) = 0 \right\} \tag{3.5.2}$$

where the linear system for $\hat{\eta}$ is in RIF-form and ultimately we will show that $\mathscr{H}(\hat{\eta})$ can be taken as $\hat{R}$. First we note the linear operator $\mathscr{H}$ cannot have any coefficients depending on $\hat{u}$. If not, and a coefficient did depend on a particular $\hat{u}^{\ell}$, then differentiating (3.5.2) with respect to $\hat{u}^{\ell}$ would yield a relation between parametric quantities, violating the freedom to assign values

independently to these parametric quantities. This would violate the RIF-form and its existence and uniqueness Theorem. So $\mathscr{H}$ only has coefficients depending on $\hat{x}$. As a remark we note that it is now easily verified that $\mathscr{L}^{**}$ generates an abelian Lie pseudogroup.

Exponentiating the infinitesimal symmetry (3.5.2) and applying its prolongation to a solution $\hat{u}(\hat{x})$ in $\hat{\mathscr{N}}(\hat{x}_0, \hat{u}_0^{(\leq d_R)})$ yields another solution in $\hat{\mathscr{N}}(\hat{x}_0, \hat{u}_0^{(\leq d_R)})$ given by $\tilde{u}(\hat{x}) = \hat{u}(\hat{x}) + \hat{\eta}$ where $\mathscr{H}(\hat{\eta}) = 0$. We have assumed in Step 9 of Algorithm 7 that HF(R) = HF(Q), which implies that all local analytic solutions in $\hat{\mathscr{N}}(\hat{x}_0, \hat{u}_0^{(\leq d_R)})$ are of form $\tilde{u}(\hat{x}) = \hat{u}(\hat{x}) + \hat{\eta}$ in $\hat{\mathscr{N}}(\hat{x}_0, \hat{u}_0^{(\leq d_R)})$. Consequently by a point change $\tilde{u}(\hat{x}) \to \tilde{u}(\hat{x}) - \hat{u}(\hat{x})$, $\hat{R}$ is equivalent to the linear homogeneous system $\mathscr{H}(\hat{\eta}(\hat{x})) = 0$. The RIF-form of Q includes the system for Ψ that determines mappings to $\hat{R}$ given by $\mathscr{H}(\hat{\eta}(\hat{x})) = 0$.

Case II: IsLinearizable = false

Suppose to the contrary that Algorithm 7 returns IsLinearizable = false, yet a local analytic linearization exists. Since the tests in Algorithm `PreEquivTest` in Step 3 of Algorithm 7 are all necessary conditions for a linearization to exist, they all test true.

Step 5 of Algorithm 7 applies RIF using binary splitting, partitioning the jet locus into disjoint cases; and the linearization must belong to some of these cases. By assumption, and the discussion above, there is diffeomorphism Ψ of R in some neighborhood $\mathscr{N}(x_0, u_0^{(\leq d_R)})$ in J^{d_R} to a linear system $\hat{R}$. Further the equations of the cases corresponding to linearization in terms of $v = (\xi, \eta, \psi, \phi, \hat{\eta})$ must have dimension d.

It cannot belong to a case of dimension $< d$, the ones discarded by the mindim = d option. Therefore by disjointness it must belong to one of the nc cases in P, say P_s. Therefore this case must fail the condition that HF(R) = HF(P_s) which is contrary to our assumption that such a linearization exist, completing our proof of correctness.

3.6 Examples

To illustrate the `MapDE` Algorithm 7 we consider some examples.

Example 3.6.1 *(Continuation and conclusion for Examples 3.3.1,3.3.2 and 3.3.3 using Algorithm 7.)*

The input is 3.3.5 which is in RIF*-form with respect to the orderly ranking* $u \prec u_x \prec u_{xx} \prec \cdots$, *together with the inequations* $u \neq 0, uu_x + x \neq 0, u^2 + x^2 \neq 0$ *or equivalently* $u(uu_x + x)(u^2 + x^2) \neq 0$. *This can be regarded as being derived from the leading linear* DPS *which results from multiplication by factors in its denominators.*

Step 1: *Set IsLinearizable := null. Here ID*(R) = $[u(x_0) = c_1, u_x(x_0) = c_2, u_{xx}(x_0) = c_3]$ *and* $\dim R = 3$, *subject to* $u(uu_x + x)(u^2 + x^2) \neq 0$.

Step 2: *See Example 3.3.1 for* S := RIF(*DetSys*($\mathscr{L}$)) *in 3.3.6, together with its ID*(S) *and* $\dim \mathscr{L} = \dim(S) = 4$. *See Example 3.3.3 and in particular 3.3.15 for* S' := RIF(*DetSys*($\mathscr{L}'$)) *which yields* $\dim \mathscr{L}' = \dim(S') = 3$.

Step 3: *Since* $\dim S = 4 \geq d + 1 = 4$ *and* $\dim S' = 3 \geq d = 3$, *the simplest necessary conditions for linearizability hold. Also* $\boldsymbol{d}(S) = \boldsymbol{d}(S') = \boldsymbol{d}(R) = 0$. *Application of Algorithm 5 for the LGMLinTest in Example 3.3.2 shows that R is linearizable, subject to* $u(uu_x + x)(u^2 + x^2) \neq 0$.

Step 4: *DetJac*$(\Psi) = \psi_x\phi_u - \psi_u\phi_x \neq 0$, $\hat{S}^\star := \left\{\hat{\xi} = 0, \hat{\eta}_{\hat{u}} = 0\right\}$ *and* $\hat{\mathscr{L}}^*$ *is replaced with* $\hat{\mathscr{L}}^\star$ *in 3.3.16 to yield:*

$$M_{BK}(\mathscr{L}', \hat{\mathscr{L}}^\star) = \left\{\hat{\xi}(\hat{x}, \hat{u}) = 0 = \xi\psi_x + \eta\psi_u, \hat{\eta}(\hat{x}, \hat{u}) = \xi\phi_x + \eta\phi_u\right\} \tag{3.6.1}$$

Evaluate $\hat{S}^\star$ *modulo* $\Psi : \hat{x} = \psi(x, u), \hat{u} = \phi(x, u)$ *to obtain* $\hat{S}^\star|_\Psi$. *This yields* $\hat{S}^\star|_\Psi = \left\{\hat{\xi} = 0, \psi_u\hat{\eta}_x - \psi_x\hat{\eta}_u = 0\right\}$. *Note that for brevity of notation we have replaced* $\hat{\xi}(\hat{x}, \hat{u})$ *with* $\hat{\xi}(x, u)$ *and* $\hat{\eta}(\hat{x}, \hat{u})$ *with* $\hat{\eta}(x, u)$. *Thus, the mapping system* $M = S' \cup \hat{S}^\star|_\Psi \cup M_{BK}(\mathscr{L}', \hat{\mathscr{L}}^\star) \cup \{DetJac(\Psi) \neq 0\}$ *is:*

$$\begin{aligned} M = [&\xi = -\frac{\eta\, u}{x}, \quad \eta_x = \frac{\left(u^2 + x^2\right)\eta}{xu^2} + \frac{\eta_u x}{u}, \\ &\eta_{u,u,u} = -\frac{\left(8\,u^8 + 8\,u^6x^2 + 8\,u^6 + 3\,u^2 + 3\,x^2\right)\eta}{\left(u^2 + x^2\right)u^3} + 3\,\frac{\eta_u}{u^2}, \\ &\hat{\xi} = 0, \quad \psi_u\hat{\eta}_x - \psi_x\hat{\eta}_u = 0, \quad \hat{\xi} = \xi\psi_x + \eta\psi_u, \\ &\hat{\eta} = \xi\phi_x + \eta\phi_u, \quad \psi_x\phi_u - \psi_u\phi_x \neq 0] \end{aligned} \tag{3.6.2}$$

Step 5: *Compute* $P :=$ RIF$(M, <, casesplit, mindim = d)$ *where* $d = 3$. *This results in* 3 *cases, two of which are rejected before their complete calculation since an upper bound in the computation drops below* $mindim = d = 3$. *The output for the single consistent case* P_1 *found is:*

$$\begin{aligned} P_1 = [&\xi = -\frac{\eta\, u}{x}, \eta_x = \frac{\left(u^2 + x^2\right)\eta}{xu^2} + \frac{\eta_u x}{u}, \\ &\eta_{u,u,u} = -\frac{\left(8\,u^8 + 8\,u^6x^2 + 8\,u^6 + 3\,u^2 + 3\,x^2\right)\eta}{\left(u^2 + x^2\right)u^3} + 3\,\frac{\eta_u}{u^2}, \\ &\phi_{x,x} = \frac{2\,\phi_{x,u}xu^2 - \phi_{u,u}x^2u + \phi_u u^2 + \phi_u x^2}{u^3}, \ \psi_x = \frac{\psi_u x}{u}, \\ &\hat{\eta} = -\frac{(u\phi_x - x\phi_u)\,\eta}{x}, \ \hat{\xi} = 0, \ x\phi_u - u\phi_x \neq 0, \ \psi_u \neq 0] \end{aligned} \tag{3.6.3}$$

Step 6: $P \neq \emptyset$ *contains* 1 *case.*

Step 7: *Initialize* $Q :=$ [ ]

Step 8: $k = nc = 1$

Step 9: *Also* $HF(R) = 1 + s + s^2$ *and the* ID *for* P_1 *yields* $HF(P_1) = 1 + s + s^2$. *So* $HF(R) = HF(P_1)$ *and the system is linearizable.*

Step 12: *We set* $Q := [P_1]$. *To extract the target we apply Algorithm 3.4.3 so that* $\hat{R}_1 :=$ *ExtractTarget*(Q_1) *which yields* $\hat{R}_1$ *in the form:*

$$\left(\frac{d}{d\hat{x}}\right)^3 \hat{u}(\hat{x}) = a_2(\hat{x})\left(\frac{d}{d\hat{x}}\right)^2 \hat{u}(\hat{x}) + a_1(\hat{x})\frac{\partial}{\partial\hat{x}}\hat{u}(\hat{x}) + a_0(\hat{x})\hat{u}(\hat{x}) \tag{3.6.4}$$

where $a_2(\hat{x})$, $a_1(\hat{x})$, $a_0(\hat{x})$ *are explicit expressions in* $(\hat{x}, \hat{u}, \hat{\psi}(\hat{x}, \hat{u}), \hat{\phi}(\hat{x}, \hat{u}))$ *and derivatives of* $\hat{\psi}(\hat{x}, \hat{u}), \hat{\phi}(\hat{x}, \hat{u})$.

Step 13: *The* ψ *system here is* $\psi_x = \frac{\psi_u x}{u}$. *Using* `Invariants` *from the* LAVF *package yields a single invariant* $x^2 + u^2$ *and so* $\psi = x^2 + u^2$. *Here and elsewhere the* `Invariants` *removes*

the need for us to specify arbitrary functions which would be the case if we started from the general solution of the ψ equation which is in this case $\psi = F(x^2 + u^2)$. Then substitution and solution of the ϕ equation then yields $\phi = G(x^2+u^2)x+H(x^2+u^2)$. The program specializes the arbitrary functions and constants to satisfy the inequations including the Jacobian condition, and in this case yields

$$\begin{aligned} \hat{x} &= \psi = x^2 + u^2 \\ \hat{u} &= \phi = x \end{aligned} \tag{3.6.5}$$

Substitution of 3.6.5 into the target 3.6.4 requires first inverting 3.6.5 using `Maple`*'s* `solve` *to give $x = \hat{\psi} = \hat{u}, u = \hat{\phi} = \left(\hat{x} - \hat{u}^2\right)^{1/2}$ yields it explicitly as:*

$$\left(\frac{d}{d\hat{x}}\right)^3 \hat{u}(\hat{x}) = -\frac{(\hat{x}+1)}{\hat{x}}\hat{u}(\hat{x}) \tag{3.6.6}$$

So far our work on the integration of the mapping equations to determine the transformations is preliminary and experimental. We have shown that the basic structure of the linear target can be determined implicitly. It remains to be seen how useful this would be in applications, where the mapping cannot be determined explicitly. Heuristic methods appear to be useful here, and we encourage the reader to try explore their own approaches.

From the output we also subsequently explored how far we could make the Target explicit before the integration of the map equations. In particular we exploited the transformation (as do [25]) that any such ODE is point equivalent to one with its highest coefficients (here a_2, a_1) being zero. This yields additional equations on ψ, ϕ and the target takes the very simple form:

$$\left(\frac{d}{d\hat{x}}\right)^3 \hat{u}(\hat{x}) = -\frac{8u^3(u^2 + x^2 + 1)}{(u^2 + x^2)\psi_u^3}\hat{u}(\hat{x}) \tag{3.6.7}$$

The RIF*-form of the system for ϕ, ψ is:*

$$\begin{aligned} \psi_x &= \frac{\psi_u x}{u} \\ \psi_{u,u,u} &= -1/2\,\frac{-3\,{\psi_{u,u}}^2 u^2 + 3\,{\psi_u}^2}{\psi_u u^2} \\ \phi_{x,x} &= 2\,\frac{x\psi_{u,u}\,(\phi_x u - \phi_u x)}{\psi_u u^2} + \frac{\phi_{u,u}x^2 u + \phi_u u^2 - 2\,\phi_x u x + \phi_u x^2}{u^3} \\ \phi_{x,u} &= \frac{\psi_{u,u}\phi_x u - \psi_{u,u}\phi_u x + \psi_u \phi_{u,u} x - \psi_u \phi_x}{\psi_u u} \end{aligned} \tag{3.6.8}$$

The general solution of the system is found by `Maple` *and yields the same particular solution as before for ψ, ϕ. It seems to have made a straightforward problem, more difficult!*

Example 3.6.2 *(Lyakhov, Gerdt and Michels Test Set) Lyakhov, Gerdt and Michels [25] consider the following test set of ODE of order d, for $3 \le d \le 15$:*

$$\left(\frac{d}{dx}\right)^d (u(x)^2) + u(x)^2 = 0 \tag{3.6.9}$$

By inspection this has the linearization for any d:

$$\Psi = \{\hat{x} = x,\ \hat{u} = u^2\} \tag{3.6.10}$$

They report times on an Intel(R)Xeon(R) X5680 CPU clocked at 3.33 GHz and 48GB RAM. All the following times are measured from the entry to the program. The times for detecting the existence of the linearization by the LGM Test in [25] range from 0.2 secs for $d = 3$ *to about 150 secs for* $d = 15$*. For comparison, we run MapDE with our own implementation of the LGMLinTest using* LAVF*. Our runs of the same tests to detect the existence of the linearization using on a 2.61 GHz I7-6600U processor with 16 GB of RAM range from 0.422 secs when* $d = 3$ *to 11.5 secs when* $d = 15$*.*
Their method from start to existence and then construction of the linearization (existence and construction), takes 7512.9 secs for $d = 9$ *and is out of memory for* $d \geq 10$*. In contrast, we report times for existence and construction that are only slightly longer than our existence times for* $3 \leq d \leq 15$*. For* $d = 3$ *to* $d = 15$ *we also report the time for our Hilbert test for existence of linearization and the total time for the existence and construction of the explicit linearization. Thus LGMTest time < Hilbert Test time < Existence and Construction. These results are displayed in Fig. 3.2 on a* $\log_{10}$ *axis.*

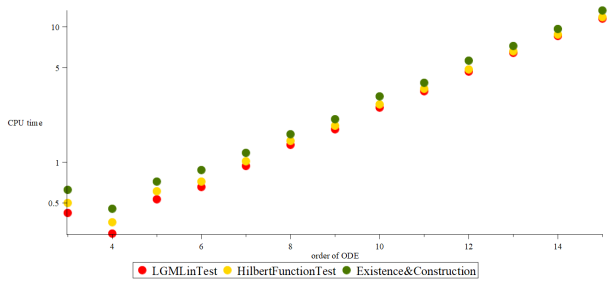


Figure 3.2: The graph represents the CPU times for $\left(\frac{d}{dx}\right)^d (u(x)^2) + u(x)^2 = 0$. Timings from $t = 0$ (start of `MapDE`) to the time to LGMLin linearization Existence confirmation (Red), time to Hilbert Existence confirmation (Yellow), time from $t = 0$ to existence and construction of the linearizing transformations (Green).

On this test, our approach appears to have more favorable memory behavior, which possibly is due to our equations being less nonlinear than those of Lyakhov et al. [25]. However, more testing and analysis are needed to make a reasonable comparison.

Example 3.6.3 *Consider Burger's equation, modeling the simplest nonlinear combination of convection and diffusion:*

$$u_{x,x} = u_t - uu_x$$

Using our algorithm `MapDE` *with* `TargetClass = LinearDE` *shows that it has finite dimensional Lie symmetry algebra with* $\dim \mathscr{L} = 5 < \infty$*. Thus by the preliminary equivalence test* `PreEquivTest`*, it is not linearizable by point transformation. However rewriting this equation in conserved form* $\frac{\partial}{\partial x}(u_x + \frac{1}{2}u^2) = \frac{\partial}{\partial t}u$ *implies that there exists v:*

$$v_x = u, \quad v_t = u_x + \frac{1}{2}u^2 \tag{3.6.11}$$

Applying `MapDE` *with* `TargetClass = LinearDE` *to 3.6.11 shows that this new system is linearizable, with the* RIF*-form of the* Ψ *system given by:*

$$\begin{aligned}&\phi_{u,u} = 0,\ \varphi_{u,u} = 0,\ \phi_{u,v} = -1/2\,\phi_u,\\ &\varphi_{u,v} = -1/2\,\varphi_u,\ \phi_{v,v} = -1/2\,\phi_v,\\ &\varphi_{v,v} = -1/2\,\varphi_v, \Upsilon_u = 0, \psi_u = 0, \Upsilon_v = 0, \psi_v = 0\end{aligned} \tag{3.6.12}$$

After integration this yields Ψ*:*

$$\hat{x} = \psi = x,\ \hat{t} = \Upsilon = t,\ \hat{u} = \varphi = u\exp\left(-\frac{v}{2}\right),\ \hat{v} = \phi = \exp\left(-\frac{v}{2}\right)$$

and the Target system $\hat{R}$ *is* $[\hat{u}_{\hat{x}} = -2\hat{v}_{\hat{t}}, \hat{v}_{\hat{x}} = -\hat{u}/2]$ *so that* $\hat{v}_{\hat{t}} = \hat{v}_{\hat{x}\hat{x}}$, $\hat{u}_{\hat{t}} = \hat{u}_{\hat{x}\hat{x}}$*. This implies that the original Burger's equation is apparently also linearizable, through the introduction of the auxiliary nonlocal variable v. This paradox is resolved in that the resulting very useful transformation is not a point transformation, since it effectively involves an integral. For extensive developments regarding such nonlocally related systems see [8].*

Example 3.6.4 *(Nonlinearizable examples with infinite groups) Consider the KP equation*

$$u_{x,x,x,x} = -6\,uu_{x,x} - 6\,{u_x}^2 - 4\,u_{x,t} - 3\,u_{y,y} \tag{3.6.13}$$

which has

$$\begin{aligned}ID(R) = \{&u(x_0, y, t) = F_1(y, t),\ u_x(x_0, y, t) = F_2(y, t),\\ &u_{x,x}(x_0, y, t) = F_3(y, t),\ u_{x,x,x}(u)(x_0, y, t) = F_4(y, t)\}.\end{aligned} \tag{3.6.14}$$

Applying `MapDE` *shows that the defining system S for symmetries* $\xi\frac{\partial}{\partial x} + \eta\frac{\partial}{\partial y} + \tau\frac{\partial}{\partial t} + \beta\frac{\partial}{\partial u}$*, has initial data which is the union of infinite data along the Hyperplane* $p = (x_0, y_0, t, u_0)$ *and finite initial data at the point* $z_0 = (x_0, y_0, t_0, u_0)$*:*

$$\begin{aligned}ID(S) = \{&\beta(p) = H_1(t), \beta_y(p) = H_2(t), \beta_{y,y}(p) = H_3(t)\}\\ &\cup\\ \{&\beta_x(z_0) = c_1, \beta_u(z_0) = c_2, \eta(z_0) = c_3, \eta_t(z_0) = c_4, \tau(z_0) = c_5, \xi(z_0) = c_6\}\end{aligned} \tag{3.6.15}$$

So both the KP equation and its symmetry system have infinite dimensional solution spaces: $\dim R = \dim S = \infty$ *since both have arbitrary functions in their data. However the KP equation has differential dimension* $\boldsymbol{d}(R) = 2$ *since there are a max of* 2 *free variables* (y, t) *in its initial data while its symmetry system has* $\boldsymbol{d}(S) = 1$ *since it has a max of one free variable in its data. Thus* $\boldsymbol{d}(S) = 1 < 2 = \boldsymbol{d}(R)$ *and by the* `PreEquivTest` *the KP equation is not linearizable by point transformation.*

Consider Liouville's equation $u_{x,x} + u_{y,y} = e^u$ *which we rewrite as a* DPS *using* `Maple`*'s function* `dpolyform`*. That yields* $v = e^u$ *and the Liouville equation in the form* $v_{x,x} = -v_{y,y} + v_x^2/v + v_y^2/v + v^2$*.* `MapDE` *determines that* $\dim\mathscr{L} = \infty = \dim R$*, and also that the Liouville equation is not linearizable by point transformation. Interestingly it is known that Liouville's equation is linearizable by contact transformation (a more general transformation involving derivatives). For extensive developments regarding such contact related systems see [8].*

Example 3.6.5 *Given that exactly linearizable systems are not generic among the class of nonlinear systems, a natural question is how to identify such linearizable models. Since linearizability requires large (e.g. ∞ dimensional) symmetry groups a natural approach is to embed a model in a large class of systems and seek the members of the class with the largest symmetry groups. Indeed in Wittkopf and Reid [35] developed such as approach. We now illustrate how this approach can be used here. For a nonlinear telegraph equation one might embed it in the general class of spatially dependent nonlinear telegraph systems*

$$v_x = u_t, \quad v_t = C(x,u)u_x + B(x,u) \tag{3.6.16}$$

where $B_u \neq 0, C_u \neq 0, B_x \neq 0, C_x \neq 0$. Then applying the `maxdimsystems` *algorithm available in* `Maple` *with dim $= \infty$ a quick calculation yields* 11 *cases, only* 4 *of which satisfy the dimension restriction which we further narrow by requiring restriction to those that have the greatest freedom in $C(x,u)$, $B(x,u)$. Integration yields the linearizable class:*

$$v_x = u_t, \quad v_t = \frac{1}{q_x u} f\left(\frac{u}{q_x}\right) u_x - \frac{q_{xx}}{q_x^2} f\left(\frac{u}{q_x}\right) \tag{3.6.17}$$

and the linearizing transformation

$$\hat{x} = \frac{u}{q_x}, \quad \hat{t} = v, \quad \hat{u} = q(x), \quad \hat{v} = t \tag{3.6.18}$$

Similarity, we can seek the maximal dimensional symmetry group for the normalized linear Schrödinger Equation

$$i\hbar\varphi_t = -\frac{\hbar^2}{2m}\nabla^2\varphi + V(x,y,t)\varphi \tag{3.6.19}$$

Restricting to 2 *space plus one time yields $V(x,y,t) = \omega(t)(x^2+y^2) + b(t)x + c(t)y + d(t)$, and satisfies the conditions for mapping to a constant coefficient DE via the methods of our previous paper [29].*

3.7 Discussion

In this paper we give an algorithmic extension of `MapDE` introduced in [29], that decides whether an input DPS can be mapped by local holomorphic diffeomorphism to a linear system, returning equations for the mapping in RIF-form, useful for further applications. This work is based on creating algorithms that exploit results due to Bluman and Kumei [4, 8] and some aspects of [25]. This is a natural partner to the algorithm for deciding the existence of an invertible map of a linear DPS to a constant coefficient linear DE given in our previous paper [29].

The mapping approach [25] for ODE explicitly introduces a target linear system $\hat{R}$ with undetermined coefficients, then uses the full nonlinear determining equations for the mapping and applies the `ThomasDecomposition` Algorithm [42, 37]. In contrast, like Bluman and Kumei, we exploit the fact that the target appears implicitly as a subalgebra of the Lie symmetry algebra $\mathscr{L}$ of R and avoid using the full nonlinear determining equations for the transformations. Unlike Bluman and Kumei, who depend on extracting this subalgebra by explicit

non-algorithmic integration, we use algorithmic differential algebra. We exploit the fact that the subalgebra corresponding to linear superposition appears as a subalgebra of the derived algebra $\mathscr{L}'$ of $\mathscr{L}$, generalizing the technique for ODE in [25]. Instead of using the BK mapping equations (3.3.12), [25] apply the transformations directly to the ODE. In contrast, our method works at the linearized Lie algebra level instead of the nonlinear Lie Group level used in [25] which may be a factor in the increased space and time usage for their test set compared to our timings given in Fig. 3.2.

Bluman and Kumei give necessary conditions in [8, Theorem 2.4.1] and sufficient conditions [8, Theorem 2.4.2] for linearization of nonlinear PDE systems with $m \geq 2$. Their requirement $\dim \mathscr{L} = \infty$ is dropped in our approach allowing us to deal with ODE and also linearization of overdetermined PDE systems. They also use the Jacobian condition to introduce a solved form of the BK equations with coefficients $\alpha^i_\sigma(x,u), \beta^\nu_\sigma(x,u)$ (see [8, Eq 2.69]), and further decompose the resulting system with respect to their $f^\sigma(\phi)$'s (our $\hat{\eta}$'s). This decomposition results for input PDE having no zero-order (i.e. algebraic) relations among the input systems of PDE, a condition that is not explicitly given in the hypotheses of their theorems. We are planning to take advantage of this decomposition in future work, as an option to `MapDE`, since it can improve efficiency, when applicable. For the more general case of contact transformations for $m = 1$, not considered here, see [8, Theorems 2.4.3-2.4.4].

An important aspect of Theorem 3.5.1 concerning the correctness of `MapDE`, is to show that the existence of an infinitesimal symmetry $\sum_\ell \hat{\eta}^\ell(\hat{x}) \frac{\partial}{\partial \hat{u}^\ell}$ where $\mathscr{H}(\hat{\eta}) = 0$, when exponentiated to act on a local analytic solution of $\hat{R}$ produces all local analytic solutions in a neighborhood. Showing this depends on showing $\mathrm{HF}(R) = \mathrm{HF}(P_k)$ in Step 9 of Algorithm 7 or in intuitive terms, the size of solution space of the input system is the same as the size of the symmetry subgroup corresponding to P_k. In contrast the statement and proof of [8, Theorem 2.4.2] appears to miss this crucial hypothesis about the size of the solution space of the input system being equal to the size of the solution space of the symmetry sub-group for linearization (as measured by Hilbert Series).

It is important to develop simple, efficient tests to reject the existence of mappings, based on structural and dimensional information. In addition to existing tests [25], [8, 1, 45] we introduced a refined dimension test based on Hilbert Series. We will extend these tests in future work. We note that the potentially expensive change of rankings needed by our algorithms (for example to determine the derived algebra when it is infinite dimensional) could be more efficiently accomplished by the change of rankings approach given in [11].

Mapping problems such as those considered in this paper are theoretically and computationally challenging. Given that nonlinear systems are usually not linearizable, a fundamental problem is to identify such linearizable models. For example [1, 45] use multipliers for conservation laws to facilitate the determination of linearization mappings. Wolf's approach [45] enables the determination of partially linearizable systems. Setting up such problems by finding an appropriate space to define the relevant mappings is important for discovering new non-trivial mappings. See Example 3.6.3 and [8] for such embedding approaches where the model is embedded in spaces that have a natural relation to the original space in terms of solutions but are not related by invertible point transformation. Another method is to embed a given model in a class of models and then efficiently seek the members of the class with the largest symmetry groups and most freedom in the functions/parameters of the class. Example

3.6.5 illustrates this strategy.

In this paper, we have emphasized algorithmic results by using algorithms involving a finite number of differentiations and eliminations. However, differential-elimination can often be expensive, and the early use of heuristic integration, can some times quickly yield results, which take longer by the algorithmic methods [44]. In particular, if we are seeking whether a system of PDE is linearizable, we can try to integrate its equations (for example its one term equations of the form derivative = 0). Then use the differential Hilbert function to pick out an appropriate subalgebra.

Example 3.7.1 *Consider the nonlinear diffusion equation*

$$u_t = u_x^{-2} u_{xx}$$

which is discussed in Olver [32, Example 6.47, pg 210]. The determining system for this example is

$$\tau_x = 0, \eta_x = 0, \tau_u = 0, \xi_{x,x} = 0, \eta_{t,t} = 0, \eta_{u,u} = 0,$$
$$\xi_{u,u} = \xi_t, \tau_t = 2\,\eta_u, \xi_{x,u} = -\eta_t/2, \xi_{x,t} = -\eta_{t,u}/2$$

where the first row, is a row of one term equations. Integation of the one term equations yields

$$\eta = (c_1\,u + c_2)\,t + c_3\,u + c_4, \tau = f\,(t)\,, \xi = g\,(t,u)\,x + h\,(t,u)$$

Substitution of this into the remaining equations and reduction using `rif` *yields:*

$$h_{u,u} = h_t, f_t = 2\,c_1\,t + 2\,c_3, g_t = -c_1/2, g_u = -c_1\,u/2 - c_2/2$$

where $f = f(t), g = g(t,u)$ and $h = h(t,u)$. Application of the differential Hilbert function shows that the only Lie symmetry sub-algebra that could lead to linearization corresponds to the infinite sub-algebra with generator: $h(t,u)\frac{\partial}{\partial x}$ where $h_{u,u} = h_t$.

In particular that sub-algebra has the same differential Hilbert function as the the original PDE $u_t = u_x^{-2} u_{xx}$. We introduced an option to `MapDE` *InputRDetSys, which when present in the optionlist of* `MapDE`*:*

$$InputRDetSys\ = [\xi_t = \xi_{uu}, \xi_x = 0, \tau = 0, \eta = 0]$$

yielded the linearizing transformation

$$\hat{x} = t, \quad \hat{t} = u, \quad \hat{u} = x$$

We invite the user to compare with Olver's example. This example shows that judicious integration may facilitate the determination of the mapping.

We provide a further integration phase to attempt to find the mappings explicitly, based on `Maple`'s `pdsolve` which will be developed in further work. Even if the transformations can't be determined explicitly, they can implicitly identify important features. Linearizable systems have a rich geometry that we are only beginning to exploit, such as the availability group action on the source and target. This offers interesting opportunities to use invariantized methods, such as invariant differential operators, and also moving frames [27, 17, 14, 3, 24]. Furthermore, they are available for the application of symbolic and symbolic-numeric approximation methods, a possibility that we will also explore. Finally a model that is not exactly linearizable may be *close* to a linearizable model or other attractive target, providing motivation for our future work on approximate mapping methods.

Acknowledgments

One of us (GR) acknowledges his debt to Ian Lisle who was the initial inspiration behind this work, especially his vision for Lie pseudogroups and an algorithmic calculus for symmetries of differential systems. GR and ZM acknowledge support from GR's NSERC discovery grant.

Our program and a demo file included some part of our computations are publicly available on GitHub at: https://github.com/GregGitHub57/MapDETools

Bibliography

[1] S. Anco, G. Bluman, and T. Wolf. Invertible mappings of nonlinear PDEs to linear PDEs through admitted conservation laws. Acta Applicandae Mathematicae, 101, 21–38, 2008. 35, 43, 64, 86, 94

[2] I. M. Anderson, and C. G. Torre. New symbolic tools for differential geometry, Gravitation, and Field theory. Journal of Mathematical Physics, 53, 2012. 24, 42, 93

[3] O. Arnaldsson. Involutive Moving Frames. PhD thesis, University of Minnesota, 2017. 24, 42, 65

[4] G. Bluman and S. Kumei. Symmetries and Differential Equations. Springer, 1989. 27, 50, 63

[5] G. Bluman and S. Kumei. Symmetry based algorithms to relate partial differential equations: I. Local symmetries. Eur. J. Appl. Math., 1, 189–216, 1990. 24, 46

[6] G. Bluman and S. Kumei. Symmetry based algorithms to relate partial differential equations: II. Linearization by nonlocal symmetries. Eur. J. Appl. Math., 1, 217–223, 1990. 24, 46, 52

[7] G. Bluman, and Z. Yang. Some Recent Developments in Finding Systematically Conservation Laws and Nonlocal Symmetries for Partial Differential Equations. Similarity and Symmetry Methods Applications in Elasticity and Mechanics of Materials, Ganghoffer, J.F. and Mladenov, I., 73, 1-59, 2014. 43

[8] G. W. Bluman, A. F. Cheviakov, and S. C. Anco. Applications of Symmetry Methods to Partial Differential Equations. Springer, 2010. 1, 6, 8, 11, 18, 23, 24, 27, 30, 31, 32, 50, 51, 62, 63, 64, 93, 94, 95

[9] F. Boulier, D. Lazard, F. Ollivier, and M. Petitot. Representation for the radical of a finitely generated differential ideal. Proc. ISSAC '95, 158–166, 1995. 12, 13, 42, 44, 49, 57

[10] F. Boulier, D. Lazard, F. Ollivier, M. Petitot. Computing representations for radicals of finitely generated differential ideals. Journal of AAECC, 2009. 12, 42, 44

[11] F. Boulier, F. Lemaire, and M. M. Maza. Computing differential characteristic sets by change of ordering. Journal of Symbolic Computation, 45(1), 124–149, 2010. 64, 94

[12] J. Carminati and K. Vu. Symbolic computation and differential equations: Lie symmetries. Journal of Symbolic Computation, 29, 95–116, 2000. 6, 24, 46, 87

[13] A. F. Cheviakov. GeM software package for computation of symmetries and conservation laws of differential equations. Computer Physics Communications, 176(1), 48–61, 2007. 6, 8, 24, 46, 87

[14] M. Fels, and P. Olver. Moving Coframes II. Regularization and theoretical foundations. Acta Applicandae Mathematicae, 55, 127–208, 1999. 23, 42, 65

[15] O. Golubitsky, M. Kondratieva, A. Ovchinnikov, and A. Szanto. A bound for orders in differential Nullstellensatz. Journal of Algebra, 322(11), 3852–3877, 2009. 24, 42

[16] S.L. Huang. Properties of Lie Algebras of Vector Fields from Lie Determining System. PhD thesis, University of Canberra, 2015. 43, 47

[17] E. Hubert. Differential invariants of a Lie group action: Syzygies on a generating set. Journal of Symbolic Computation, 44(4), 382–416, 2009. 24, 42, 65

[18] B. Kruglikov, and D. The. Jet-determination of symmetries of parabolic geometries. *Mathematische Annalen*, 371(3), 1575–1613, 2018. 42

[19] S. Kumei and G. Bluman. When nonlinear differential equations are equivalent to linear differential equations. SIAM Journal of Applied Mathematics, 42, 1157–1173, 1982. 11, 23, 24, 43

[20] M. Lange-Hegermann The Differential Counting Polynomial. Foundations of Computational Mathematics, 18(2), 291–308, 2018. 44

[21] F. Lemaire. Contribution à l'algorithmique en algèbre différentielle, Université des Sciences et Technologie de Lille - Lille I, 2002. 12, 44

[22] I. Lisle and S.-L. Huang. Algorithms calculus for Lie determining systems. Journal of Symbolic Computation, 482–498, 2017. 23, 24, 29, 43, 47, 51, 94

[23] I.G. Lisle, and G.J. Reid. Geometry and Structure of Lie Pseudogroups from Infinitesimal Defining Systems. J. Symb. Comput., 26(3), 355–379, 1998. 29, 43, 47, 51

[24] I.G. Lisle, and G.J. Reid. Symmetry Classification Using Noncommutative Invariant Differential Operators Found. Comput. Math., 6(3),1615-3375, 2006. 65

[25] D. Lyakhov, V. Gerdt, and D. Michels. Algorithmic verification of linearizability for ordinary differential equations. In Proc. ISSAC '17, ACM, 285–292, 2017. 11, 23, 28, 35, 43, 48, 49, 52, 56, 57, 60, 61, 63, 64, 93, 94

[26] F.M. Mahomed, and P.G.L. Leach. Symmetry Lie Algebra of nth Order Ordinary Differential Equations. J. Math. Anal. Appl., 1151: 80-107, 1990. 48

[27] E. Mansfield. A Practical Guide to the Invariant Calculus. Cambridge Univ, Press, 2010. 24, 42, 65

[28] A.V. Mikhalev, A.B. Levin, and E.V. Pankratiev, and M.V. Kondratieva. Differential and Difference Dimension Polynomials. Mathematics and Its Applications, Springer Netherlands, 2013. 44, 46

[29] Z. Mohammadi, G. Reid, and S.-L.T Huang. Introduction of the MapDE algorithm for determination of mappings relating differential equations. Proceedings of International Symposium on Symbolic and Algebraic Computation (ISSAC'19), ACM, 331–338, 2019. 41, 42, 43, 49, 50, 51, 63, 93

[30] S. Neut, M. Petitot, and R. Dridi. Élie Cartan's geometrical vision or how to avoid expression swell. Journal of Symbolic Computation, 44(3), 261 – 270, 2009. Polynomial System Solving in honor of Daniel Lazard. 23, 42

[31] P. Olver. Application of Lie groups to differential equations. Springer-Verlag, 2nd edition, 1993. 4, 25, 30, 44, 73, 86

[32] P. Olver. Equivalence, invariance, and symmetry. Cambridge University Press, 1995. 2, 3, 4, 9, 23, 27, 42, 48, 50, 52, 65, 93

[33] G.J. Reid. Finding abstract Lie symmetry algebras of differential equations without integrating determining equations. European Journal of Applied Mathematics, 2, 319–340, 1991. 11, 17, 23, 43

[34] G.J. Reid, I.G Lisle, A. Boulton, and A.D. Wittkopf. Algorithmic determination of commutation relations for Lie symmetry algebras of PDEs. Proc. ISSAC '92, 63–68, 1992. 43, 47, 51

[35] G.J. Reid, and A.D. Wittkopf. Determination of Maximal Symmetry Groups of Classes of Differential Equations. Proc. ISSAC '2000, 272-280, 2000. 63

[36] C. Riquier. Les systèmes déquations aux dérivées partielles. Paris: Gauthier-Villars, 1910. 44

[37] D. Robertz. Formal Algorithmic Elimination for PDEs. Lect. Notes Math. Springer, 2121, 2014. 12, 42, 44, 49, 57, 63

[38] T. Rocha Filho and A. Figueiredo. SADE: A Maple package for the symmetry analysis of differential equations. Computer Physics Communications, 182(2), 467–476, 2011. 6, 24, 46, 87

[39] C.J. Rust, and G.J. Reid. Rankings of partial derivatives. Proc. ISSAC '97, 9–16, 1997. 13, 29, 44

[40] C.J. Rust, G.J. Reid, and A.D. Wittkopf. Existence and uniqueness theorems for formal power series solutions of analytic differential systems. Proc. ISSAC '99, 105–112, 1999. 12, 13, 29, 42, 44, 49, 57

[41] W. Seiler. Involution: The formal theory of differential equations and its applications in computer algebra, Algorithms and Computation in Mathematics. 24, Springer, 2010. 13, 15, 25, 28, 42, 44, 73, 75, 81

[42] J.M. Thomas. Differential Systems. AMS Colloquium Publications, XXI, 1937. 12, 44, 49, 63

[43] F. Valiquette. Solving local equivalence problems with the equivariant moving frame method. SIGMA: Symmetry Integrability Geom. Methods Appl., 9, 2013. 23, 42

[44] T. Wolf. Investigating differential equations with crack, liepde, applsymm and conlaw. Handbook of Computer Algebra, Foundations, Applications, Systems, 37, 465–468, 2002. 6, 7, 8, 35, 43, 65, 87

[45] T. Wolf. Partial and Complete Linearization of PDEs Based on Conservation Laws. Differential Equations with Symbolic Computations,Trends in Mathematics, Zheng, Z. and Wang, D., 291-306, 2005. 43, 64, 94

Index

M_{BK}, Bluman-Kumei mapping equations, 49
$\mathscr{I}$, An infinite set of data, 45
$\mathscr{P}$, A set of parametric derivatives, 45
DetSys, Determining system, 48
DetJac, Determinant of the Jacobian, 42
HF,Differential Hilbert Function, 45
dord, Differential Order, 45
free, The number of free variables in $\mathscr{I}$ set, 46
LGMLinTest, Algorithm5, 49
ExtractTarget, Algorithm 3.4.3, 53
DPS, Differential Polynomial System, 41
ID, Initial data, 46
LAVF, LieAlgebrasOfVectorFields Maple package, 47
`MapDE`, Algorithm 7, 55
ODE, Ordinary Differential Equation, 41
PDE, Partial Differential Equation, 41
`PreEquivTest`, Algorithm 6, 53
DEC, Differential-elimination completion algorithms, 42
d, Differential dimension, 46

Chapter 4

A completion algorithm for approximate real differential polynomial systems to Involutive Form

Real numerical algebraic geometry is a new approach in which each connected component of a real algebraic set is represented by at least one point. Such points, called critical points, are efficiently computed by Numerical Homotopy continuation methods [11, 38]. We extend this idea to systems of differential polynomial equations where differentiation (prolongation) of the system is required to identify their hidden constraints (integrability conditions) to yield existence and uniqueness criteria for their power series solutions.

This work is part of a sequence focused on developing a numeric-symbolic algorithm to analyze a system of differential polynomial equations in the framework of the classical Jet geometry. We apply the results of Wu et al. [41] to determine critical points on the jet variety of the prolongation of the system. In our algorithm, we combine the penalty-based critical point method with the geometric involutive form approach to address the instability of the exact completion algorithms where the system contains real approximate coefficients. We remove the regularity conditions assumed in previous work [38].

Some applications related to approximate defining systems are given. We also discuss the potential of application to approximate symmetry by an example.

4.1 Introduction

There is a well-known isomorphism between polynomials and constant coefficient linear homogeneous partial differential equations.

$$x^i \leftrightarrow \frac{\partial}{\partial x^i}$$

See Gerdt et al. [10]. In fact, studying differential polynomial systems is much more challenging than polynomial systems.

Systems of differential equations can be over or under-determined, and also contain algebraic constraints. Increasingly, such systems arise from mathematical modeling of engineering

and science problems such as multibody mechanics, electrical circuit design, etc. These systems require a finite number of prolongations (differentiations) and projections (eliminations) to determine their hidden constraints and determine all their integrability conditions. Much progress has been made in terms of exact differential elimination methods to get the involutive form of a differential system. See Seiler [32], Boulier [4], Chen and Gao [9], Hubert [12], Mansfield [16], Reid and Rust [30]. Often, these methods are not stable in the approximate case. The coordinate dependency on ordering the variables that are used in these methods can lead to numerical instability, especially in approximate systems.

The failing of exact differential elimination algorithms to get the involutive form of real approximate differential polynomial systems (DPS) motivates our interest in real Numerical Jet Geometry methods. In fact, numerical algebraic geometry methods are combined with the geometric prolongation methods to address numerical instability in these methods.

Complex numerical algebraic geometry was pioneered by Sommese et al. [33, 2]. For complex varieties, they described the solution components of an algebraic equation by slicing the solution set with appropriate random planes. The intersection points are known as witness points. The obvious extension of the complex approach to a real case will fail, since such random planes may not intersect some components at real points. For example, let $f(u, v) = u^2 + v^2 - 1 = 0$. Then a random real line $au + bv + c = 0$ can miss the variety of f with high probability. Several algorithms have been proposed to compute real solutions of algebraic equations [15, 3]. The efficient method that we use is calculating real witness points (critical points) by setting up an optimization problem [38, 11]. In these methods Lagrange multipliers are used to optimize the distance from a random (plane) point to the real variety of an algebraic system. Later, Wu, Chen, and Reid improved their method in [41].

Directly applying these point-based algebraic methods to differential systems in Jet space is not geometrically correct. Although these points belong to certain manifolds in jet space, they may not belong to the solutions of the differential equation. By differentiation, the graphs of solution can be embedded as curves lying in the jet variety. The union of these graphs yields a solution variety $\mathscr{V}_{\text{sol}}$. We have $\mathscr{V}_{\text{sol}} \subseteq \mathscr{V}$ in general and whenever $\mathscr{V}_{\text{sol}} = \mathscr{V}$ the differential equation is said to be locally solvable. For details concerning jet geometry see [19, 32].

In this article, we implement a symbolic-numeric completion algorithm for differential polynomial systems that contain real approximate coefficients. We use numerical homotopy methods to determine real jet witness points on the jet variety of the prolongations of differential systems. In particular, we combine aspects of Cartan-Kuranishi theory for partial differential equations with the result of Wu et al. [41] to identify missing constraints of approximate nonlinear polynomial differential systems raising from applications. We implement our algorithm in the symbolic computation language `Maple` with an interface to `Bertini`, which is an efficient numerical homotopy-based polynomial solver written by Bates and his collaborators [2].

Some basic material is reviewed in section §4.2. We provide detailed descriptions of the algorithm in section §4.3. In §4.4, we give examples of the algorithm and conclude in §4.5.

4.2 Mathematical Background

Suppose a system of differential polynomial equations $R = (R^1, \ldots, R^\ell) = 0$ over a field $\mathbb{F}$ ($\mathbb{C}$ or $\mathbb{R}$) with n independent variables $x = (x^1, x^2, \ldots, x^n)$ and m dependent variables $u = (u^1, \ldots, u^m)$. The operator of formal total derivation is:

$$D_{x^j} = \frac{\partial}{\partial x^j} + \sum_{\ell=1}^{m} u^\ell_{x^j} \frac{\partial}{\partial u^\ell} + \cdots$$

If we consider $\underset{i}{u}$ as the ith order derivatives of u, then the jet variety of a q-th order system in $J^q(\mathbb{F}^n, \mathbb{F}^m) \approx \mathbb{F}^{N(n,m,q)}$ is

$$\mathscr{V}(R) = \left\{ (x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q}{u}) \in J^q : R^i(x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q}{u}) = 0, \ for\ i = 1, \ldots, \ell \right\} \tag{4.2.1}$$

where $R^i : J^q \approx \mathbb{F}^{N(n,m,q)} \to \mathbb{F}$, and $N(n, m, q) = n + m \begin{pmatrix} n+q \\ q \end{pmatrix}$ is the number of jet variables of order less than or equal to q.

Example 4.2.1 (Pendulum curtain [37]) *Consider the dynamics of the curtain made of many pendula hanging under constant gravity g as shown in Fig. 4.1. In the limit as the number of pendula becomes infinite, we obtain a continuous curtain modeled by*

$$X_{t,t} + \Lambda X = \mathbb{K} X_{s,s}, \quad Y_{t,t} + \Lambda Y + g = \mathbb{K} Y_{s,s}, \quad X^2 + Y^2 = 1 \tag{4.2.2}$$

where X, Y and the Lagrange multiplier Λ, are functions of s, t. The Jet variety of (4.2.2) is a 17 dimensional submanifold of $\mathbb{F}^{20}$.

$$\begin{aligned} \mathscr{V}(R) = \{ & (t, s, X, Y, \Lambda, X_t, Y_t, \Lambda_t, X_s, Y_s, \Lambda_s, X_{t,t}, Y_{t,t}, \Lambda_{t,t}, X_{t,s}, Y_{t,s}, \Lambda_{t,s}, X_{s,s}, Y_{s,s}, \Lambda_{s,s}) \in J^2 : \\ & X_{t,t} + \Lambda X = \mathbb{K} X_{s,s},\ Y_{t,t} + \Lambda Y + g = \mathbb{K} Y_{s,s},\ X^2 + Y^2 = 1 \} \end{aligned}$$

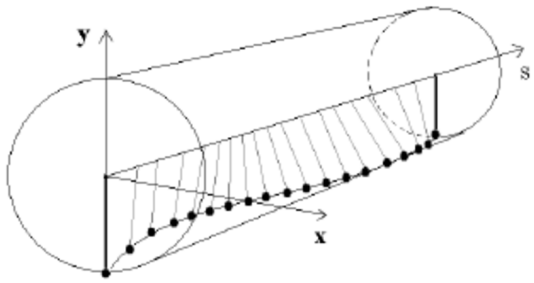


Figure 4.1: Curtain Pendulum([37])

This system similar to most differential systems arising from applications, contains missing constraints. The key to determining these missing constraints lies in prolonging the system to

get the involutive form of the system. Generally, uncovering all hidden constraints is needed for finding exact or numeric solutions of a differential system since the initial value problem (IVP) for its solution should justify their constraints.

Note: There are different methods of treating this example: a differential-geometric approach [21], an approach using *Gröbner* bases [20], and an approach using characteristic sets [34]. Obviously, one can remove the constraint by using polar coordinates in this example, but for multi-body systems this may not be possible.

A single prolongation of the Jet variety of the qth order system denoted by $D(R)$ is locally the list of first order total derivatives of R^i , $i = 1, \cdots, \ell$ with respect to all independent variables:

$$D(R) = \left\{ (x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q+1}{u}) \in J^{q+1} : R = 0, \;\; D_{x^j} R^i = 0; \; i = 1, \cdots, \ell \right\}$$

It extends the locus of a system from lower order to higher order Jet space.

A single projection maps J^q to J^{q-1} is locally defined as:

$$\pi(R) = \left\{ (x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q-1}{u}) \in J^q : R(x, \underset{0}{u}, \underset{1}{u}, \ldots, \underset{q}{u}) = 0 \right\}.$$

Multiple projections $\pi^r : J^q \to J^{q-r}$ are defined iteratively.

The completion of nonlinear partial differential equations to the involutive form is based on the Cartan-Kuranishi algorithm [32, 25]. In general, the algorithm prolongs and projects the differential system until no new constraints are found. The final system is involutive, and the algorithm has terminated when the symbol of the q-order system is involutive.

Definition 4.2.1 (Symbol [5]) *The symbol of a qth order system is locally the linearized highest order part of the system, which is given by the matrix:*

$$\text{Symbol } R := \frac{\partial R}{\partial \underset{q}{u}} \tag{4.2.3}$$

The symbol of a qth order system R is involutive if

$$\text{rank Symbol}(DR) := \sum_{j=1}^{n} j \beta_j^{(q)} \tag{4.2.4}$$

and the $\beta_j^{(q)}$ are the dimensions of certain subspaces of the Null Space of the Symbol of R. See [32] for more details.

Definition 4.2.2 *The system R is said to be involutive at order k and projected order r, if $\pi^r D^k R$ is involutive. Equivalently, it satisfies*

$$\dim \; \pi^r D^k R = \dim \; \pi^{r+1} D^{k+1} R$$

and the symbol $\pi^r D^k R$ is involutive.

Symbolic differential elimination compilations algorithms such as `rifsimp` use a finite number of exact differentiation and eliminations to reduce an over-determined system of differential equations to involutive form and uncover all hidden constraints. See [23, 42]. For example, the Pendulum system (4.2.2) when $\mathtt{K} = 0$ has two missing constraints identified by differentiating the constraint twice.

$$X X_{t,t} + Y Y_{t,t} + X_t^2 + Y_t^2 = 0, \quad X X_t + Y Y_t = 0 \tag{4.2.5}$$

Definition 4.2.3 (Constant Rank Conditions [5]) *Suppose a qth order system* $R(x, \underset{q}{v}) = 0$ *with independent variables* x *and jet variables* $\underset{q}{v} = (u, \underset{1}{u}, \cdots, \underset{q}{u})$ *corresponding to dependent variables and their derivatives. This system satisfies the constant rank conditions at* $(x^0, \underset{q}{v^0}) = (x^0, u^0, \underset{1}{u^0}, \cdots, \underset{q}{u^0}) \in \mathscr{V}(R) \subseteq J^q$ *if there exist nonzero constants* α, β *such that*

$$rank \frac{\partial R(x, \underset{q}{v})}{\partial (x, \underset{q}{v})} = \alpha = rank \frac{\partial R}{\partial \underset{q}{v}}, \quad rank \frac{\partial R}{\partial \underset{q}{u}} = \beta$$

in a neighborhood of $(x^0, \underset{q}{v^0})$ *in the usual Euclidean norm. We call* $(x^0, \underset{q}{v^0})$ *Euclidean point.*

Traditionally, most symbolic completion algorithms apply coordinate dependent differential elimination methods. In particular, triangular decomposition or Gröbner basis techniques are generalized to differential equation systems in such methods. The dependency on ordering of variables often leads to numerical instability like the numerical instability of ordered Gaussian Elimination. However, geometric approaches such as representing the variety via certain points, give an independent coordinate description of properties of systems in Jet space. Such coordinate independent methods can have greater numerical stability since they do not pivot on small leading quantities.

4.2.1 Symbolic-Numeric Method for Linear Homogeneous DPS

The Singular Value Decomposition of Numerical Linear Algebra is a powerful tool in implementing the symbolic-numeric algorithm for completion of linear differential systems. Reid and his collaborators [5] apply this technique in their symbolic-numeric methods for general linear systems of differential equations. An early application of this approach appears in the problem of Camera Pose determination, which is an essential problem in computer vision [22]. We give a brief overview of their hybrid symbolic-numeric approach.

Any linear homogeneous polynomial differential system R with constant coefficients and differential order q can be written in the matrix form, with an identical representative matrix A^q

$$A^q(x) \underset{q}{v} = \mathbf{0}, \quad \underset{q}{v} = \begin{pmatrix} \underset{q}{u} \\ \vdots \\ \underset{1}{u} \\ u \end{pmatrix} \tag{4.2.6}$$

where $x = (x^1, \ldots, x^n)$ and $\underset{q}{v}$ is a column vector of all partial derivatives of order $\leqslant q$.
The singular value decomposition technique computes the null spaces of these polynomial matrices. This construction is based on characterizing the constant rank conditions (4.2.3).

The sequence of linear homogeneous matrix systems:

$$A^q(x)\underset{q}{v} = \mathbf{0}, \quad A^{q+1}(x)\,\underset{q+1}{v} = \mathbf{0}, \quad A^{q+2}(x)\,\underset{q+2}{v} = \mathbf{0}, \quad \cdots \tag{4.2.7}$$

yield the prolongations of the system R, DR, D^2R, $\cdots$ respectively. The right hand side of each system is a zero vector with the appropriate dimension.
As $rank\ A = rank\ A(x^0)$ for any generic point $x^0 \in \mathbb{F}^n$, a random point $x = x^0$ is chosen to avoid degenerate ranks of the matrix systems. This yields a sequence of constant matrix systems:

$$A^q(x^0)\underset{q}{v} = \mathbf{0}, \quad A^{q+1}(x^0)\,\underset{q+1}{v} = \mathbf{0}, \quad A^{q+2}(x^0)\,\underset{q+2}{v} = \mathbf{0}, \quad \cdots \tag{4.2.8}$$

In the next step, the projected system $\pi^r D^k R$, where $r = 0, 1, \cdots, k$ are constructed for each prolongation, $D^k R$. The projection π on vector $\underset{q}{v} \in J^q$ which is presented by $\pi\underset{q}{v} \in J^{q-l}$ is obtained by deleting coordinates in $\underset{q}{v}$ of order q.

$$\pi^r R = \left\{ \pi^r \underset{q}{v} \in J^{q-l} : \quad A^q(x^0)\underset{q}{v} = \mathbf{0} \right\} \tag{4.2.9}$$

The key tool in this approach is the singular value decomposition. After each symbolic prolongation, the singular value decomposition (SVD) of the related matrix is computed at a random point, x^0:

$$A^{q+k}(x^0) = U\Sigma V^t$$

where U is a unitary matrix, V^t an orthogonal matrix, and Σ a diagonal matrix whose diagonal entries are known as singular values. The number of nonzero singular values represents the numerical rank of $A^{q+k}(x^0)$. A basis for the Null space of $A^{q+k}(x^0)$ is a sub-matrix of V^t obtained by deleting the first r rows of V^t. It yields an estimate for dim $(D^k R)$. An approximate basis for the kernel of the prolonged matrix is obtained by considering user input tolerance and setting singular values below it to zero. For the projection step, deleting coordinates corresponding to highest order derivatives from the basis for $D^k R$ yields a spanning set for $\pi D^k R$, which can be converted to an orthogonal basis by using the singular value decomposition again. We apply this idea in our main algorithm (9).

4.2.2 Critical Point Approach for Nonlinear PDS

The zero set of a differential polynomial system of equations in Jet space can be considered as an algebraic variety. We are interested in describing the set of real solutions of a differential polynomial system R of order q,

$$\mathscr{V}(R) \subseteq \mathbb{R}^n$$

arising from applications.

Sommese et al. [33] pioneered complex numerical algebraic geometry which is an approach that represents the irreducible components of an algebraic variety by certain complex points, called witness points. Witness points are intersections of irreducible components with a random linear space. Directly applying such point-based methods to differential equations in Jet space is not correct. Wu, Reid, and Golubitsky [39] extend these algebraic concepts to differential equations. Finding such witness points in a variety R relies on the below property:

$$\pi^r \mathscr{V}(D^k(R_{involv})) = \mathscr{V}(R_{involv}) \subseteq \mathscr{V}(R);\ \ for\ any\ r,k \in \mathbb{N}$$

where R_{involv} is an involutive form of R. Since R_{involv} contains all integrability conditions (hidden constrains) there is a power series solution of system R constructable order by order at witness points in $\mathscr{V}(R_{involv})$. Some progress has been made towards a numeric involutive form based on complex witness points (witness Jet points) in the complex case [39, 36].

Recent progress in numerical approaches for real algebraic equations developed by Wu, Chen, and Reid [41] motivates us to extend their remarkable algorithm to a system of differential polynomial equations, and especially to approximate systems where symbolic algorithms are not successful. Rouillier et al. [28] pioneered the idea of finding at least one real witness point, called critical point, on each connected component of an algebraic set. In particular, critical points on a real variety are obtained by constructing an optimization problem for the distance of a random point (not on the variety) to components of the variety. The critical point method is currently under intense development. Our work is more related to methods introduced by Wu et al. [38] and Hauenstein [11]. The algorithm in Wu et al. [38] works under regularity conditions, whereas Hauenstein's algorithm works with input that is pure dimension. Later, Wu and his collaborators introduced a new method that does not require the previous restrictions. Their algorithm has three stages. First, embedding the system of algebraic equations into a higher-dimensional space. Second, finding at least one approximate critical point on each connected component. Finally, refining the points by a homotopy continuation method. See [41] for more details.

Our first algorithm, presented below, is based on the critical point approach introduced by Wu et al. [41] for polynomial systems. We partition the input system into linear and nonlinear parts to reduce the number of the paths followed by numeric homotopy solvers. We apply this algorithm to our main algorithm.

The key idea in Wu et al. method [41] is using slack variables and embedding the system into higher-dimensional space. Indeed, this idea helps remove the singularity. See the following example.

Example 4.2.2 *Let* $R = \{v^2-(1-u^2)^3 = 0\}$. *Its augmented system is* $\tilde{R} = \{v^2-(1-u^2)^3+z_1 = 0\}$. *See Fig. 4.2. The Jacobian matrix*

$$\frac{\partial \tilde{R}}{\partial(u,v,z_1)} = \begin{pmatrix} 6u(1-u^2)^2 \\ 2v \\ 1 \end{pmatrix}$$

has full rank at any point $(u,v,z_1) \in \mathscr{V}_{\mathbb{R}}(\tilde{R})$. *Meanwhile, each point of* $\mathscr{V}_{\mathbb{R}}(\tilde{R})$ *approaches* $\mathscr{V}_{\mathbb{R}}(R)$ *when the norm of the* z_1, *slack variable, approaches zero.*

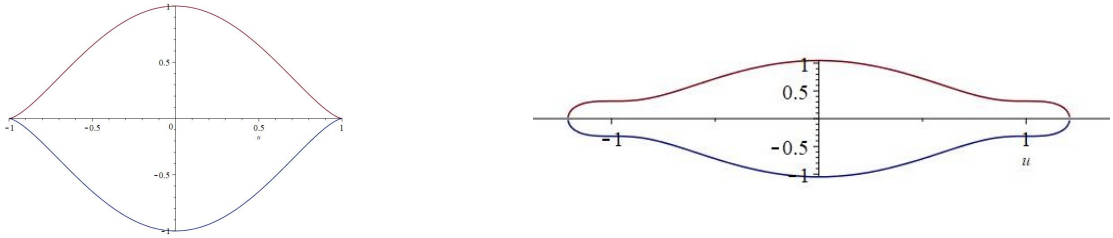


Figure 4.2: $v^2 - (1-u^2)^3 = 0$ (left), $v^2 - (1-u^2)^3 + z_1 = 0$ (right)

Lemma 4.2.4 *[41] Let $F = \{f^1, \ldots, f^k\}$ be a set of polynomials with $u = (u^1, \ldots, u^m)$. The Jacobian matrix of corresponding augmented system $\tilde{F} = \{f^1 + z_1, \ldots, f^k + z_k\}$ of F with variables $(u^1, \ldots, u^m, z_1, \ldots, z_k)$ has rank k at any point of smooth submanifold $\mathscr{V}_{\mathbb{R}}(\tilde{F})$ of $\mathbb{R}^{m+k}$ with dimension m. Also, points of each point of $\mathscr{V}_{\mathbb{R}}(\tilde{F})$ approaches $\mathscr{V}_{\mathbb{R}}(F)$ when the norm of the slack variables $z = (z_1, \ldots, z_k)$ approaches zero.*

We present the subroutine CritcalPoints of our main algorithm for finding critical points on a variety which is the extension of Wu et al. [41].

Algorithm 8 CritcalPoints

Input: A system of polynomials $F = \{f^1, \cdots, f^k\}$ in the variables $u = (u^1, \cdots, u^m) \in \mathbb{R}^m$ with real coefficients

Output: At least one critical point on each connected component of $\mathscr{V}_{\mathbb{R}}(F)$

1: Set $G := \{f \in F;\ f \text{ is Linear}\}$ and the nonlinear equations $H := \{F \setminus G\}$
2: Compute $G_{sol} := solve(G)$
3: Simplify the nonlinear part wrt G, $H := subs(G_{sol}, H)$
4: $\ell := nops(H)$
5: $u := u \cap \texttt{indets}(H)$
6: $n := nops(u)$
7: Perturb H to construct the augmented system
$$\tilde{H} := \{\tilde{h}^i = h^i + z_i;\ \ \forall\ i \text{ with } h^i \in H\}$$
8: Choose a random point $a = (a^1, \cdots, a^n) \notin \mathscr{V}_{\mathbb{R}}(H)$
9: Formulate the optimization problem (4.2.10) to find the minimum distance from the random point a to $\mathscr{V}_{\mathbb{R}}(H)$:
$$F_1 := \{\mathbf{u} + \beta\, \mathrm{J}^t\, \mathbf{h} - \mathbf{a} = \mathbf{0}\}$$
10: u^* := solution of F_1 found by numeric solver ▷ (`Bertini` or `Hom4Ps2`)
11: Refine the solution u^* by substituting $t = \frac{1}{\beta}$ in F_1 and constructing a homotopy system:
$$F_2 := \{t(\mathbf{u} - \mathbf{a}) + \mathrm{J}^t\, \mathbf{h} = \mathbf{0}\}$$
12: H_{CP} := solve the homotopy system F_2 ▷ (`Bertini` or `Hom4Ps2`)
13: CriticalPoints := $G_{sol} \cup H_{CP}$

Abbreviations used above: β : Penalty factor:: parameter, J : Jacobian

In line 7 of the CritcalPoints algorithm, we formulate the optimization problem for the augmented system of the nonlinear part, $\tilde{H} = \{\tilde{h}^i = h^i + z_i;\ \ \forall\ i \text{ with } h^i \in H\}$ with penalty

function $\beta\ (z_1^2 + \cdots + z_\ell^2)/2$ to extremize the distance from $\mathscr{V}_{\mathbb{R}}(H)$ to the random point $a = (a^1, \cdots, a^n) \notin \mathscr{V}_{\mathbb{R}}(H)$. In particular, let

$$\begin{gathered} \mu(u, z, \beta, a) = (\beta(z_1^2 + \cdots + z_\ell^2) + \sum_{i=1}^{n} (u^i - a^i)^2)/2 \\ \textit{then} \quad u^* = \min\ \mu(u, z, \beta, a)|_{\tilde{H}=0} \end{gathered} \tag{4.2.10}$$

So Lagrange multiplier techniques $\nabla\mu = \lambda\nabla\tilde{H}$ yield:

$$\begin{pmatrix} u^1 - a^1 \\ \vdots \\ u^n - a^n \\ \beta\, z_1 \\ \vdots \\ \beta\, z_\ell \end{pmatrix} = \begin{pmatrix} \frac{\partial h^1}{\partial u^1} & \cdots & \frac{\partial h^\ell}{\partial u^1} \\ \vdots & \ddots & \vdots \\ \frac{\partial h^1}{\partial u^n} & \cdots & \frac{\partial h^\ell}{\partial u^n} \\ 1 & & \\ & \ddots & \\ & & 1 \end{pmatrix} \begin{pmatrix} \lambda_1 \\ \vdots \\ \lambda_\ell \end{pmatrix} \tag{4.2.11}$$

Substitute $\lambda_i = -\beta\, z_i = -\beta\, h^i\ for\ i = 1, \ldots, \ell$ yields the square system:

$$\begin{pmatrix} u^1 \\ \vdots \\ u^n \end{pmatrix} + \beta\ (\text{Jacobian } H)^t \begin{pmatrix} h^1 \\ \vdots \\ h^\ell \end{pmatrix} = \begin{pmatrix} a^1 \\ \vdots \\ a^n \end{pmatrix} \tag{4.2.12}$$

which is represented by $F_1 = \{\mathbf{u} + \beta\ J^t\ \mathbf{h} - \mathbf{a} = \mathbf{0}\}$.

Choosing a large value for β to find critical points close to singular points of $\mathscr{V}_{\mathbb{R}}(H)$ leads to numerical instability. The solution for F_1 is refined by using homotopy techniques as discussed by Wu et al. [41]. The following step formulates this approach.

In line 9 of algorithm 8, to refine the solution u^*, we substitute $t = \frac{1}{\beta}$ in (4.2.12):

$$t \begin{pmatrix} u^1 - a^1 \\ \vdots \\ u^n - a^n \end{pmatrix} + (\text{Jacobian } H)^t \begin{pmatrix} h^1 \\ \vdots \\ h^\ell \end{pmatrix} = \begin{pmatrix} 0 \\ \vdots \\ 0 \end{pmatrix}_{n \times 1} \tag{4.2.13}$$

Then $F_2 = \{t(\mathbf{u} - \mathbf{a}) + J^t\ \mathbf{h} = \mathbf{0}\}$. We use `Bertini`, a homotopy solver, with initial point $(t_0 = \frac{1}{\beta}\ , u^*)$ to solve F_2. So the homotopy path approaches points of $\mathscr{V}_{\mathbb{R}}(H)$ when t approaches zero.

Correctness and termination See [28, 41] for the proof of termination. The use of optimization techniques to find critical points was introduced by Rouillier and his collaborators [28]. Wu et al. used the penalty function in this approach [41]. Their analysis of the convergence rate makes their method more reliable and matches the theory.

How can these real solving methods determine the hidden constraints of a differential polynomial system of equations? Also, how can they be applied to approximate symmetry? We respond to these questions in the next sections.

4.3 Implementation

4.3.1 Symbolic-Numeric completion algorithm for approximate real DPS

In this section, we sketch the outline of a symbolic-numeric completion algorithm for a system of differential polynomial equations that have approximate real coefficients. The key idea in our algorithm is based on the termination of the Cartan- Kuranishi algorithm [32]. In particular, the geometric prolongation methods of Cartan and Kuranishi are married with critical point methods in the classical (jet) geometry of differential equations in our algorithm. The numerical method helps to efficiently approximate critical points on each connected component of the involutive form of such systems.

The procedure of completing the nonlinear system depends on finding critical points on the related variety, which is explained in section §4.2.2. The CriticalPointCompletion algorithm, besides computing critical points, gives us some information in the neighborhood of singular critical points on real jet variety corresponding a given differential polynomial system of equations. This work is a sequel to [38, 41, 14] and is aimed at developing a numeric-symbolic approach to geometric prolongation of a differential system containing floating-point coefficients.

For numeric computation in our algorithm, we use the `Bertini` software, a numerical solver for polynomial systems, developed by Bates et al. [2]. It is an open-source software package that solves systems of polynomial equations numerically based on homotopy continuation. The key features of the `Bertini` that convinced us to use it in our implementation include: finding isolated solutions and positive-dimensional solutions, adaptive multi-precision that is implemented to deal with singular cases, increasing speedup of computers and capacity to interface with `Maple`.

Algorithm 9 CriticalPointCompletion

Input: System $R = \{R^1, \ldots, R^\ell\}$.
Output: InvCP which includes critical points p on the prolonged variety of the system R and $r(p)$, $k(p)$: $\pi^{r(p)} D^{k(p)} R$ is involutive at p.

1: InvCP := $\emptyset$
2: **for** $k = 0$ by 1 do
3: Prolong the system k times: $R = D^k R$.
4: Compute critical points of $D^k R$: $CP = CriticalPoints(D^k R)$
5: Compute the Jacobian $D^k R$ at point $p \in CP$, and related regularity information using local information of p determined by = RegCP
6: **if** $p \in CP$ is not regular **then** $p \in$ NonRegCP **end if**
7: Compute Tangent space $T_p(D^k R)$ for $p \in$ RegCP
8: **for** r from 0 to k **do**
9: **for** each $p \in$ RegCP use $T_p(D^k R)$ to decide involutivity of $\pi^r D^k R$
10: **if** $I_p = \{(p, r(p), k(p)) : \pi^{r(p)} D^{k(p)} R$ involutive at $p\}$ then InvCP := InvCP $\cup\, I_p$.
11: **end do**
12: **until** all $p \in$ RegCP test involutive for some projection

The CriticalPointCompletion algorithm consists of prolonging the system increasing at

each main iteration (its differential order by one) until the following criteria are satisfied. In particular, at each of these prolongations, we find the real critical points by the CritcalPoints algorithm and linearize the system about these critical points formed the tangent space $T_p(D^kR)$. The obtained linear homogeneous partial differential equation (LHPDE) for $T_p(D^kR)$ can have its projections $\pi^r D^k R$ for involutivity computed by the numerical algorithm described in §4.2.1. The Cartan-Kuranishi characters are numerically evaluated to determine if the symbol of the projected system is involutive, together with the equality of dimensions at successive prolongation-projection steps.

We remove the regularity restrictions by computing critical points based on the penalty function in step 4 of the CriticalPoints algorithm. The Cartan-Kuranishi method also makes regularity restrictions, which are a significant challenge to algorithmic computation of involutive form for nonlinear systems.

Correctness and termination. The termination of the CriticalPointCompletion algorithm is based on famous Cartan-Kuranishi prolongation theorem [6, 13]. It guarantees that a finite number of prolongations and projections yield the involutive form of a given differential system.

In our approach sketched above, certain non-regular cases are not analyzed:

(i) Regularity fails due to the generators of the system, not generating a real radical ideal. For example, $(u + v - 1)^3 = 0$, or $u^2 + (v - 1)^2 = 0$.

(ii) Regularity fails at points where the Symbol of a projected system has degenerate rank. For example, $R = u_x^2 - u = 0$ has a degenerate singular case when $\frac{\partial R}{\partial u_x} = 2u_x = 0$.

To address these difficulties:

(i) There are evolving methods in Semi-Definite Programming that can in some cases approximately compute an equivalent set of generators, generating the real radical of a real ideal. See Wang, Reid and Wolkowicz [24].

(ii) As in Reid, Wu and Golubitsky [39] we can embed a space with extra coordinates. For the example this space has coordinates $(x, u, u_x, dx, du, du_x)$ and $\frac{\partial R}{\partial u_x} z = 0$ yields the generic regular case when $\frac{\partial R}{\partial u_x} = 2u_x \neq 0$ so $z = 0$. The singular case arises when $\frac{\partial R}{\partial u_x} = 2u_x = 0$. So $u_x = 0$ and $u = 0$.

In previous work, a check was needed to determine if the CP is a regular jet point on the real variety. The rank of the Jacobian matrix is constant in some ball centered at the CP, and this rank does not vary as the independent variables are varied in a neighborhood of the CP. Now, by applying the algorithm CriticalPoints we can omit this restriction. Suppose p is a non-regular critical point when p is not a manifold point of the jet variety. Such points arise as the intersection of two smooth components. For example, $(u, v) = (0, 0)$ of $u^2 - v^2 = 0$ or $(u, v) = (\pm 1, 0)$ of $v^2 - (u^2 - 1)^3 = 0$. We address this difficulty by constructing the augmented system. See the example 4.2.2.

4.3.2 HybridInvolutiveFormLHPDE

The partition idea which we used in the CriticalPoint algorithm is efficient especially in the polynomial cases. However our experiments using this approach in CriticalPointCompletion did not show as much improvement as we expected. This lack of improvement mostly results from the need in this method to prolong the whole system at each iteration. To address the difficulty of prolonging the whole system, a natural approach is to partition the system into two disjoint subsystems: one that is purely symbolic and one which has approximate coefficients.

Then efficient symbolic differential elimination that exploits sparsity can be applied to the exact subsystem. Subsequent simplification of the approximate subsystem and its prolongations reduces its size and improves the difficulty of SVD based numerical methods when they are applied to this system.

In particular for the case of input linear homogeneous approximate DPS this leads naturally to the new algorithm described in the algorithm 10.

Algorithm 10 HybridInvolutiveFormLHPDE

Input: Linear Homogeneous differential polynomial system $R = \{R^1, \ldots, R^\ell\}$.
Output: Geometric Involutive Form for system R

1: Select the subsystem with exact coefficients from R: ExSys := ExactSystem(R)
2: The remaining equations are regarded as approximate: ApSys := $R \setminus$ ExSys
3: Find the `rif` form of ExSys wrt an orderly ranking: rExSys := `rif`(ExSys)
4: Use the Mansfield prolongation theorem (See [17]) to predict q (order of prolongation): prolongation of rExSys to order q is involutive.
5: IDExSys := `initialdata`(rExSys)
6: HFExSys := DifferentialHilbertFunction(IDExSys) (See [18])
7: Simplify the approximate system with respect to the exact system:
SimpApSys := `dsubs`(rExSys, ApSys), $\hat{q}$:= `difforder`(SimpApSys)
8: **for** k **from** 0 **do**
compute prolongations SimpApSys[k] := `dsubs`(rExSys, D^kSimpApSys)
9: **until** the joint system rExSys $\cup$ SimpApSys[k] tests projectively involutive for k prolongations (D^k) and r projections (π^r)
10: **end do**;
11: **return** rExSys,SimpApSys[k], r, k,

The full approximate (and inefficient) method prolongs the joint system rExSys $\cup$ ApSys until an involutive projected system $\pi^r D^k$(rExSys $\cup$ ApSys) is found. See Scott et al. [31] for proof of termination for that case. The main difference in the above algorithm is that the dimensions of the symbols of the prolongations of rExSys can be extracted from using the DifferentialHilbertFunction *without* prolonging the exact system. Further as each prolongation is made of the approximate system, it simplified with respect to the exact system using the `Maple` command `dsubs`. Then the algorithm produces the same dimension information as the full approximate algorithm, but exploits the availability of the RIF-form of the exact system rExSys.

Note: The assumption that part of the system is exact needs to be carefully considered with respect to the application. For example, in a circuit system a junction of two wires with currents

I_1 and I_2 naturally results in an output at a third wire $I_3 = I_1 + I_2$ which is an exact relation, and should not be viewed approximately (e.g. $I_3 = I_1 + 1.0025 * I_2$ does not make physical sense).

4.4 Example

In §4.4.1, we describe our algorithm through an example to represent how the numerical algebraic methods can be used in the prolongation and projection stages of the completion of nonlinear differential polynomial systems of equations to the involutive form. We implement our algorithms on symbolic computing language `Maple`, with homotopy numerical calculations being carried out by `Bertini` [2].

In §4.4.2 an application of our algorithm to a problem involving symmetry is given. We show that our method can be applied to the defining symmetry system containing floating-point coefficients. The successful results yield new approaches to approximate symmetry and approximate mapping.

4.4.1 Illustrative example

Example 4.4.1 *Let*

$$R = \{u_x - v = 0, \quad u_y - v = 0, \quad u^2 + v^2 - 1 = 0\} \tag{4.4.1}$$

Algorithm CriticalPointCompletion is applied to 4.4.1 and prolongs the system to order one:

$$\begin{aligned} DR = &[u_x - v = 0,\ u_y - v = 0,\ u^2 + v^2 - 1 = 0, \\ &2\,uu_x + 2\,vv_x = 0, 2\,uu_y + 2\,vv_y = 0 \quad] \end{aligned}$$

The algorithm requires critical points on the variety of this system to check the involutivity conditions. So sub-algorithm CriticalPoints is employed on DR. The CriticalPoints algorithm partitions the system DR into linear and nonlinear parts:

$$\begin{aligned} Lin\ DR &= [u_x = v, u_y = v] \\ NonLin\ DR &= [u^2 + v^2 - 1 = 0, 2\,uu_x + 2\,vv_x = 0, 2\,uu_y + 2\,vv_y = 0] \end{aligned}$$

We use `rifsimp` *of the* `DEtools` *package, to solve the linear part (Other packages such as* `RegularChains` *could be used). The output is Lin* $DR_{sol} = \{u_x = v, u_y = v\}$. *We can then simplify nonlinear part by using these solutions;*

$$NonLin\ DR = [u^2 + v^2 - 1 = 0, 2uv + 2vv_x = 0, 2uv + 2vv_y = 0]$$

For the nonlinear part with variables $\{u, v, v_x, v_y\}$ *we need to construct and solve two systems* F_1 *and* F_2. *We consider* $\beta = 1000$ *and random point*

$$a = (.358837425248938,\ .741287241561886,\ .561438802681635,\ 0.807180189405902e-1)$$

to construct system F_1 *in the CriticalPoint algorithm, Fig 4.3. The system is solved numerically by* `bertini`. *Here, we choose one solution of* F_1 *to present the process of refining.*

$$\begin{aligned} NonLin\ DR_{CP} = (&u = -0.99482675475317300000,\ v = -0.09376990666497330000, \\ &v_x = 0.95180174176519900000,\ v_y = 0.96955501366575500000) \end{aligned}$$

This solution needs to be refined as we explained in the CriticalPoint algorithm and its refinement is u^* (NonLin DR_{CP}). So, the homotopy system F_2 is constructed with start point $(t_* = 1/\beta, NonLin\ DR_{CP})$. See Fig 4.4 for the input `Bertini` *for refining u^*.* `Bertini` *returns the below values. It is the critical point on the variety for the nonlinear part of R.*

$$NonLin\ DR_{CP} = (u = -1.00000000000003,\ v = -0.137343822443538e-12,$$
$$v_x = 1.00000000000008,\ v_y = 1.00000000000007)$$

We use `maple` *to interface with* `Bertini` *to systematically complete all steps 4 to 9 of CriticalPoints algorithm.*

```
CONFIG
UseRegeneration : 1;
END;

INPUT
variable_group  u, v, vx, vy;
function    f1, f2, f3, f4;

f1 = 2000*u^3+10000*u*v^2-1999*u+4000*v^2*vx+4000*v^2*vy+.358837425248938;
f2 = 10000*u^2*v+2000*v^3-1999*v+8000*u*v*vx+4000*v*vx^2+8000*u*v*vy+4000*v*vy^2+.7412
f3 = 4000*u*v^2+4000*v^2*vx+vx+.561438802681635;
f4 = 24000*u*v^2+4000*v^2*vy+vy-0.807180189405902e-1;

END;
```

Figure 4.3: System F1

```
CONFIG
UserHomotopy: 1;
END;

INPUT
variable    u, v, vx, vy;
function    f1, f2, f3, f4;
parameter   t;
pathvariable    ps;
t = (1/1000)*ps;

f1 = 2*u^3+10*u*v^2-2*u+4*v^2*vx+4*v^2*vy+t*u+.358837425248938*t;
f2 = 10*u^2*v+2*v^3-2*v+8*u*v*vx+4*v*vx^2+8*u*v*vy+4*v*vy^2+t*v-.741287241561886*t;
f3 = 4*u*v^2+4*v^2*vx+t*vx-.561438802681635*t;
f4 = 4*u*v^2+4*v^2*vy+t*vy+0.807180189405902e-1*t;

END;
```

Figure 4.4: System F2

So the critical point on the variety R is

$$R_{CP} = Lin\ DR_{sol} \cup NonLin\ DR_{CP} = \tag{4.4.2}$$
$$(\ u = -1.00000000000003,\quad v = -0.137343822443538e-12,$$
$$u_x = -0.137343822443538e-12,\quad u_y = -0.137343822443538e-12,$$
$$v_x = 1.00000000000008,\quad v_y = 1.00000000000007) \tag{4.4.3}$$

We check that the linearized system is involutive at this point. Then $\pi^0 DR$ is the involutive form of the system at this point. The output of the CriticalPointCompletion algorithm is

$$InvCP = \{(R_{CP}, 0, 1) : \pi^0 DR\ \ involutive\ at\ \ R_{CP}\}$$

where R_{CP} is 4.4.2.

We next apply our algorithm to Example 4.2.1 with `K` = 0.

Example 4.4.2 *The Lagrange equations of motion for a unit length planar Pendulum moving under constant gravity are*

$$X_{t,t} + \Lambda X = 0,\quad Y_{t,t} + \Lambda Y + g = 0,\quad X^2 + Y^2 = 1$$

After choosing the initial time t randomly, four sequences of projection-prolongation steps were executed to find the involutive completion of the input system by running CriticalPointCompletion algorithm. The sub-algorithm CriticalPoints of the main algorithm found 8 critical points in the first iteration, and the process of checking the involutivity was executed for all of them. The results are consistent with those determined by the exact symbolic methods.

Example 4.4.3 *Real varieties admit diverse singular behavior which will be interesting to explore in future work. For example, consider the Whitney Umbrella as a constraint of a differential polynomial system. Over the real number, it is singular along the z-axis. The canopy is two dimensional and has a self-intersection along the positive z-axis, whereas the negative z-axis has dimension one. By applying the CriticalPoints algorithm after embedding the system into a higher dimension (adding slack variables), we can remove the singularity and find a point near the stick and then get a critical point on the stick by the refining step of the CriticalPoints algorithm.*

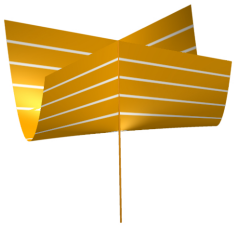


Figure 4.5: Whitney umbrella (This picture is taken from Wikipedia)

4.4.2 Application to approximate symmetry

In this section, we apply our numeric-symbolic algorithm [9] to get the involutive form of the overdetermined linearized homogeneous system for the unknown symmetries. Many symmetry properties can be determined from their involutive forms. For example, it is possible to uncover the structure of the Lie symmetry algebra and the dimension of their symmetry group, etc.

A local transformation group of a differential equation can be formed by a set of continuous point symmetries of a system R with independent variables $x = (x^1, \ldots, x^n)$ and dependent variables $u = (u^1, \ldots, u^m)$ whose associated vector field is

$$V = \sum_{i=1}^{n} \xi^i(x,u)\frac{\partial}{\partial x^i} + \sum_{j=1}^{m} \eta^j(x,u)\frac{\partial}{\partial u^j}. \tag{4.4.4}$$

The group of transformation leaves invariant the solution set of the differential equation. The linearized form of such symmetry groups in a neighborhood of their identity yields the infinitesimal Lie symmetry:

$$\begin{aligned} \tilde{x} &= x + \xi(x,u)\epsilon + O(\epsilon^2) \\ \tilde{u} &= u + \eta(x,u)\epsilon + O(\epsilon^2) \end{aligned} \tag{4.4.5}$$

and generates a Lie algebra. Determining the components $\xi^i(x,u)$ and $\eta^j(x,u)$ of (4.4.4) leads a linear homogeneous system called determining equations [1, 19]. The determining system is

derived by an explicit algorithm. Some existing computer algebra implementations are [35, 7, 8, 29].

Most of the differential equations arising from applications contain approximate coefficients, since differential systems that describe a model which are only known approximately. So applying exact symbolic methods to their related determining equations will not be reliable. Reid and his collaborators develop symbolic-numeric algorithms to address such difficulties for symmetry analysis [5, 14]. They implement the symbolic-numeric algorithm to determine the structure of the Lie algebra specified by the approximate determining system. As explained briefly in §4.2.1, their method is a combination of symbolic differential, and numeric elimination methods.

In this section, we show how CriticalPointCompletion algorithm can be applied to the approximate determining system. We try to use the numerical techniques in the prolongation (differentiation) step to get the involutive form of the determining system. The success of this idea dramatically shrinks the cost of the prolongation step.

We consider a simple example to illustrate our idea. The successful result motivates our further study of approximate symmetry.

Example 4.4.4 *Consider the symmetry defining system:*

$$\begin{aligned} R = [\eta_x + 2.1\,\xi_y = 0, \quad \gamma_x = 0 \quad \eta_y - 1.1\,\xi_x = 0, \\ \gamma_y = 0, \quad H_y = 0, \quad -2\,H\,\xi_x - \xi\,H_x = 0] \end{aligned} \tag{4.4.6}$$

The computation is performed with `Maple` *by setting* `Digits:=15`. *CriticalPoints algorithm partitions the input into two-parts, Linear and Non-Linear, to control the number of paths being followed by the numeric solver,* `Bertini`.
The output of the Non-Linear part after refining the solution which is computed by `Bertini` *is:*

$$\begin{aligned} [H = .283717587235499, \quad H_x = -.835404842698773, \\ \xi = -.225709735554639, \quad \xi_x = -.332300524553122] \end{aligned} \tag{4.4.7}$$

We consider $\beta = 1000$ *as the penalty function, and tolerance* $0.1e-5$ *for the projection stage. The dimension of the* $\pi^r D^k(R)$ *and dimension of the symbol matrix of the* $\pi^r D^k(R)$ *was computed at this critical point to check the involutity criteria.*

4.5 Conclusion

We provide a symbolic-numeric algorithm based on numerical algebraic techniques to get the involutive form of differential polynomial systems of equations, especially those containing floating-point coefficients.

Under-over determined systems of differential equations also arise in applications and have hidden constraints. They require differentiation and projection to determine their missing constraints. The difficulty of exact differentiation elimination algorithms in the case of approximate systems has motivated Reid and his collaborators to apply numerical techniques to address

this problem. In early work, Reid et al. illustrated a numeric method based on the computation of Riquier Bases and analyzed nonlinear systems of differential polynomial equations including hidden constraints [40]. Their algorithm works for a narrow class of differential systems (square systems) that are dominated by pure derivatives in one of the independent variables. Later, Reid et al. introduce geometric involutive methods where numerical homotopy continuation techniques are combined with the Cartan-Kuranishni approach [38, 41, 14]. This paper is a sequel to these works.

Adapting numerical algebraic geometry methods (witness point methods) to differential geometry has difficulties, including those difficulties from characterizing geometric prolongation. Some progress has been made in this area [39, 38, 41, 14]. No doubt, the real case is more challenging than the complex case. In the previous work, a real polynomial input system needs to satisfy certain regularity conditions [38]. In this paper, we give a real symbolic-numeric completion algorithm for real differential polynomial systems of equations containing floating-point coefficients. Our algorithm includes loosening the regularity conditions and the pure dimension for the input that was a necessary condition in previous works [11, 38].

We provide a different prolongation phase based on critical points to attempt to find at least one point on each connected component of real involutive variety of a differential system. This work is still in progress. We do not propose a general method achieving this goal, but we provide criteria that are helpful in practice. The success of this strategy in primary examples is a motivation to partition a given system into approximate and exact parts which will be developed in further work. We aim to apply reliable existing symbolic algorithms for the exact part and the CriticalPoints algorithm for the approximate part. Numerical techniques will help us to bypass the instability of exact differential elimination algorithms for approximate systems. In the future we would like to use this new method of Numerical Jet Geometry to investigate approximate symmetry and approximate mappings.

Acknowledgements

The authors would like to thank Dr. Allan Wittkopf for his helpful suggestions to work on `Bertini-Maple` interface.

Bibliography

[1] S. Anco, G. Bluman, and T. Wolf. Invertible mappings of nonlinear PDEs to linear PDEs through admitted conservation laws. Acta Applicandae Mathematicae, 101, 21–38, 2008. 35, 43, 64, 86, 94

[2] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler, Numerically Solving Polynomial Systems with the Software Package Bertini. In preparation. To be published by SIAM, 2013. 15, 16, 18, 73, 81, 84, 94

[3] G.M. Besana, S. Di Rocco, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler. Cell decomposition of almost smooth real algebraic surfaces. Num. Algorithms, DOI:10.1007/s11075-012-9646-y, published online Sept 28, 2012. 15, 73

[4] F. Boulier, D. Lazard, F. Ollivier, and M. Petitot. Representation for the radical of a finitely generated differential ideal. Proc. ISSAC 1995. ACM Press, 158–166, 1995. 13, 73

[5] J. Bonasia, F. Lemaire, G. Reid and L. Zhi. Determination of approximate symmetries of differential equations. Group Theory and Numerical Analysis, 249–266, 2005. 75, 76, 87

[6] Cartan. Systèmes différentiels, théories d'équivalence. Oeuvres complèes, Part 2, Vol. 2: Groupes finis, 1953. 14, 15, 82, 93

[7] J. Carminati and K. Vu. Symbolic computation and differential equations: Lie symmetries. Journal of Symbolic Computation, 29, 95–116, 2000. 6, 24, 46, 87

[8] A. F. Cheviakov. GeM software package for computation of symmetries and conservation laws of differential equations. Computer Physics Communications, 176(1), 48–61, 2007. 6, 8, 24, 46, 87

[9] G. Chèze and A. Galligo Four lectures on polynomial absolute factorization.In: Bronstein M. et al. (eds) Solving Polynomial Equations: Foundations, Algorithms and Computation Algorithms and Computation in Mathematics, 14, Springer, 339–392, 2005. 73

[10] V. Gerdt and Y. Blinkov. Involutive Bases of Polynomial Ideals. Mathematics and Computers in Simulation, 45, 519–541, 1998. 72

[11] J. Hauenstein. Numerically computing real points on algebraic sets. Acta Appl. Math. DOI:10.1007/s10440-012-9782-3, published online September 27, 2012. 72, 73, 78, 88, 94

[12] E. Hubert. Detecting degenerate cases in non-linear differential equations of first order Theoretical Computer Science, 187(1-2), 7–25, 1997. 73

[13] M. Kuranishi. On E. Cartans Prolongation Theorem of Exterior Differential Systems. Amer.J. of Math., 79, 1–47, 1957. 14, 15, 82

[14] I. Lisle, S.-L. Huang, and G. Reid. Structure of symmetry of pde: Exploiting partially integrated systems. Proceedings of the 2014 Symposium on Symbolic-Numeric Computation, 61–69, 2014. 23, 36, 81, 87, 88

[15] Y. Lu. Finding all real solutions of polynomial systems, Ph.D Thesis, University of Notre Dame, 2006. Results of this thesis appear in: (with D.J. Bates, A.J. Sommese, and C.W. Wampler), Finding all real points of a complex curve, Contemporary Mathematics 448, 183–205, 2007. 73

[16] E. Mansfield. Differential Gröbner Bases. Ph.D. thesis, Univ. of Sydney, 1991. 73

[17] E. MANSFIELD. A simple criterion for involutivity, J. London Math. Soc., 54, 323–345, 1996. 83

[18] Z. Mohammadi, G. Reid, and S.-L.T. Huang. Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDS to linear PDS. Technical Report arXiv:1903.03727v1 [math.AP]. ArXiV. 2019. 35, 83, 94

[19] P. Olver. Application of Lie groups to differential equations. Springer-Verlag, 2nd edition, 1993. 4, 25, 30, 44, 73, 86

[20] F. Pritchard and W. Sit. On initial value problems for ordinary DAEs. J. Symbolic Computation. 75

[21] P.J. Rabier and W.C. Rheinboldt. A geometric treatment of implicit differential-algebraic equations. Journal of Differential Equations, 109, 110–146, 1994. 75

[22] G. Reid, J. Tang, and L. Zhi. A complete Symbolic-Numeric Linear Method for Camera Pose Determination. Proc. ISSAC 2003. ACM Press. 215–223, 2003. 76

[23] G. Reid, C. Smith and J. Verschelde Geometric Completion of Differential Systems using Numeric-Symbolic Continuation. SIGSAM, Volume 36 (2),1–17, 2002. 76

[24] G. Reid, F. Wang, H. Wolkowicz and W. Wu Semidefinite Programming and facial reduction for 2 Systems of Polynomial Equations. Theoretical Computer Science, arXiv:1504.00931, 2015. 82

[25] G.J. Reid,P.Lin and A.D. Wittkopf Differential elimination-completion algorithms for DAE and PDAE. Studies in Applied Math, 106(1), 1, 45, 2001. 75

[26] G.J. Reid. Finding abstract Lie symmetry algebras of differential equations without integrating determining equations. European Journal of Applied Mathematics, 2, 319–340, 1991. 11, 17, 23, 43

[27] G.J. Reid, I.G Lisle, A. Boulton, and A.D. Wittkopf. Algorithmic determination of commutation relations for Lie symmetry algebras of PDEs. Proc. ISSAC '92, 63–68, 1992. 43, 47, 51

[28] F. Rouillier, M.-F. Roy, and M. Safey El Din. Finding at least one point in each connected component of a real algebraic set defined by a single equation. J. Complexity, 16 (4), 716-750, 2000. 78, 80

[29] T. Rocha Filho and A. Figueiredo. SADE: A Maple package for the symmetry analysis of differential equations. Computer Physics Communications, 182(2), 467–476, 2011. 6, 24, 46, 87

[30] C.J. Rust. Rankings of derivatives for elimination algorithms and formal solvability of analytic partial differential equations, Ph.D. Thesis, University of Chicago, 1998. 73

[31] R. Scott, G. Reid, W. Wu, and L. Zhi. Geometric Involutive Bases and Applications to Approximate Commutative Algebra. Approximate Commutative Algebra. Texts and Monographs in Symbolic Computation (A Series of the Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria). Springer, Vienna, 98–124, 2009. 83

[32] W. Seiler. Involution: The formal theory of differential equations and its applications in computer algebra, Algorithms and Computation in Mathematics. 24, Springer, 2010. 13, 15, 25, 28, 42, 44, 73, 75, 81

[33] A.J. Sommese and C.W. Wampler. The Numerical solution of systems of polynomials arising in engineering and science. World Scientific Press, 2005. 15, 73, 78

[34] G. Thomas. Symbolic computation of the index of quasilinear differential-algebraic equations. Proc. ISSAC, ACM Press, 196–203, 1996. 75

[35] T. Wolf. Investigating differential equations with crack, liepde, applsymm and conlaw. Handbook of Computer Algebra, Foundations, Applications, Systems, 37, 465–468, 2002. 6, 7, 8, 35, 43, 65, 87

[36] W. Wu and G. Reid. Application of Numerical Algebraic Geometry and Numerical Linear Algebra to PDE. Proc. of ISSAC06, 345–352, ACM, 2006. 15, 16, 78

[37] W. Wu and G. Reid. Symbolic-numeric Computation of Implicit Riquier Bases for PDE. Proc. of ISSAC07, 377–385, ACM, 2007. ix, 74

[38] W. Wu and G. Reid. Finding points on Real Solution Components and Applications to Differential polynomial systems. Proc. of ISSAC 2013, ACM Press, 339–346, 2013. 72, 73, 78, 81, 88, 94

[39] W. Wu, G. Reid and O. Golubitsky. Towards Geometric Completion of Differential Systems by Points. Approximate Commutative Algebra. Texts and Monographs in Symbolic Computation (A Series of the Research Institute for Symbolic Computation, Johannes Kepler University, Linz, Austria). Springer, Vienna, 79–97, 2009. 15, 16, 78, 82, 88

[40] W. Wu, G. Reid and S. Ilie, Implicit Riquier Bases for PDAE and their semi-discretizations. Journal of Symbolic Computation, 44(7), 923-941, 2009. 88

[41] W. Wu, C. Chen and G. Reid. Penalty Function Based Critical Point Approach to Compute Real Witness Solution Points of Polynomial System. Computer Algebra in Scientific Computing CASC, 377-391, 2017. 15, 18, 72, 73, 78, 79, 80, 81, 88, 94, 95

[42] A. Wittkopf. Algorithms and Implementations for Differential Elimination. Ph.D. Thesis, Simon Fraser University, 2004. 13, 76

Chapter 5

Conclusion and Future work

Finding an algorithmic method that determines whether a given differential equations system can be mapped into a target system of interest is a fundamental geometric problem that is important in applications. Such theoretically and computationally challenging mapping problems have motivated us to explore the approach we present in this thesis.

The main goal of this thesis is to develop an exact and approximate mapping algorithm that exploits symmetry information. Mappings of differential equations and differential structures have a long history in geometry and applications. Significant contributions relevant to our work are those of Riemann and Lie in the 19th-century, Cartan [7], Eisenhart [8] and more recently Olver [14] and Anderson [2]. In particular, our approach has been influenced by the work of Lyakhov-Gerdt-Michels [11] and Bluman-Kumei [4, 5].

The mapping method introduced for ODE in [11] works at the nonlinear Lie Group level, whereas we work at the linearized Lie algebra level. Lyakhov et al. [11] explicitly introduce a target with undetermined coefficients and use the full nonlinear determining equations for the mapping. In our method, we avoid using the full nonlinear determining equations for the transformations. We exploit the fact that the target appears implicitly as a subalgebra of the Lie symmetry algebra of the given differential equation. It is used in the Bluman-Kumei method [5], which depends on extracting this subalgebra by explicit non-algorithmic integration, whereas we use algorithmic differential algebra. We exploit the fact that the subalgebra corresponding to linear super-position appears as a subalgebra of the derived algebra related to the given differential equation.

In this thesis, our main goal is to develop an exact mapping algorithm. In chapter 2, we introduce our mapping algorithm, called `MapDE`, for two applications. The algorithm is given where the input and target systems are specified as Differential Polynomial System DPS, and it returns a reduced involutive form (rif-form) for the mapping equations. Meanwhile, an explicit form of the transformations is returned if the analytic integration of the related mapping equations is successful. In addition, we give an algorithm for deciding if a linear differential polynomial equation can be mapped to a linear differential polynomial equation with constant coefficients. These results are published in Mohammadi, Reid and Haung [12].

In chapter 3, we extend the `MapDE` algorithm to determine whether a nonlinear Differential Polynomial system can be invertibly mapped to a linear system and return equations for the mapping in rif-form if the map exists. This work is based on creating algorithms that exploit the results of Bluman and Kumei [5] and some aspects of [11]. One of the important achievements

of the MapDE algorithm is providing a linearization test based on the equality of the size of solution space of the given system with the size of the symmetry sub-group characterizing linear superposition which are measured by differential Hilbert Series. This crucial hypothesis appears to be missing in [5]. The results are submitted to Mathematics in Computer Science [13].

Most differential equations that arise in applications are nonlinear systems and are not usually linearizable. Consequently developing efficient tests to reject the existence of mappings based on structural and dimensional information is important. An important question is how to facilitate the search for linearizable differential systems. One approach is to find the system in an appropriate space [15]. Another approach is to embed a differential equation in a class and look for the member in the class with the largest symmetry group [5]. Anco, Bluman and Wolf [1] and Wolf [15] to facilitate the determination of linearization mappings by using multipliers for conservation laws. Besides the existing linearization tests in [1, 15, 5, 11], we implement a test based on using Hilbert Series in the MapDE algorithm. We plan to extend this test in our future work based on the change of rankings approach given in [6].

We use the Lie Algebras of Vector Fields (LAVF), Maple package in the development of our MapDE algorithm [10]. Since we work at the linearized Lie Algebra level to construct the map, LAVF commands facilitate our computation. LAVF can automatically compute defining systems and Lie algebra structure for various objects, including derived algebras, which are core operations needed by MapDE.

We heavily employ differential-elimination algorithms in our approach. Approximate examples, such as Poissons equation result in different regions of symmetry structure and dimension (See Fig. 2.5). These results have motivated us to generalize such differential-elimination algorithms to the more realistic case of differential polynomial systems with approximate real coefficients. The dependency of the MapDE on exact differential completion algorithms and failure of exact symbolic completion algorithms in the case of differential systems with floating point coefficients motivated us to work on the ideas presented in chapter 4.

In chapter 4, we give a symbolic-numeric algorithm to compute the solution (critical) points on the real variety of approximate DPS. This work is influenced by the significant results of Wu et al. [16] for algebraic equations. We extend that approach to exploiting fundamental features of the jet geometry of differential equations such as differential Hilbert functions. Our algorithm enables loosening the regularity conditions and the pure dimension assumptions required in previous works [9, 17]. We implement our algorithm in Maple with an interface to the numerical homotopy Bertini [3]. We apply our algorithm to an approximate defining system of DE and compute critical points on the involitive form of system. We plan to extend this approach to approximate symmetry and approximate mapping.

Highlights of this thesis:

(1) We extend and improve the algorithm introduced by Lyakhov et al. [11] for determining if an ODE is linearizable. Our algorithm works for ODE and PDE. For comparison, see Example 3.2.

(2) We provide an efficient test for existence of exact linearization based on differential Hilbert series.

(3) We discovered a missing crucial hypothesis in [5, Theorem 2.4.1] concerning sufficient conditions for the existence of an invertible linearization mapping.

(4) The `LAVF` package does not deal with infinite Lie pseudogroups. We make a progress on algorithms for infinite Lie pseudogroups.

(5) We apply critical points techniques introduced by Wu et al. [16] in terms of Jet Geometry.

(6) We implement our mathematical results of mapping in the computer algebra language `Maple` in the form of `MapDETools`, a sub-package of `LAVF` library. The `MapDETools` package provides tools to help the user to characterize the map if it exists. See appendix A.

We will now discuss possible future work. First, we intend to improve our main `MapDETools` package that is publicly available on GitHub at: https://github.com/GregGitHub57/MapDETools Some potential new features to our package have been discussed such as extending the existing linearization tests, and providing a potentiality to work with other differential-elimination completion (DEC) algorithms in `Maple` packages, such as `Rosenfeld Groebner` and `Thomas Decomposition`. We also plan to improve the `MapDE` algorithm for the infinite Lie pseudogroups case.

Second, many extensions of our main `MapDE` algorithm are possible. For example our algorithm is based on point symmetry. The results can be extended to generalized, contact symmetry and nonlocal symmetries. The `MapDE` works with DPS. We plan to extend `MapDE` to other systems beyond DPS. Last, we plan to extend `MapDE` to approximate systems.

Bibliography

[1] S. Anco, G.W. Bluman, and T. Wolf. Invertible Mappings of Nonlinear PDEs to Linear PDEs Through Admitted Conservation Laws. *Acta Applicandae Mathematica*, 101, 21–38, 2008. 35, 43, 64, 86, 94

[2] Anderson and Torre. New symbolic tools for differential geometry, gravitation, and field theory. Journal of Mathematical Physics, 53(1), 2012. 24, 42, 93

[3] D.J. Bates, J.D. Hauenstein, A.J. Sommese, and C.W. Wampler, Numerically Solving Polynomial Systems with the Software Package Bertini. In preparation. To be published by SIAM, 2013. 15, 16, 18, 73, 81, 84, 94

[4] Bluman and Kumei. Symmetries and Differential Equations. SpringerVerlag, 1989. 6, 93

[5] G.W. Bluman, A.F. Cheviakov, and S.C. Anco. Applications of Symmetry Methods to Partial Differential Equations, Springer, 2010. 1, 6, 8, 11, 18, 23, 24, 27, 30, 31, 32, 50, 51, 62, 63, 64, 93, 94, 95

[6] F. Boulier, F. Lemaire, and M. M. Maza. Computing differential characteristic sets by change of ordering. Journal of Symbolic Computation, 45(1), 124–149, 2010. 64, 94

[7] Cartan. Systèmes différentiels, théories d'équivalence. Oeuvres complèes, Part 2, Vol. 2: Groupes finis, 1953. 14, 15, 82, 93

[8] Eisenhart. Riemannian Geometry, Princeton University Press. 1949. 93

[9] J. Hauenstein. Numerically computing real points on algebraic sets. *Acta Appl. Math.* DOI:10.1007/s10440-012-9782-3, published online September 27, 2012. 72, 73, 78, 88, 94

[10] I.G. Lisle and S.-L.T. Huang. Algorithms calculus for Lie Determining Systems. Journal of Symbolic Computation, 482–498, 2017. 23, 24, 29, 43, 47, 51, 94

[11] D.A. Lyakhov, V.P. Gerdt, and D.L. Michels. Algorithmic Verification of Linearizability for Ordinary Differential Equations. In Proc. ISSAC 17. ACM, 285–292, 2017. 11, 23, 28, 35, 43, 48, 49, 52, 56, 57, 60, 61, 63, 64, 93, 94

[12] Z. Mohammadi, G. Reid, and S.-L.T Huang. Introduction of the MapDE algorithm for determination of mappings relating differential equations. Proceedings of International Symposium on Symbolic and Algebraic Computation (ISSAC'19), ACM, 331–338, 2019. 41, 42, 43, 49, 50, 51, 63, 93

[13] Z. Mohammadi, G. Reid, and S.-L.T. Huang. Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDS to linear PDS. Technical Report arXiv:1903.03727v1 [math.AP]. ArXiV. 2019. 35, 83, 94

[14] P. Olver. Equivalence, invariance, and symmetry. Cambridge University Press, 1995. 2, 3, 4, 9, 23, 27, 42, 48, 50, 52, 65, 93

[15] T. Wolf. Partial and Complete Linearization of PDEs Based on Conservation Laws. Differential Equations with Symbolic Computations,Trends in Mathematics, Zheng, Z. and Wang, D., 291-306, 2005. 43, 64, 94

[16] W. Wu, C. Chen and G. Reid. Penalty Function Based Critical Point Approach to Compute Real Witness Solution Points of Polynomial System *Computer Algebra in Scientific Computing CASC*, 377–391, 2017. 15, 18, 72, 73, 78, 79, 80, 81, 88, 94, 95

[17] W. Wu and G. Reid. Finding points on Real Solution Components and Applications to Differential polynomial systems. Proc. of ISSAC 2013, ACM Press, 339–346, 2013. 72, 73, 78, 81, 88, 94

Appendix A

Demo for the MapDETools Package

In this appendix we give the reader an example-based introduction to the function `MapDE` in the `MapDETools` package. The `MapDETools` Package will be a part of the `LieAlgebrasOfVectorFields` (`LAVF`) `Maple` package. `MapDETools` gives tools to help the user to determine the linearizability of nonlinear differential polynomial systems and some times construct the map if it exists. The `MapDETools` package is available at the following GitHub repository:

https://github.com/GregGitHub57/MapDETools

The `MapDETools` package is a work in progress. At the moment, some commands, e.g. `DerivedAlgebraInf(...)`, provide internal information for implementing the `MapDE` algorithm. All of them will work outside of the `MapDE` in the final version of the `MapDETools` package.

MapDETools Package

```
> with(MapDETools);
```
[*DerivedAlgebraExt*, *DerivedAlgebraInf*, *DifferentialHilbertFunction*, *MapDE*, *MyRosGrob*] **(1)**

MapDETools:-MapDE

Description

The command

```
> MapDE(Source, Target, Map, Options)
```

implements the MapDE algorithm and is designed to determine the existence of mappings of a given input Source polynomially nonlinear system of differential equations to a Target differential system (or class), via an invertible mapping on the space of independent and dependent variables of the differential system.

Please see the paper ISSAC 2019 for the case of a specific source and specific target, and also for mappings from a variable coefficient source PDE to a constant coefficient PDE, *"Introduction of the MapDE Algorithm for Determination of Mappings Relating Differential Equations",* Proceedings of the 2019 on International Symposium on Symbolic and Algebraic Computation, Pages 331-338, ACM, 2019.

The implementation we give here is for the case where the source is a DPS and the target is a linear differential system with theory and examples given in our paper "*Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE*" , Z. Mohammadi, G. Reid, and S. -L.T. Huang. 2019. Technical Report arXiv:1903.03727v1 [math.AP]. ArXiV. http://arxiv.org/abs/1903.03727v1.

```
> MapDE(Source, ToLinearDE, Map, Options)
```

Parameters:

Source: is a list **[DE::list, vars::list, infinitesimals::list]** where
DE: Is a list of polynomially nonlinear differential equations and inequations.
vars: [[x], [u]] where [x] is a list of independent variable names and [u] is a list of dependent variable names.
infinitesimals: [[xi], [eta]] is a list of the names of the infinitesimals corresponding to independent and dependent variables respectively.

ToLinearDE: For the application in this worksheet (mapping nonlinear to linear) the entry for Target is: **ToLinearDE**

Map: [[], [[psi],[phi]]] A list of inputs related to the mapping functions.
[]: Empty in the examples below (under development). The user can enter a system of differential equations, in psi, phi as extra constraints for the mapping problem.

[psi]: is a list of the names for the components of the mappings variables corresponding to the independent variables. Thus the number of components of [psi] is equal to the number of independent variables of the Source.
[phi]: is a list of the names for the components of the mappings variables corresponding to the depenthat variables. Thus the number of components of [phi] is equal to the number of dependent variables of the Source

`Options:` A list of options controlling outputs and other aspects of MapDE,
outputvars = [[indep], [deps]], where
[indep]: is a list of the names of the output target independent variables
[deps]: is a list of the names of the output target dependent variables

OutPutDetails: which user can print more output from MapDE

infolevel[MapDETools] := integer Gives useful details about MapDE, during its application. Where integer {0, 1, 2, 3, 4, 5, 6}

Example 1: Running Example (with details)

$$u\,u_{x,x,x}+3\,u_x\,u_{x,x}-\frac{3\left(u\,u_{x,x}+u_x^2+1\right)^2}{u\,u_x+x}-\frac{8\,x\left(u\,u_x+x\right)^4\left(u^2+x^2+1\right)}{u^2+x^2}=0$$

The first example is the DE above used for the running Example in the paper "*Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE*".

The running example begins with Example 1 and Eqn (15) given by DE[1] below:

```
> infolevel[MapDETools] := 0;
```

$$infolevel_{MapDETools} := 0 \qquad \textbf{(2.1)}$$

```
> DE[1] := [u(x)*(diff(diff(diff(u(x), x), x), x))+3*(diff(u(x),
    x))*(diff(diff(u(x), x), x))-3*((diff(u(x), x))^2+u(x)
    *(diff(diff(u(x), x), x))+1)^2/(u(x)*(diff(u(x), x))+x)-8*x
    *(u(x)*(diff(u(x), x))+x)^4*(1+u(x)^2+x^2)/(u(x)^2+x
    ^2) = 0]:
```

```
> PDEtools:-ToJet(DE[1], [u](x));
```

$$\left[u\,u_{x,x,x}+3\,u_x\,u_{x,x}-\frac{3\left(u\,u_{x,x}+u_x^2+1\right)^2}{u\,u_x+x}-\frac{8\,x\left(u\,u_x+x\right)^4\left(u^2+x^2+1\right)}{u^2+x^2}=0\right] \qquad \textbf{(2.2)}$$

Let's try Maple's dsolve:

```
> dsolve(DE[1]);
```

No solutions returned.

What does MapDE function enable us to do with this?

Lets apply MapDE. Here [[x], [u]] inputs indep (x) and dep var (u) names; [[xi], [eta]] inputs the

symmetry infinitesimal names; [[psi], [phi]] the input names for the components of the Mapping; [[xh], [uh]] the target indep and dep variable names.

```
> MDE[1] := MapDE([DE[1], [[x], [u]], [[xi], [eta]]], ToLinearDE, [[ ],
      [[psi], [phi]]], [OutputDetails, outputvars = [[xh], [uh]]]) :

MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Single ODE so apply the Lyakhov-Gerdt-Michels ODE Test
for Existence of Linearization
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim = 3
ranking = [rho[1] xi eta] RifStrategy = ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = 0.78e-1
MapDE: After rif ncases = 3
MapDE: Number consistent cases satisfying minimum dim condns 1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= s^2+s+1 = s^2+s+1 =HF(P) for consistent case j =
1
MapDE: System linearizable, now attempting to integrate
mapping eqns
```

```
> indices(MDE[1]);
```

$$[IDDetSys], [PsiPhiSys], [DerAlgID], [DerAlgSys], [RifMapEqs], [IsDATrivial], [MapEqs], [TargetLinearDE], [RifMapPivs], [ElimRhoTarget], [IsLinearizable], [rDetSys], [IDFullElimSys], [RifMapEqsSysComplete], [TranRho], [MapTransSYS], [StructRels], [TimeInfo] \tag{2.3}$$

```
> MDE[1][RifMapPivs];
```

$$\left[\frac{\partial}{\partial u}\,\psi(x,u) \neq 0, \left(\frac{\partial}{\partial x}\,\phi(x,u)\right) u - \left(\frac{\partial}{\partial u}\,\phi(x,u)\right) x \neq 0\right] \tag{2.4}$$

```
> MDE[1][PsiPhiSys];
```

$$\left[\frac{\partial^2}{\partial x^2}\,\phi(x,u) = -\frac{1}{u^3}\left(\left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right) u\,x^2 - 2\left(\frac{\partial^2}{\partial u \partial x}\,\phi(x,u)\right) u^2 x - \left(\frac{\partial}{\partial u}\,\phi(x,u)\right) u^2 - \left(\frac{\partial}{\partial u}\,\phi(x,u)\right) x^2\right), \frac{\partial}{\partial x}\,\psi(x,u) = \frac{\left(\frac{\partial}{\partial u}\,\psi(x,u)\right) x}{u}\right] \tag{2.5}$$

```
> MDE[1][MapTransSYS];
```

$$[xh = u^2 + x^2, uh = -x] \tag{2.6}$$

```
> MDE[1][TargetLinearDE];
```

$$\left[\frac{d^3}{dxh^3}\, uh(xh) = -\frac{uh(xh)\,(xh+1)}{xh}\right] \tag{2.7}$$

>

Note that above MapDE has obtained the target DE which is Eq (38) in our paper.

```
> MDE[1][IsLinearizable];
```

$$true \tag{2.8}$$

```
> MDE[1][TimeInfo];
```

$$[2.391,\ 2.469,\ 3.016,\ true] \tag{2.9}$$

TimeInfo reports the CPU time for the existence of the linearization of the LGM test, HilbertFunction test and the total time for the existence and construction steps of MapDE algorithm, respectively. Also, this list contains the result of the linearizability test.

Detailed Output:

This is the rif-form of the determining system S for symmetries given in

```
> PDETools:-ToJet(MDE[1][rDetSys], [xi, eta](x, u));
```

$$\left[\eta_{u,u,u} = \frac{1}{u^3\left(u^2+x^2\right)}\left(8\,\eta_x u^8 x - 8\,\eta_u u^7 x^2 + 8\,\eta_x u^6 x^3 - 8\,\eta_u u^5 x^4 - 16\,\eta\, u^8\right.\right. \tag{2.10}$$

$$-24\,\eta\, u^6 x^2 - 8\,\eta\, u^4 x^4 + 8\,\eta_x u^6 x - 8\,\eta_u u^5 x^2 - 16\,\eta\, u^6 - 8\,\eta\, u^4 x^2 + 3\,\eta_u u^3$$

$$\left.+3\,\eta_u x^2 u - 3\,\eta\, u^2 - 3\,\eta\, x^2\right),\ \eta_{x,x}$$

$$= \frac{\eta_{u,u}\, u^2 x^4 + 2\,x\,\eta_x u^4 + \eta_u u^3 x^2 + \eta_u u\, x^4 - 2\,\eta\, u^4 + \eta\, u^2 x^2 - \eta\, x^4}{u^4 x^2},\ \eta_{x,u}$$

$$\left.= \frac{x^2\,\eta_{u,u}\, u^2 + \eta_u u^3 - x\,u^2\,\eta_x + \eta_u x^2 u + \eta\, u^2 - \eta\, x^2}{u^3 x},\ \xi = -\frac{\eta\, u}{x}\right]$$

>

This is the ID for the Det Sys, ID(S), given in (17). Note it also includes the dim(S)=3, DiffDim(R) = differential dimension = 0, HF(R) = $s^2 + s + 1$:

```
> MDE[1][IDDetSys];
```

$$\left[\left[\eta(x_0, u_0) = \_C1,\ D_1(\eta)(x_0, u_0) = \_C2,\ D_2(\eta)(x_0, u_0) = \_C3,\ D_{2,2}(\eta)(x_0, u_0) = \_C4\right],\right. \tag{2.11}$$

$$\left.[\,],\ 3,\ 0,\ s^2 + s + 1\right]$$

The structure relations above evaluated at a regular point (x,u) = (1,1) give (18) in our paper:

```
> MDE[1][StructRels];
```

$$\begin{bmatrix} 1..4\ x\ 1..4\ x\ 1..4\ Array \\ Data\ Type:\ anything \\ Storage:\ sparse \\ Order:\ Fortran\_order \end{bmatrix} \tag{2.12}$$

```
> LieAlgebrasOfVectorFields:-DisplayStructure((2.12), 'Y', format
      = "commutatorList");
```

$$\left[[Y_1, Y_2] = -\frac{u\,Y_1}{x} - \frac{(u^2+x^2)\,Y_2}{u\,x^2}, [Y_1, Y_3] = Y_1 - \frac{(u^2+x^2)\,Y_3}{u\,x^2}, [Y_1, Y_4] = -\frac{(u^2+x^2)\,Y_4}{u\,x^2}, [Y_2, Y_3] = Y_2 + \frac{u\,Y_3}{x}, [Y_2, Y_4] = \frac{u\,Y_4}{x}, [Y_3, Y_4] = -Y_4 \right] \tag{2.13}$$

```
> subs([x = 1, u = 1], (2.13));
```

$$[[Y_1, Y_2] = -Y_1 - 2\,Y_2, [Y_1, Y_3] = Y_1 - 2\,Y_3, [Y_1, Y_4] = -2\,Y_4, [Y_2, Y_3] = Y_2 + Y_3, [Y_2, Y_4] = Y_4, [Y_3, Y_4] = -Y_4] \tag{2.14}$$

A basis for Derived Algebra generated by the commutators of elements of the Lie Algebra can be easily derived by Gauss Elimination of the coefficient matrix for the commutators from the commutators above, yielding the basis given in Eq (20) of our paper, then shown to be abelian giving (21) of our paper, and by Algorithm 1, that the ODE is linearizable. In fact we do not follow this in MapDE, instead getting the determing equations for the derived algebra, by the approach of Lisle and Huang resulting in the determining equations for the Derived Algebra given Eq (25) of our paper:

```
>
> PDEtools:-ToJet(MDE[1][DerAlgSys], [xi, eta](x, u));
```

$$\left[\eta_{u,u,u} = \frac{-8\,\eta\,u^8 - 8\,\eta\,u^6x^2 - 8\,\eta\,u^6 + 3\,\eta_u u^3 + 3\,\eta_u x^2 u - 3\,\eta\,u^2 - 3\,\eta\,x^2}{u^5 + u^3x^2}, \eta_x = -\frac{-\eta_u x^2 u - \eta\,u^2 - \eta\,x^2}{x\,u^2}, \xi = -\frac{\eta\,u}{x} \right] \tag{2.15}$$

```
>
```

The derived algebra (DA) initial data ID is:

```
> MDE[1][DerAlgID];
```

$$[[\eta(x_0, u_0) = \_C1, D_2(\eta)(x_0, u_0) = \_C2, D_{2,2}(\eta)(x_0, u_0) = \_C3], [\,]] \tag{2.16}$$

```
>
```

So the DA is 3-dim, and MapDE shows shown to be abelian by using LAVF comands (internally IsAbelian(DA)).

Note in the output above is calculated ID(P) and its HilbertFunction HF(P).

In particular the key computation of the algorithm, checking that HF(R)= s^2+s+1 = HF(P) so the system is linearizable.

The Target Linear DE is:

```
> MDE[1][TargetLinearDE];
```

$$\left[\frac{d^3}{dxh^3}\, uh(xh) = -\frac{uh(xh)\ (xh+1)}{xh}\right] \tag{2.17}$$

```
> MDE[1][MapTransSYS];
```

$$[xh = u^2 + x^2,\ uh = -x] \tag{2.18}$$

```
> PDEtools:-ToJet(MDE[1][MapEqs], [xi, eta, rho[1], psi, phi](x, u));
```

$$\left[\eta_{u,\,u,\,u} = \frac{-8\,\eta\,u^8 - 8\,\eta\,u^6 x^2 - 8\,\eta\,u^6 + 3\,\eta_u u^3 + 3\,\eta_u x^2 u - 3\,\eta\,u^2 - 3\,\eta\,x^2}{u^3\,(u^2+x^2)},\ \eta_x\right. \tag{2.19}$$

$$= \frac{\eta_u x^2 u + \eta\,u^2 + \eta\,x^2}{x\,u^2},\ \xi = -\frac{\eta\,u}{x},\ 0 = \eta\,\psi_u + \xi\,\psi_x,\ \rho_1(x,u) = \eta\,\phi_u + \xi\,\phi_x,\ -\phi_u\,\psi_x$$

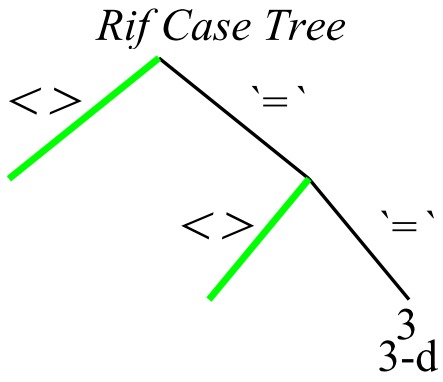
$$\left. + \phi_x\,\psi_u \neq 0,\ \left(\frac{\partial}{\partial x}\,\rho_1(x,u)\right)\psi_u - \left(\frac{\partial}{\partial u}\,\rho_1(x,u)\right)\psi_x = 0\right]$$

```
>
```

Note that rho[1](x,u) above is $\hat{\eta}$ as a function of (x,u).

```
> eval(MDE[1][RifMapEqsSysComplete]) :
>
>
> DEtools:-caseplot(eval(MDE[1][RifMapEqsSysComplete]), [xi, eta,
    rho[1]]);
```

Case 3: 3-d



The caseplot above shows the case analysis by rifsimp. The green branches are where the system was either inconsistent or the dimension fell below the minimum for linearization. The black branch corresponds to the the linearizable case.

Below are the inequations for invertibility of the map:

```
> eval(MDE[1][RifMapEqsSysComplete][3][Pivots]);
```

$$\left[\frac{\partial}{\partial u}\,\psi(x,u)\neq 0,\ \left(\frac{\partial}{\partial x}\,\phi(x,u)\right)u-\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)x\neq 0\right] \tag{2.20}$$

Checking the result using dchange

You can check with dchange, that this nice linear DE, MDE[1][*TargetLinearDE*] is (invertibly) mapped to the complicated nonlinear DE by the transformation:

```
> MDE[1][MapTransSYS];
```

$$[xh=u^2+x^2,\ uh=-x] \tag{2.1.1}$$

```
> MDE[1][TargetLinearDE];
```

$$\left[\frac{d^3}{dxh^3}\,uh(xh)=-\frac{uh(xh)\,(xh+1)}{xh}\right] \tag{2.1.2}$$

```
> simplify(PDEtools:-dchange( {xh = x^2 + u(x)^2, uh(xh) =-x},
    (2.1.2),  [x, u(x)]));
```

$$\left[\frac{1}{8\left(u(x)\left(\frac{d}{dx}\,u(x)\right)+x\right)^5}\left(u(x)\left(u(x)\left(\frac{d}{dx}\,u(x)\right)+x\right)\left(\frac{d^3}{dx^3}\,u(x)\right)\right.\right. \tag{2.1.3}$$

$$-3\left(\frac{d^2}{dx^2}\,u(x)\right)^2u(x)^2+\left(-3\,u(x)\left(\frac{d}{dx}\,u(x)\right)^2+3\left(\frac{d}{dx}\,u(x)\right)x\right.$$

$$\left.\left.-6\,u(x)\right)\left(\frac{d^2}{dx^2}\,u(x)\right)-3\left(\left(\frac{d}{dx}\,u(x)\right)^2+1\right)^2\right)=\frac{x\left(1+u(x)^2+x^2\right)}{u(x)^2+x^2}\right]$$

```
> rDE[1] := DEtools:- rifsimp(DE[1]) :
> PDEtools:-dsubs(rDE[1][Solved], (2.1.3));
```

$$\left[\frac{x\left(1+u(x)^2+x^2\right)}{u(x)^2+x^2}=\frac{x\left(1+u(x)^2+x^2\right)}{u(x)^2+x^2}\right] \tag{2.1.4}$$

So we have independently checked the result of MapDE.

Example 2: $diff\left(u(x)^2,x,x,x\right)+u(x)^2=0$

The second example, `diff(u(x)^2, x$d) + u(x)^2 = 0`, used as a member of the test set of ODE of order $3 \le d \le 15$, in our paper "*Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE*". It originates from Example 5, Eqn(41). This set was considered by Lyakhov, Gerdt and Michels [ISSAC 17. ACM, 285–292, 2017].

Now apply MapDE where d = 3 where [[x], [u]] inputs indep (x) and dep var (u) names and[[xh], [uh]] are the target indep and dep variable names.

```
>
> infolevel[MapDETools] := -1;
```

$$infolevel_{MapDETools} := -1 \tag{3.1}$$

```
> DE[2] := [diff(u(x)^2, x, x, x) + u(x)^2 = 0];
```

$$DE_2 := \left[6\left(\frac{d}{dx}u(x)\right)\left(\frac{d^2}{dx^2}u(x)\right) + 2\,u(x)\left(\frac{d^3}{dx^3}u(x)\right) + u(x)^2 = 0\right] \tag{3.2}$$

```
> MDE[2] := MapDE([DE[2], [[x], [u]], [[xi], [eta]]], ToLinearDE, [[],
      [[psi], [phi]]], [OutputDetails, outputvars = [[xh], [uh]]]) :

> indices(MDE[2]);
```

$$[IDDetSys], [PsiPhiSys], [DerAlgID], [DerAlgSys], [RifMapEqs], [IsDATrivial], [MapEqs], [TargetLinearDE], [RifMapPivs], [IsLinearizable], [rDetSys], [ElimRhoTarget], [IDFullElimSys], [RifMapEqsSysComplete], [TranRho], [MapTransSYS], [StructRels], [TimeInfo] \tag{3.3}$$

```
> MDE[2][IsLinearizable];
```

$$true \tag{3.4}$$

```
> MDE[2][MapTransSYS];
```

$$[xh = x, uh = u^2] \tag{3.5}$$

```
> MDE[2][TargetLinearDE];
```

$$\left[\frac{d^3}{dxh^3}uh(xh) = -uh(xh)\right] \tag{3.6}$$

```
> MDE[2][PsiPhiSys];
```

$$\left[\frac{\partial^2}{\partial u^2}\phi(x,u) = \frac{\frac{\partial}{\partial u}\phi(x,u)}{u}, \frac{\partial}{\partial u}\psi(x,u) = 0\right] \tag{3.7}$$

```
> MDE[2][RifMapPivs];
```

$$\left[\frac{\partial}{\partial u}\phi(x,u) \neq 0, \frac{\partial}{\partial x}\psi(x,u) \neq 0\right] \tag{3.8}$$

Example 3: Test Class of linearizable ODE of order d:

$diff(u^2(x), x\$d) + u^2(x) = 0,\ d = 3..15$

```
>
>
> infolevel[MapDETools] := -3;
```

$$infolevel_{MapDETools} := -3 \tag{4.1}$$

Now apply MapDE to the test set ,

`diff(u(x)^2, x$d) + u(x)^2 = 0 for order 3 ≤ d ≤ 15`, considered by Lyakhov, Gerdt and Michels [ISSAC 17. ACM, 285–292, 2017] and present the existance and constructed CPU times. See our paper "*Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE*" for more details.

```
> Nstar := 3 :
  Nfin := 15 :
  ODEtime := [ ]:
  LinTest := [ ]:
  for n from Nstar to Nfin do
       ODE := [diff(u(x)^2, x$n) + u(x)^2 = 0];
       st := time( );
     mode[n] := MapDETools:-MapDE([ODE, [[x], [u]], [[xi], [eta]]], ToLinearDE, [[ ],
       [[psi], [phi]]], [outputvars = [[xh], [uh]]]);
     LinTest := [op(LinTest), mode[n][TimeInfo]] :
        ODEtime := [op(ODEtime), time( ) - st];
  end do:

  Test := evalb(LinTest = [seq(true, j = 1 .. (Nfin - Nstar + 1))]) :
```

ODEtime represents the existence and construction time for MapDE for d=3..15.

```
> ODEtime;
```

$$[0.610, 0.390, 0.672, 0.813, 1.140, 1.438, 2.047, 3.265, 3.547, 4.938, 6.718, 9.110, 12.625] \quad (4.2)$$

LinTest is the lists of CPU time for Linearizability existence of LGM test, HilbertFunction test and the time for the existence and construction steps of MapDE algorithm with the result of the test for each order. More precise, timings from t = 0 (start of MapDE) to the time to LGMLin linearization Existence confirmation ,time to Hilbert Existence confirmation, time from t = 0 to existence and construction of the linearizing transformations, respectively.

```
> LinTest;
```

$$[[0.390, 0.453, 0.578, true], [0.250, 0.297, 0.390, true], [0.500, 0.563, 0.672, true], [0.610, 0.688, 0.813, true], [0.890, 0.968, 1.140, true], [1.219, 1.282, 1.438, true], [1.719, 1.812, 2.047, true], [2.609, 2.750, 3.265, true], [3.063, 3.188, 3.547, true], [4.375, 4.516, 4.938, true], [6.015, 6.156, 6.718, true], [8.125, 8.313, 9.110, true], [11.312, 11.547, 12.625, true]] \quad (4.3)$$

So, the existence time for each test will be:

```
> LGMtime := [seq(LinTest[i][1], i = 1 ..13)];
```

$$LGMtime := [0.390, 0.250, 0.500, 0.610, 0.890, 1.219, 1.719, 2.609, 3.063, 4.375, 6.015, 8.125, 11.312] \quad (4.4)$$

```
> HilbertFunctiontime := [seq(LinTest[i][2], i = 1 ..13)];
```

$$HilbertFunctiontime := [0.453, 0.297, 0.563, 0.688, 0.968, 1.282, 1.812, 2.750, 3.188, \quad (4.5)$$

4.516, 6.156, 8.313, 11.547]

>

Also, the exsitence and construction time in MapDE algorithm for d=3..15 will be:

```
> MapDEExistenceConsTime := [seq(LinTest[i][3], i=1..13)];
```

$$MapDEExistenceConsTime := [0.578, 0.390, 0.672, 0.813, 1.140, 1.438, 2.047, 3.265, 3.547, 4.938, 6.718, 9.110, 12.625] \quad (4.6)$$

>

We note that the times differ by roughly a factor of 2 from those in row 4 of Table 1 in our first submission. These are probably due to code changes since submission, and we investigated them further. These times are still far less than the times given in LGM (out of memory for d $\geq$ 10).

Example 4: Nonlinear System $[v_t = v^3 u_x, u_t = u^5 v_x]$

Lets try another example which Maple's pdsolve yielded no result after about 1000 secs and 700 MB of RAM. We stopped the computation.

```
> infolevel[MapDETools] := 0;
```

$$infolevel_{MapDETools} := 0 \quad (5.1)$$

```
> DE[4] := [diff(v(x, t), t) = v(x, t)^3*(diff(u(x, t), x)),
      diff(u(x, t), t) = u(x, t)^5*(diff(v(x, t), x))];
```

$$DE_4 := \left[\frac{\partial}{\partial t} v(x,t) = v(x,t)^3 \left(\frac{\partial}{\partial x} u(x,t)\right), \frac{\partial}{\partial t} u(x,t) = u(x,t)^5 \left(\frac{\partial}{\partial x} v(x,t)\right)\right] \quad (5.2)$$

>

The command below yields no result after 1000 seconds.

```
> pdsolve(DE[4]);
```

```
> MDE[4] := MapDE([DE[4], [[x, t], [u, v]], [[xi, tau], [eta, beta]]],
      ToLinearDE, [[ ], [[psi, Upsilon], [varphi, phi]]],
      [OutputDetails, outputvars = [[xh, th], [uh, vh]]]) :
MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim =
infinity ranking = [rho[1] rho[2] xi tau eta beta] RifStrategy
= ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = 7.141
MapDE: After rif ncases = 46
MapDE: Number consistent cases satisfying minimum dim condns 2
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 2/(1-s) = 2/(1-s) =HF(P) for consistent case j =
1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 2/(1-s) = 2/(1-s) =HF(P) for consistent case j =
2
```

```
MapDE: System linearizable, now attempting to integrate
mapping eqns
>
>
> MDE[4][IsLinearizable];
```

$$true \tag{5.3}$$

```
> MDE[4][MapTransSYS];
```

$$[xh = u,\ th = v,\ uh = x,\ vh = t] \tag{5.4}$$

```
> MDE[4][TargetLinearDE];
```

$$\left[\frac{\partial}{\partial xh}\ uh(xh, th) = \frac{\frac{\partial}{\partial th}\ vh(xh, th)}{xh^5},\ \frac{\partial}{\partial xh}\ vh(xh, th) = \left(\frac{\partial}{\partial th}\ uh(xh, th)\right) th^3\right] \tag{5.5}$$

```
> MDE[4][PsiPhiSys];
```

$$\left[\frac{\partial^2}{\partial x^2}\ \phi(x, t, u, v) = 0,\ \frac{\partial^2}{\partial x^2}\ \varphi(x, t, u, v) = 0,\ \frac{\partial^2}{\partial t \partial x}\ \varphi(x, t, u, v) = 0,\ \frac{\partial^2}{\partial t^2}\ \varphi(x, t, u, v) = 0,\ \frac{\partial}{\partial x}\ \Upsilon(x, t, u, v) = 0,\ \frac{\partial}{\partial x}\ \psi(x, t, u, v) = 0,\ \frac{\partial}{\partial t}\ \Upsilon(x, t, u, v) = 0,\ \frac{\partial}{\partial t}\ \phi(x, t, u, v) = 0,\ \frac{\partial}{\partial t}\ \psi(x, t, u, v) = 0\right] \tag{5.6}$$

```
> MDE[4][RifMapPivs];
```

$$\left[\left(\frac{\partial}{\partial u}\ \psi(x, t, u, v)\right)\left(\frac{\partial}{\partial v}\ \Upsilon(x, t, u, v)\right) - \left(\frac{\partial}{\partial v}\ \psi(x, t, u, v)\right)\left(\frac{\partial}{\partial u}\ \Upsilon(x, t, u, v)\right) \neq 0,\ \frac{\partial}{\partial t}\ \varphi(x, t, u, v) \neq 0,\ \frac{\partial}{\partial x}\ \phi(x, t, u, v) \neq 0\right] \tag{5.7}$$

```
>
>
```

Example 5: Burger's equation $\left[u_{x,x} = -u\,u_x + u_t\right]$

Consider Burger's equation appeared in Example 6. Early tests in MapDE shows that it is not linearizable by point transformation.

```
> DE[5] := [diff(u(x, t), x, x) = diff(u(x, t), t) - u(x, t)·diff(u(x, t),
    x)];
```

$$DE_5 := \left[\frac{\partial^2}{\partial x^2}\ u(x, t) = \frac{\partial}{\partial t}\ u(x, t) - u(x, t)\left(\frac{\partial}{\partial x}\ u(x, t)\right)\right] \tag{6.1}$$

```
> PDEtools:-ToJet(DE[5], [u](x, t));
```

$$\left[u_{x,x} = -u\,u_x + u_t\right] \tag{6.2}$$

```
> MDE[5] := MapDE([DE[5], [[x, t], [u]], [[xi, tau], [eta]]],
```

```
      ToLinearDE, [[ ], [[psi, Upsilon], [phi]]], [OutputDetails,
     outputvars = [[x, t], [u]]]);
MapDE: Input R not linearizable since dim(L}= 5  <  infinity
= dim(R))
```

$$MDE_5 := \text{table}\left(\left[IsLinearizable = false,\ DimIDR = \left[\left[\infty,\ 1,\ \frac{1}{1-s} + \frac{s}{1-s}\right]\right],\ [[\ ],\ [u(x_0, t) = \_F1(t),\ D_1(u)(x_0, t) = \_F2(t)\ ]]\right],\ DimIDS = [[5,\ 0,\ 3+2s]],\ [[\eta(x_0, t_0, u_0) = \_C1,\ D_2(\eta)(x_0, t_0, u_0) = \_C2,\ D_3(\eta)(x_0, t_0, u_0) = \_C3,\ \tau(x_0, t_0, u_0) = \_C4,\ \xi(x_0, t_0, u_0) = \_C5],\ [\ ]]]\right) \quad (6.3)$$

```
> MDE[5][IsLinearizable];
```

$$false \quad (6.4)$$

```
>
```

Example 6: Burger's System $v_x = u,\ v_t = u_x + \frac{u^2}{2}$

Rewriting Burger's equation in conserved form, see Example 6 and Equ (43) in "*Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE*", and run MapDE in the new system.

```
> infolevel[MapDETools] := 0;
```

$$infolevel_{MapDETools} := 0 \quad (7.1)$$

```
> DE[6] := [diff(v(x, t), x) = u(x, t),  diff(v(x, t), t) = diff(u(x,
     t), x)  +  u(x, t)^2/2];
```

The expected computation time for this example is about 50 sec.

$$DE_6 := \left[\frac{\partial}{\partial x} v(x,t) = u(x,t),\ \frac{\partial}{\partial t} v(x,t) = \frac{\partial}{\partial x} u(x,t) + \frac{u(x,t)^2}{2}\right] \quad (7.2)$$

```
> MDE[6] := MapDE([DE[6], [[x, t], [u, v]], [[xi, tau], [eta, beta]]],
     ToLinearDE, [[ ], [[psi, Upsilon], [varphi, phi]]],
    [outputvars = [[xh, th], [uh, vh]]]) :
MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim =
infinity ranking = [rho[1] rho[2] xi tau eta beta] RifStrategy
= ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = 31.766
MapDE: After rif ncases = 43
MapDE: Number consistent cases satisfying minimum dim condns 2
```

```
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 2/(1-s) = 2/(1-s) =HF(P) for consistent case j =
1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 2/(1-s) = 2/(1-s) =HF(P) for consistent case j =
2
MapDE: System linearizable, now attempting to integrate
mapping eqns
```

```
> 
```

```
> indices(MDE[6]);
```

$$[PsiPhiSys], [MapTransSYS], [IsLinearizable], [TargetLinearDE], [RifMapPivs] \tag{7.3}$$

```
> MDE[6][IsLinearizable];
```

$$true \tag{7.4}$$

```
> MDE[6][MapTransSYS];
```

$$\left[xh = x,\ th = t,\ uh = u\, e^{\frac{v}{2}},\ vh = e^{\frac{v}{2}}\right] \tag{7.5}$$

```
> MDE[6][TargetLinearDE];
```

$$\left[\frac{\partial}{\partial xh}\ uh(xh, th) = \frac{vh(xh, th)}{2},\ \frac{\partial}{\partial xh}\ vh(xh, th) = 2\ \frac{\partial}{\partial th}\ uh(xh, th)\right] \tag{7.6}$$

```
> MDE[6][PsiPhiSys];
```

$$\left[\frac{\partial^2}{\partial u^2}\ \phi(x, t, u, v) = 0,\ \frac{\partial^2}{\partial u^2}\ \varphi(x, t, u, v) = 0,\ \frac{\partial^2}{\partial u \partial v}\ \phi(x, t, u, v) = \frac{\frac{\partial}{\partial u}\ \phi(x, t, u, v)}{2},\ \frac{\partial^2}{\partial u \partial v}\right. \tag{7.7}$$

$$\varphi(x, t, u, v) = \frac{\frac{\partial}{\partial u}\ \varphi(x, t, u, v)}{2},\ \frac{\partial^2}{\partial v^2}\ \phi(x, t, u, v) = \frac{\frac{\partial}{\partial v}\ \phi(x, t, u, v)}{2},\ \frac{\partial^2}{\partial v^2}\ \varphi(x, t, u, v)$$

$$= \frac{\frac{\partial}{\partial v}\ \varphi(x, t, u, v)}{2},\ \frac{\partial}{\partial u}\ \Upsilon(x, t, u, v) = 0,\ \frac{\partial}{\partial u}\ \psi(x, t, u, v) = 0,\ \frac{\partial}{\partial v}\ \Upsilon(x, t, u, v) = 0,$$

$$\left.\frac{\partial}{\partial v}\ \psi(x, t, u, v) = 0\right]$$

```
> MDE[6][RifMapPivs];
```

$$\left[\left(\frac{\partial}{\partial x}\ \psi(x, t, u, v)\right)\left(\frac{\partial}{\partial t}\ \Upsilon(x, t, u, v)\right) - \left(\frac{\partial}{\partial t}\ \psi(x, t, u, v)\right)\left(\frac{\partial}{\partial x}\ \Upsilon(x, t, u, v)\right) \neq 0,\right. \tag{7.8}$$

$$\left(\frac{\partial}{\partial u}\ \varphi(x, t, u, v)\right)\left(\frac{\partial}{\partial v}\ \phi(x, t, u, v)\right) - \left(\frac{\partial}{\partial v}\ \varphi(x, t, u, v)\right)\left(\frac{\partial}{\partial u}\ \phi(x, t, u, v)\right) \neq 0,$$

$$\left.\frac{\partial}{\partial x}\ \psi(x, t, u, v) \neq 0\right]$$

Example 7: KP Equation $u_{x,t} + \frac{3\,u_x^2}{2} + \frac{3\,u\,u_{x,x}}{2} + \frac{u_{x,x,x,x}}{4} + \frac{3\,u_{y,y}}{4} = 0$

KP Equation $u_{x,t} + \frac{3\,u_x^2}{2} + \frac{3\,u\,u_{x,x}}{2} + \frac{u_{x,x,x,x}}{4} + \frac{3\,u_{y,y}}{4} = 0$ correspond to equation (45) and

Example7 in "*Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE*".

```
> DE[7] := [diff(diff(u(x, y, t), t) + 3/2·u(x, y, t)·diff(u(x, y, t),
      x) + 1/4·diff(u(x, y, t), x, x, x), x) + 3/4·diff(u(x, y, t), y, y)
      = 0];
```

$$DE_7 := \left[\frac{\partial^2}{\partial t \partial x} u(x,y,t) + \frac{3\left(\frac{\partial}{\partial x} u(x,y,t)\right)^2}{2} + \frac{3\,u(x,y,t)\left(\frac{\partial^2}{\partial x^2} u(x,y,t)\right)}{2} + \frac{\frac{\partial^4}{\partial x^4} u(x,y,t)}{4} + \frac{3\frac{\partial^2}{\partial y^2} u(x,y,t)}{4} = 0\right] \quad (8.1)$$

```
> MDE[7] := MapDE([DE[7], [[x, y, t], [u]], [[xi, eta, tau], [beta]]],
      ToLinearDE, [[], [[psi, Upsilon, varphi], [phi]]],
     [outputvars = [[xh, yh, th], [uh]]]);
MapDE: Input R not linearizable since DiffDim(L}= 1  <  2  =
DiffDim(R))
```

$$MDE_7 := \text{table}([IsLinearizable = false]) \quad (8.2)$$

Example 8: Liouville's equation $u_{x,x} + u_{y,y} = e^u$

>

Consider Liouville's equation $u_{x,x} + u_{y,y} = e^u$ which we rewrite as a differential polynomial system (DPS) by using Maple's function dpolyform. That yields $v = e^u$ and the Liouville equation in the form $v_{x,x} = -v_{y,y} + \frac{v_x^2}{v} + \frac{v_y^2}{v} + v^2$.

```
> DE[8] := [∂^2/∂x^2 u(x, y) + ∂^2/∂y^2 u(x, y) = exp(u(x, y))];
```

$$DE_8 := \left[\frac{\partial^2}{\partial x^2} u(x,y) + \frac{\partial^2}{\partial y^2} u(x,y) = e^{u(x,y)}\right] \quad (9.1)$$

```
> MDE[8] := MapDE([DE[8], [[x, y], [u]], [[xi, eta], [beta]]],
      ToLinearDE, [[], [[psi, Upsilon], [phi]]], [outputvars = [[xh,
      yh], [uh]]]);
Error, (in DEtools/Rif/setup) Unable to handle functions of
constants and/or dependent variables such as {exp(u(x,y))}
```

```
> PDETools:- dpolyform(DE[8]);
```

$$\left[\frac{\partial^2}{\partial x^2} u(x,y) + \frac{\partial^2}{\partial y^2} u(x,y) - \_F1(x,y) = 0,\ -\_F1(x,y)\left(\frac{\partial}{\partial x} u(x,y)\right) + \frac{\partial}{\partial x}\_F1(x,y) = 0,\ -\_F1(x,y)\left(\frac{\partial}{\partial y} u(x,y)\right) + \frac{\partial}{\partial y}\_F1(x,y) = 0\right], [\_F1(x,y) \neq 0], [\_F1(x,y) = e^{u(x,y)}] \quad (9.2)$$

```
> PDEtools:-dsubs([_F1(x, y) = v(x, y)], [∂²/∂x² u(x, y) + ∂²/∂y² u(x, y)
      − _F1(x, y) = 0, -_F1(x, y) (∂/∂x u(x, y)) + ∂/∂x _F1(x, y) = 0, -_F1(x,
      y) (∂/∂y u(x, y)) + ∂/∂y _F1(x, y) = 0]);
```

$$\left[\frac{\partial^2}{\partial x^2}u(x,y)+\frac{\partial^2}{\partial y^2}u(x,y)-v(x,y)=0,\ -v(x,y)\left(\frac{\partial}{\partial x}u(x,y)\right)+\frac{\partial}{\partial x}v(x,y)=0,\ -v(x,y)\left(\frac{\partial}{\partial y}u(x,y)\right)+\frac{\partial}{\partial y}v(x,y)=0\right] \tag{9.3}$$

```
> DEtools[rifsimp](PDEtools:-dsubs([_F1(x, y) = v(x, y)], [∂²/∂x² u(x,
      y) + ∂²/∂y² u(x, y) − _F1(x, y) = 0, -_F1(x, y) (∂/∂x u(x, y)) + ∂/∂x
      _F1(x, y) = 0, -_F1(x, y) (∂/∂y u(x, y)) + ∂/∂y _F1(x, y) = 0]), [[u],
      [_F1]]);
```

$$\text{table}\left(\left[Pivots=[v(x,y)\neq 0],\ Case=\left[\left[v(x,y)\neq 0,\ \frac{\partial}{\partial x}u(x,y)\right]\right],\ Solved=\left[\frac{\partial}{\partial x}u(x,y)=\frac{\frac{\partial}{\partial x}v(x,y)}{v(x,y)},\ \frac{\partial}{\partial y}u(x,y)=\frac{\frac{\partial}{\partial y}v(x,y)}{v(x,y)},\ \frac{\partial^2}{\partial x^2}v(x,y)=-\frac{-v(x,y)^3+\left(\frac{\partial^2}{\partial y^2}v(x,y)\right)v(x,y)-\left(\frac{\partial}{\partial x}v(x,y)\right)^2-\left(\frac{\partial}{\partial y}v(x,y)\right)^2}{v(x,y)}\right]\right]\right) \tag{9.4}$$

```
> DE[8] := [∂²/∂x² v(x, y) = -∂²/∂y² v(x, y) + (∂/∂x v(x, y))²/v(x, y) + (∂/∂y v(x, y))²/v(x, y)
      + v(x, y)²];
```

$$DE_8 := \left[\frac{\partial^2}{\partial x^2}v(x,y)=-\frac{\partial^2}{\partial y^2}v(x,y)+\frac{\left(\frac{\partial}{\partial x}v(x,y)\right)^2}{v(x,y)}+\frac{\left(\frac{\partial}{\partial y}v(x,y)\right)^2}{v(x,y)}+v(x,y)^2\right] \tag{9.5}$$

```
> MDE[8] := MapDE([DE[8], [[x, y], [v]], [[xi, eta], [beta]]],
      ToLinearDE, [[ ], [[psi, Upsilon], [phi]]], [outputvars = [[xh,
      yh], [vh]], OutputDetails]) :
MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
```

```
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim =
infinity ranking = [rho[1] xi eta beta] RifStrategy =
ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = 2.188
MapDE: After rif ncases = 12
MapDE: Number consistent cases satisfying minimum dim condns 0
MapDE: System is not linearizable
```

```
> indices(MDE[8]);
```

[*rDetSys*], [*RifDE*], [*DerAlgSys*], [*MapEqs*], [*IsLinearizable*], [*RifMapEqsSysComplete*], [*RifMapEqs*] **(9.6)**

```
> eval(MDE[8][RifDE]);
```

$$\text{table}\left(\left[4=\left[\frac{\partial}{\partial t}\,v(x,t)=v(x,t)^3\left(\frac{\partial}{\partial x}\,u(x,t)\right),\ \frac{\partial}{\partial t}\,u(x,t)=u(x,t)^5\left(\frac{\partial}{\partial x}\,v(x,t)\right)\right],5\right. \tag{9.7}$$

$$=\left[\frac{\partial^2}{\partial x^2}\,u(x,t)=\frac{\partial}{\partial t}\,u(x,t)-u(x,t)\left(\frac{\partial}{\partial x}\,u(x,t)\right)\right],6=\left[\frac{\partial}{\partial x}\,v(x,t)=u(x,t),\ \frac{\partial}{\partial t}\right.$$

$$v(x,t)=\frac{\partial}{\partial x}\,u(x,t)+\frac{u(x,t)^2}{2}\Bigg],7=\Bigg[\frac{\partial^2}{\partial t\partial x}\,u(x,y,t)+\frac{3\left(\frac{\partial}{\partial x}\,u(x,y,t)\right)^2}{2}$$

$$+\frac{3\,u(x,y,t)\left(\frac{\partial^2}{\partial x^2}\,u(x,y,t)\right)}{2}+\frac{\frac{\partial^4}{\partial x^4}\,u(x,y,t)}{4}+\frac{3\,\frac{\partial^2}{\partial y^2}\,u(x,y,t)}{4}=0\Bigg],8$$

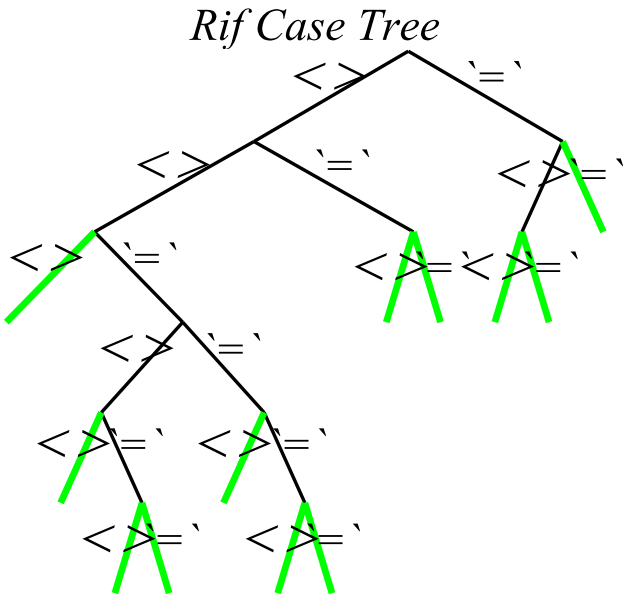
$$\left.=\left[\frac{\partial^2}{\partial x^2}\,v(x,y)=-\frac{\partial^2}{\partial y^2}\,v(x,y)+\frac{\left(\frac{\partial}{\partial x}\,v(x,y)\right)^2}{v(x,y)}+\frac{\left(\frac{\partial}{\partial y}\,v(x,y)\right)^2}{v(x,y)}+v(x,y)^2\right]\right]\Bigg)$$

```
>
> MDE[8][RifMapEqs];
```

$$[\,] \tag{9.8}$$

The system is not linearizable. The Rif Case Tree is given below:

```
> eval(MDE[8][RifMapEqsSysComplete]) :
> DEtools:-caseplot(eval(MDE[8][RifMapEqsSysComplete]), [xi, eta,
    beta, rho[1]]);
```




```
> MDE[8][IsLinearizable];
```

$$false \tag{9.9}$$

```
> MDE[8][DerAlgSys];
```

$$\left[\beta_{x,x} = -\beta_{y,y}, \xi_{y,y} = \frac{\beta_x}{2\nu}, \xi_x = -\frac{\beta}{2\nu}, \eta_x = -\xi_y, \eta_y = -\frac{\beta}{2\nu}, \xi_v = 0, \eta_v = 0, \beta_v = \frac{\beta}{\nu}\right] \tag{9.10}$$

Example 9: $\left[u_x^5 + u_t u_{x,x} = 0\right]$

```
> DE[9] := [diff(u(x, t), t)·diff(u(x, t), x, x) + (diff(u(x, t), x))^5
    = 0];
```

$$DE_9 := \left[\left(\frac{\partial}{\partial t} u(x,t)\right)\left(\frac{\partial^2}{\partial x^2} u(x,t)\right) + \left(\frac{\partial}{\partial x} u(x,t)\right)^5 = 0\right] \tag{10.1}$$

```
> PDEtools:-ToJet(DE[9], u(x, t));
```

$$\left[u_x^5 + u_t u_{x,x} = 0\right] \tag{10.2}$$

```
> MDE[9] := MapDE([DE[9], [[x, t], [u]], [[xi, beta], [eta]]],
     ToLinearDE, [[ ], [[psi, Upsilon], [phi]]], [outputvars = [[xh,
    th], [uh]]]);
MapDE: Input R not linearizable since dim(L}= 5  <  infinity
= dim(R))
```

$$MDE_9 := \text{table}([IsLinearizable = false]) \tag{10.3}$$

Example 10: Nonlinear Diffusion Equation $\left[u_t = \frac{u_{x,x}}{u_x^2}\right]$

```
> DE[10] := [diff(u(x, t), t) = diff(u(x, t), x, x)/(diff(u(x, t), x))^2];
```

$$DE_{10} := \left[\frac{\partial}{\partial t}\,u(x,t) = \frac{\frac{\partial^2}{\partial x^2}\,u(x,t)}{\left(\frac{\partial}{\partial x}\,u(x,t)\right)^2}\right] \tag{11.1}$$

```
>
> infolevel[MapDETools] := 0;
```

$$infolevel_{MapDETools} := 0 \tag{11.2}$$

```
> MDE[10] := MapDE([DE[10], [[x, t], [u]], [[xi, tau], [eta]]],
      ToLinearDE, [[ ], [[psi, Upsilon], [phi]]], [outputvars = [[xh,
     th], [uh]]]);
MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim =
infinity ranking = [rho[1] xi tau eta] RifStrategy =
ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = 1.984
MapDE: After rif ncases = 13
MapDE: Number consistent cases satisfying minimum dim condns 3
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1/(1-s)+s/(1-s) = 1/(1-s)+s/(1-s) =HF(P) for
consistent case j = 1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1/(1-s)+s/(1-s) = 1/(1-s)+s/(1-s) =HF(P) for
consistent case j = 2
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1/(1-s)+s/(1-s) = 1/(1-s)+s/(1-s) =HF(P) for
consistent case j = 3
MapDE: System linearizable, now attempting to integrate
mapping eqns
```

$$MDE_{10} := \text{table}\left(\left[PsiPhiSys = \left[\frac{\partial^2}{\partial x^2}\,\phi(x,t,u) = 0, \frac{\partial}{\partial x}\,\Upsilon(x,t,u) = 0, \frac{\partial}{\partial x}\,\psi(x,t,u) = 0\right],\right.\right. \tag{11.3}$$

$$MapTransSYS = [xh = t, th = u, uh = x], IsLinearizable = true, TargetLinearDE = \left[\frac{\partial^2}{\partial th^2}\right.$$

$$\left.\left.uh(xh,th) = \frac{\partial}{\partial xh}\,uh(xh,th)\right], RifMapPivs = \left[\frac{\partial}{\partial x}\,\phi(x,t,u) \neq 0, \left(\frac{\partial}{\partial t}\,\psi(x,t,\right.\right.$$

$$u)\Big)\left(\frac{\partial}{\partial u}\,\Upsilon(x,t,u)\right)-\left(\frac{\partial}{\partial u}\,\psi(x,t,u)\right)\left(\frac{\partial}{\partial t}\,\Upsilon(x,t,u)\right)\neq 0,\ \frac{\partial}{\partial t}\,\psi(x,t,u)\neq 0,$$
$$\frac{\partial}{\partial u}\,\psi(x,t,u)\neq 0\Big]\Big]\Big)$$

From the above output we see that the input DE is linearized to the heat equation.

>

>

Example 11:

>

```
> DE[11] := [diff(diff(u(x, t), x), x) = (1/4)*(u(x, t)^4*x^4 + x
    ^7 + u(x, t)^4*t^2 + t^2*x^3 + 4*u(x, t)^3*(diff(u(x, t), t))
    -12*u(x, t)^2*(diff(u(x, t), x))^2-6*x)/u(x, t)^3];
```

$$DE_{11} := \left[\frac{\partial^2}{\partial x^2}\,u(x,t)=\frac{1}{4\,u(x,t)^3}\left(u(x,t)^4x^4+x^7+u(x,t)^4t^2+t^2x^3+4\,u(x,t)^3\left(\frac{\partial}{\partial t}\,u(x,t)\right)-12\,u(x,t)^2\left(\frac{\partial}{\partial x}\,u(x,t)\right)^2-6x\right)\right] \tag{12.1}$$

```
> PDEtools:-ToJet(DE[11], u(x, t));
```

$$\left[u_{x,x}=\frac{u^4x^4+x^7+t^2u^4+t^2x^3+4u^3u_t-12u^2u_x^2-6x}{4u^3}\right] \tag{12.2}$$

```
> MDE[11] := MapDE([DE[11], [[x, t], [u]], [[xi, tau], [eta]]],
     ToLinearDE, [[ ], [[psi, Upsilon], [phi]]], [outputvars = [[xh,
    th], [uh]]]);


MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim =
infinity ranking = [rho[1] xi tau eta] RifStrategy =
ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = .328
MapDE: After rif ncases = 4
MapDE: Number consistent cases satisfying minimum dim condns 1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1/(1-s)+s/(1-s) = 1/(1-s)+s/(1-s) =HF(P) for
consistent case j = 1
MapDE: System linearizable, now attempting to integrate
mapping eqns
```

$$MDE_{11} := \text{table}\left(\left[PsiPhiSys=\left[\frac{\partial^2}{\partial u^2}\,\phi(x,t,u)=\frac{3\left(\frac{\partial}{\partial u}\,\phi(x,t,u)\right)}{u},\ \frac{\partial}{\partial u}\,\Upsilon(x,t,u)=0,\right.\right.\right. \tag{12.3}$$

$$\frac{\partial}{\partial u}\ \psi(x, t, u) = 0\Big], \mathit{MapTransSYS} = [xh = x, th = t, uh = u^4], \mathit{IsLinearizable} = \mathit{true},$$

$$\mathit{TargetLinearDE} = \left[\frac{\partial^2}{\partial xh^2}\ uh(xh, th) = \frac{\partial}{\partial th}\ uh(xh, th) + \left(xh^4 + th^2\right) uh(xh, th)\right],$$

$$\mathit{RifMapPivs} = \left[\frac{\partial}{\partial u}\ \phi(x, t, u) \neq 0, \left(\frac{\partial}{\partial x}\ \Upsilon(x, t, u)\right)\left(\frac{\partial}{\partial t}\ \psi(x, t, u)\right) - \left(\frac{\partial}{\partial t}\ \Upsilon(x, t, u)\right)\left(\frac{\partial}{\partial x}\ \psi(x, t, u)\right) \neq 0\right]\Big]\Big)$$

Example 12: $v_t = u_x, u_t = \frac{v_x}{u^2}$

> Linearization for Hodograph system $v_t = u_x, u_t = \frac{v_x}{u^2}$

```
> DE[12] := [diff(v(x, t), t) = diff(u(x, t), x),  diff(u(x, t), t)
      = diff(v(x, t), x)/u(x, t)^2 ];
```

>

$$DE_{12} := \left[\frac{\partial}{\partial t}\ v(x, t) = \frac{\partial}{\partial x}\ u(x, t), \frac{\partial}{\partial t}\ u(x, t) = \frac{\frac{\partial}{\partial x}\ v(x, t)}{u(x, t)^2}\right] \quad \textbf{(13.1)}$$

This example took about 170 sec on our laptops (I7 with 8GB RAM).

```
> t0 := time(); MDE[12] := MapDE([DE[12], [[x, t], [u, v]], [[xi, tau],
      [eta, beta]]], ToLinearDE, [[], [[psi, Upsilon], [varphi,
      phi]]], [outputvars = [[xh, th], [uh, vh]]]); t1 := time() - t0;
                                t0 := 209.031
MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim =
infinity ranking = [rho[1] rho[2] xi tau eta beta] RifStrategy
= ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = 100.938
MapDE: After rif ncases = 145
MapDE: Number consistent cases satisfying minimum dim condns 4
MapDE: R is linearizable by DifferentialHilbertFunction Test
```

```
since HF(R)= 2/(1-s) = 2/(1-s) =HF(P) for consistent case j =
1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 2/(1-s) = 2/(1-s) =HF(P) for consistent case j =
2
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 2/(1-s) = 2/(1-s) =HF(P) for consistent case j =
3
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 2/(1-s) = 2/(1-s) =HF(P) for consistent case j =
4
MapDE: System linearizable, now attempting to integrate
mapping eqns
```

$$MDE_{12} := \text{table}\left(\left[PsiPhiSys = \left[\frac{\partial^2}{\partial x^2}\,\phi(x,t,u,v) = 0, \frac{\partial^2}{\partial x^2}\,\varphi(x,t,u,v) = 0, \frac{\partial^2}{\partial t \partial x}\,\varphi(x,t,u,v) = 0, \frac{\partial^2}{\partial t^2}\,\varphi(x,t,u,v) = 0, \frac{\partial}{\partial x}\,\Upsilon(x,t,u,v) = 0, \frac{\partial}{\partial x}\,\psi(x,t,u,v) = 0, \frac{\partial}{\partial t}\,\Upsilon(x,t,u,v) = 0, \frac{\partial}{\partial t}\,\phi(x,t,u,v) = 0, \frac{\partial}{\partial t}\,\psi(x,t,u,v) = 0\right], MapTransSYS = [xh = u, th = v, uh = x, vh = t], IsLinearizable = true, TargetLinearDE = \left[\frac{\partial}{\partial xh}\,uh(xh,th) = \left(\frac{\partial}{\partial th}\,vh(xh,th)\right) xh^2, \frac{\partial}{\partial xh}\,vh(xh,th) = \frac{\partial}{\partial th}\,uh(xh,th)\right], RifMapPivs = \left[\left(\frac{\partial}{\partial u}\,\psi(x,t,u,v)\right)\left(\frac{\partial}{\partial v}\,\Upsilon(x,t,u,v)\right) - \left(\frac{\partial}{\partial v}\,\psi(x,t,u,v)\right)\left(\frac{\partial}{\partial u}\,\Upsilon(x,t,u,v)\right) \neq 0, \frac{\partial}{\partial t}\,\varphi(x,t,u,v) \neq 0, \frac{\partial}{\partial x}\,\phi(x,t,u,v) \neq 0, \frac{\partial}{\partial u}\,\psi(x,t,u,v) \neq 0\right]\right]\right)$$

$$t1 := 199.094 \qquad \textbf{(13.2)}$$

```
>
```

Example 13: A linearizable DE where MapDE can't find the linearizing transformation

```
>
> DE[13] := [(((-27*x^2*u(x)^4+3*u(x)^2)*(diff(u(x), x))-9*u(x)
    ^2*x^2+1)*(diff(diff(diff(u(x), x), x), x))+(81*x^2*u(x)
    ^4-9*u(x)^2)*(diff(diff(u(x), x), x))^2+((162*u(x)^3*x^2
    -36*u(x))*(diff(u(x), x))^2-54*x*u(x)*(3*u(x)^3+x)
    *(diff(u(x), x))-54*x*u(x)^2)*(diff(diff(u(x), x), x))+6
    +90*u(x)^2*(diff(u(x), x))^5+(270*u(x)^2*x^2-6)
    *(diff(u(x), x))^4+(-324*u(x)^3*x-18*x^2)*(diff(u(x), x))
    ^3+(54*u(x)^4-108*x*u(x))*(diff(u(x), x))^2+36
    *(diff(u(x), x))*u(x)^2)/(1+3*(diff(u(x), x))*u(x)^2)^5+(x
    ^3+u(x))*(x+u(x)^3+1)/(x+u(x)^3) = 0];
```

$$DE_{13} := \left[\frac{1}{\left(1+3\left(\frac{d}{dx}\,u(x)\right)u(x)^2\right)^5}\left(\left(-27x^2u(x)^4+3u(x)^2\right)\left(\frac{d}{dx}\,u(x)\right)\right.\right. \qquad \textbf{(14.1)}$$

$$-9\,u(x)^2x^2+1\Big)\left(\frac{d^3}{dx^3}\,u(x)\right)+\left(81\,x^2u(x)^4-9\,u(x)^2\right)\left(\frac{d^2}{dx^2}\,u(x)\right)^2$$
$$+\left(\left(162\,u(x)^3x^2-36\,u(x)\right)\left(\frac{d}{dx}\,u(x)\right)^2-54\,x\,u(x)\left(3\,u(x)^3+x\right)\left(\frac{d}{dx}\,u(x)\right)\right.$$
$$\left.-54\,x\,u(x)^2\right)\left(\frac{d^2}{dx^2}\,u(x)\right)+6+90\,u(x)^2\left(\frac{d}{dx}\,u(x)\right)^5+\left(270\,u(x)^2x^2\right.$$
$$\left.-6\right)\left(\frac{d}{dx}\,u(x)\right)^4+\left(-324\,u(x)^3x-18\,x^2\right)\left(\frac{d}{dx}\,u(x)\right)^3+\left(54\,u(x)^4\right.$$
$$\left.-108\,x\,u(x)\right)\left(\frac{d}{dx}\,u(x)\right)^2+36\left(\frac{d}{dx}\,u(x)\right)u(x)^2\Big)$$
$$+\frac{\left(x^3+u(x)\right)\left(x+u(x)^3+1\right)}{x+u(x)^3}=0\Bigg]$$

```
> map(PDEtools:-ToJet, DE[13], u(x));
```

$$\left[\frac{1}{\left(3\,u_x u^2+1\right)^5}\left(\left(\left(-27\,u^4x^2+3\,u^2\right)u_x-9\,u^2x^2+1\right)u_{x,x,x}+\left(81\,u^4x^2-9\,u^2\right)u_{x,x}^2\right.\qquad(14.2)$$
$$+\left(\left(162\,u^3x^2-36\,u\right)u_x^2-54\,x\,u\left(3\,u^3+x\right)u_x-54\,x\,u^2\right)u_{x,x}+6+90\,u^2u_x^5$$
$$+\left(270\,u^2x^2-6\right)u_x^4+\left(-324\,u^3x-18\,x^2\right)u_x^3+\left(54\,u^4-108\,u\,x\right)u_x^2+36\,u_xu^2\Big)$$
$$\left.+\frac{\left(x^3+u\right)\left(u^3+x+1\right)}{u^3+x}=0\right]$$

```
> MDE[13] := MapDE([DE[13], [ [x], [u]], [[xi], [eta]]], ToLinearDE,
      [[ ], [[psi], [phi]]], [outputvars = [[xh], [uh]]]) :
MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Single ODE so apply the Lyakhov-Gerdt-Michels ODE Test
for Existence of Linearization
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim = 3
ranking = [rho[1] xi eta] RifStrategy = ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = .188
MapDE: After rif ncases = 3
MapDE: Number consistent cases satisfying minimum dim condns 1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= s^2+s+1 = s^2+s+1 =HF(P) for consistent case j =
1
MapDE: System linearizable, now attempting to integrate
mapping eqns
MapDE: pdsolve could not find phi
```

```
> indices(MDE[13]);
```

$$[PsiPhiSys], [IsLinearizable], [RifMapPivs], [RifMapEqs] \tag{14.3}$$

```
> MDE[13][IsLinearizable];
```

$$true \tag{14.4}$$

```
>
```

We emphasize that MapDE is not guaranteed to determine the mapping explicitly as the above example shows. MapDE does determine if a system is or is not linearizable, as well as giving a reduced involutive system for the map. The system above is linearizable by passing the HF linearization existence test, but the heuristic integration step, using pdsolve to attempt to solve the PsiPhi map sub-system (the sub-system with highest derivatives in mapping variables phi and psi) together with its inequations to find an explicit form of the mapping was not successful. See step 13 in MapDE algorithm in our paper "*Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDE to linear PDE*" for more information.

```
> eval(MDE[13][RifMapEqs][Solved]);
```

$$\left[\frac{\partial^3}{\partial u^3}\,\eta(x,u) = \frac{1}{9\,u^7x^2+9\,x^3u^4-u^5-x\,u^2}\left(-243\,\eta(x,u)\,u^{13}x^2-243\,\eta(x,u)\,u^{10}x^3 \right. \right. \tag{14.5}$$

$$-243\,u^{10}\eta(x,u)\,x^2+27\,\eta(x,u)\,u^{11}+27\,\eta(x,u)\,u^8x+27\,\eta(x,u)\,u^8+72\left(\frac{\partial}{\partial u}\right.$$

$$\eta(x,u)\Big)u^5x^2-72\,\eta(x,u)\,u^4x^2+72\left(\frac{\partial}{\partial u}\,\eta(x,u)\right)u^2x^3-72\,\eta(x,u)\,u\,x^3$$

$$-6\left(\frac{\partial^2}{\partial u^2}\,\eta(x,u)\right)u^4+10\left(\frac{\partial}{\partial u}\,\eta(x,u)\right)u^3-6\left(\frac{\partial^2}{\partial u^2}\,\eta(x,u)\right)u\,x+10\left(\frac{\partial}{\partial u}\right.$$

$$\left.\eta(x,u)\Big)x\right), \frac{\partial}{\partial x}\,\eta(x,u)$$

$$= \frac{-54\,\eta(x,u)\,u^4x+9\left(\frac{\partial}{\partial u}\,\eta(x,u)\right)u^2x^2+18\,\eta(x,u)\,u\,x^2-\frac{\partial}{\partial u}\,\eta(x,u)}{27\,u^4x^2-3\,u^2}, \rho_1(x,u)$$

$$= -3\left(\frac{\partial}{\partial x}\,\phi(x,u)\right)\eta(x,u)\,u^2+\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)\eta(x,u), \xi(x,u) = -3\,\eta(x,u)\,u^2, \frac{\partial^2}{\partial x^2}$$

$$\phi(x,u) = \frac{1}{-81\,u^6x^2+9\,u^4}\left(-162\left(\frac{\partial}{\partial x}\,\phi(x,u)\right)u^6x-54\left(\frac{\partial^2}{\partial u\partial x}\,\phi(x,u)\right)u^4x^2\right.$$

$$+54\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)u^4x+9\left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right)u^2x^2-18\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)u\,x^2$$

$$+6\left(\frac{\partial^2}{\partial u \partial x}\,\phi(x,u)\right)u^2+6\left(\frac{\partial}{\partial x}\,\phi(x,u)\right)u-\frac{\partial^2}{\partial u^2}\,\phi(x,u)\Big),\ \frac{\partial}{\partial x}\,\psi(x,u)$$
$$=\frac{\frac{\partial}{\partial u}\,\psi(x,u)}{3\,u^2}\Bigg]$$

```
> MDE[13][PsiPhiSys];
```

$$\left[\frac{\partial^2}{\partial x^2}\,\phi(x,u)=\frac{1}{-81\,u^6x^2+9\,u^4}\left(-162\left(\frac{\partial}{\partial x}\,\phi(x,u)\right)u^6x-54\left(\frac{\partial^2}{\partial u \partial x}\,\phi(x,u)\right)u^4x^2\right.\right. \tag{14.6}$$
$$+54\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)u^4x+9\left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right)u^2x^2-18\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)ux^2$$
$$+6\left(\frac{\partial^2}{\partial u \partial x}\,\phi(x,u)\right)u^2+6\left(\frac{\partial}{\partial x}\,\phi(x,u)\right)u-\frac{\partial^2}{\partial u^2}\,\phi(x,u)\Big),\ \frac{\partial}{\partial x}\,\psi(x,u)$$
$$\left.\left.=\frac{\frac{\partial}{\partial u}\,\psi(x,u)}{3\,u^2}\right]\right]$$

```
> MDE[13][RifMapPivs];
```

$$\left[\frac{\partial}{\partial u}\,\psi(x,u)\neq 0,\ 3\left(\frac{\partial}{\partial x}\,\phi(x,u)\right)u^2-\frac{\partial}{\partial u}\,\phi(x,u)\neq 0\right] \tag{14.7}$$

(14.8)

Example 14: First order ODE

A single first order ODE always has an infinite Lie symmetry group. Indeed the superposition group that we target is a 1 dimensional subgroup of this infinite dimensional group. Further all first order ODE are linearizable by invertible mappings of the type we consider. However finding the linearization is equivalent to explicitly solving the ODE, a well-known problem, for which there is no general algorithm.

```
>
> DE[14] := [diff(u(x), x) = u(x)^2 + u(x)^4];
```

$$DE_{14}:=\left[\frac{\mathrm{d}}{\mathrm{d}x}\,u(x)=u(x)^2+u(x)^4\right] \tag{15.1}$$

```
> MDE[14] := MapDE([DE[14], [[x], [u]], [[xi], [eta]]], ToLinearDE,
      [[ ], [[psi], [phi]]], [outputvars = [[xh], [uh]]]) :
MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim = 1
ranking = [rho[1] xi eta] RifStrategy = ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
```

```
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = 2.047
MapDE: After rif ncases = 6
MapDE: Number consistent cases satisfying minimum dim condns 4
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1 = 1 =HF(P) for consistent case j = 1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1 = 1 =HF(P) for consistent case j = 2
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1 = 1 =HF(P) for consistent case j = 4
MapDE: System linearizable, now attempting to integrate
mapping eqns
```

```
> indices(MDE[14]);
```

$$[PsiPhiSys], [MapTransSYS], [IsLinearizable], [TargetLinearDE], [RifMapPivs] \quad (15.2)$$

```
> MDE[14][TargetLinearDE];
```

$$\left[\frac{\mathrm{d}}{\mathrm{d}xh}\, uh(xh) = \frac{\_C1\, uh(xh)}{xh^2\,(xh^2+1)}\right] \quad (15.3)$$

```
> MDE[14][MapTransSYS];
```

$$\left[xh = u,\ uh = \frac{\_C2\, \mathrm{e}^{-\_C1\, x}}{\_C1}\right] \quad (15.4)$$

```
>
```

Example 15: First order ODE where MapDE can't find the linearizing transformation

See the details in Examples 13 and 14.

```
> DE[15] := [diff(u(x), x) = x^3 + u(x)^2 + u(x)^4];
```

$$DE_{15} := \left[\frac{\mathrm{d}}{\mathrm{d}x}\, u(x) = x^3 + u(x)^2 + u(x)^4\right] \quad (16.1)$$

```
> MDE[15] := MapDE([DE[15], [[x], [u]], [[xi], [eta]]], ToLinearDE,
      [[ ], [[psi], [phi]]], [OutputDetails, outputvars = [[xh],
      [uh]]]) :
MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim = 1
ranking = [rho[1] xi eta] RifStrategy = ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = 4.219
MapDE: After rif ncases = 6
MapDE: Number consistent cases satisfying minimum dim condns 4
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1 = 1 =HF(P) for consistent case j = 1
MapDE: R is linearizable by DifferentialHilbertFunction Test
```

```
since HF(R)= 1 = 1 =HF(P) for consistent case j = 2
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= 1 = 1 =HF(P) for consistent case j = 4
MapDE: System linearizable, now attempting to integrate
mapping eqns
MapDE: pdsolve could not find phi
```

```
> indices(MDE[15]);
```

$$[PsiPhiSys], [IsLinearizable], [RifMapPivs], [RifMapEqs] \tag{16.2}$$

```
> MDE[15][IsLinearizable];
```

$$true \tag{16.3}$$

```
> MDE[15][PsiPhiSys];
```

$$\left[\frac{\partial^3}{\partial u^2 \partial x}\,\phi(x,u) = -\frac{1}{\frac{\partial}{\partial u}\,\phi(x,u)}\left(-\left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right)^2 u^4 + \left(\frac{\partial}{\partial u}\,\phi(x,u)\right)\left(\frac{\partial^3}{\partial u^3}\,\phi(x,u)\right)u^4 - \left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right)^2 x^3 + 4\left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right)\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)u^3 + \left(\frac{\partial}{\partial u}\,\phi(x,u)\right)\left(\frac{\partial^3}{\partial u^3}\,\phi(x,u)\right)x^3 - \left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right)^2 u^2 + 12\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)^2 u^2 + \left(\frac{\partial}{\partial u}\,\phi(x,u)\right)\left(\frac{\partial^3}{\partial u^3}\,\phi(x,u)\right)u^2 + 2\left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right)\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)u - \left(\frac{\partial^2}{\partial u \partial x}\,\phi(x,u)\right)\left(\frac{\partial^2}{\partial u^2}\,\phi(x,u)\right) + 2\left(\frac{\partial}{\partial u}\,\phi(x,u)\right)^2\right), \frac{\partial}{\partial u}\,\psi(x,u) = 0\right] \tag{16.4}$$

```
> MDE[15][RifMapPivs];
```

$$\left[\frac{\partial}{\partial u}\,\phi(x,u) \neq 0, \frac{\partial}{\partial x}\,\psi(x,u) \neq 0\right] \tag{16.5}$$

Example 16: Inappropriate inputs

One of technical conditions for our algorithms, is that our spaces are appropriately fibered. In particular we need to regard, dependent variables in the input system, as independent variables for the mapping system. This means we don't allow constraints of differential order 0, such as that in the example below.

```
> DE[16.1] := [diff(u(x), x) = 0, u(x)^7 + u(x) + 1 = 0];
```

$$DE_{16.1} := \left[\frac{d}{dx}\,u(x) = 0, u(x)^7 + u(x) + 1 = 0\right] \tag{17.1}$$

```
> MDE[16.1] := MapDE([DE[16.1], [[x], [u]], [[xi], [eta]]],
      ToLinearDE, [[ ], [[psi], [phi]]], [ outputvars = [[xh], [uh]]])
      :
Error, (in MapDETools:-MapDE) Nontrivial leading nonlinear
constraints found in rifsimp(R), [u(x)^7+u(x)+1 = 0], After
rifsimp R should be leading linear
```

```
>
```

We also require that the input systems are leading linear to avoid casesplitting on the input and also because Maple's symmetry routines (like all others to our knowledge) implicitly make this requirement.

```
> DE[16.2] := [diff(u(x), x, x) = 0, diff(u(x), x)^7 + diff(u(x), x)
      + 1 = 0];
```

$$DE_{16.2} := \left[\frac{d^2}{dx^2} u(x) = 0, \left(\frac{d}{dx} u(x) \right)^7 + \frac{d}{dx} u(x) + 1 = 0 \right] \tag{17.2}$$

```
> MDE[16.2] := MapDE([DE[16.2], [[x], [u]], [[xi], [eta]]],
      ToLinearDE, [[ ], [[psi], [phi]]], [ outputvars = [[xh], [uh]]])
      :
Error, (in MapDETools:-MapDE) Nontrivial leading nonlinear
constraints found in rifsimp(R), [(diff(u(x), x))^7+diff(u(x),
x)+1 = 0], After rifsimp R should be leading linear
>
>
```

MapDETools:-DifferentialHilbertFunction

The command DifferentialHilbertFunction(...) determine the differential dimension of DE. It is used as sub-routine in the MapDE algorithm in the exsitence step.

```
> DifferentialHilbertFunction(ID, s)
```

where ID is initialdata of a DE and s is a variable. The OutPut is a list of dimension, differential dimension and Differential Hilbert function of a DE.

```
> sys[1] := [diff(u(x, y), x, x, x) - u(x, y) = 0];
```

$$sys_1 := \left[\frac{\partial^3}{\partial x^3} u(x,y) - u(x,y) = 0 \right] \tag{18.1}$$

```
> rsys[1] := DEtools:-rifsimp(sys[1]);
```

$$rsys_1 := \text{table}\left(\left[Solved = \left[\frac{\partial^3}{\partial x^3} u(x,y) = u(x,y) \right] \right] \right) \tag{18.2}$$

```
> ID[1] := DEtools:-initialdata(rsys[1]);
```

$$ID_1 := \text{table}\left(\left[Infinite = \left[u(x_0, y) = \_F1(y), D_1(u)(x_0, y) = \_F2(y), D_{1,1}(u)(x_0, y) = \_F3(y) \right], Finite = [\,] \right] \right) \tag{18.3}$$

```
> ID[1][Finite];
```

$$[\,] \tag{18.4}$$

```
> DifferentialHilbertFunction(ID[1], s);
```

$$\left[\infty, 1, \frac{1}{1-s} + \frac{s}{1-s} + \frac{s^2}{1-s}\right] \tag{18.5}$$

```
>
```

```
> sys := [u(x)*(diff(diff(diff(u(x), x), x), x)) + 3*(diff(u(x), x))
    *(diff(diff(u(x), x), x))-3*((diff(u(x), x))^2 + u(x)
    *(diff(diff(u(x), x), x)) + 1)^2/(u(x)*(diff(u(x), x)) + x)-8*x
    *(u(x)*(diff(u(x), x)) + x)^4*(1 + u(x)^2 + x^2)/(u(x)^2 + x
    ^2) = 0];
```

$$sys := \left[u(x)\left(\frac{d^3}{dx^3}u(x)\right) + 3\left(\frac{d}{dx}u(x)\right)\left(\frac{d^2}{dx^2}u(x)\right) - \frac{3\left(\left(\frac{d}{dx}u(x)\right)^2 + u(x)\left(\frac{d^2}{dx^2}u(x)\right) + 1\right)^2}{u(x)\left(\frac{d}{dx}u(x)\right) + x} - \frac{8x\left(u(x)\left(\frac{d}{dx}u(x)\right) + x\right)^4\left(1 + u(x)^2 + x^2\right)}{u(x)^2 + x^2} = 0\right] \tag{18.6}$$

```
> rsys := DEtools:-rifsimp(sys) :
> ID := DEtools:-initialdata(rsys);
```

$$ID := \text{table}\left(\left[Infinite = [\,], Pivots = [\_C1\ \_C2 + x \neq 0, \_C1^2 + x^2 \neq 0, \_C1 \neq 0], Finite = \left[u(x_0) = \_C1, \mathrm{D}(u)(x_0) = \_C2, \mathrm{D}^{(2)}(u)(x_0) = \_C3\right]\right]\right) \tag{18.7}$$

```
>
> DifferentialHilbertFunction(ID, s);
```

$$[3, 0, s^2 + s + 1] \tag{18.8}$$

```
>
```

Also, It is computed in the MapDE where we need to check the linearizability of the DE.

```
> Msys := MapDE([sys, [[x], [u]], [[xi], [eta]]], ToLinearDE, [[ ],
      [[psi], [phi]]], [OutputDetails, outputvars = [[xh], [uh]]]) :

MapDE: Begin section where we compute sys S' of subalgebra L'
of L, containing solution of mapping problem.
MapDE: Single ODE so apply the Lyakhov-Gerdt-Michels ODE Test
for Existence of Linearization
MapDE: Mapping equations constructed.
MapDE: Set up ranking and mindim specification for
differential-elimination.
MapDE:  Settings for differential-elimination: MinDim = 3
```

```
ranking = [rho[1] xi eta] RifStrategy = ezcriteria = [1 .9]
MapDE: Apply differential-elimination (rifsimp) to Mapping
Equations.
MapDE: Completed differential-elimination and casesplitting of
Mapping System. Time in rifsimp = .141
MapDE: After rif ncases = 3
MapDE: Number consistent cases satisfying minimum dim condns 1
MapDE: R is linearizable by DifferentialHilbertFunction Test
since HF(R)= s^2+s+1 = s^2+s+1 =HF(P) for consistent case j =
1
MapDE: System linearizable, now attempting to integrate
mapping eqns
> Msys[IDDetSys];
```

$$\left[\left[\eta(x_0, u_0) = \_C1, \mathrm{D}_1(\eta)(x_0, u_0) = \_C2, \mathrm{D}_2(\eta)(x_0, u_0) = \_C3, \mathrm{D}_{2,2}(\eta)(x_0, u_0) = \_C4\right], \right. \tag{18.9}$$

$$\left. [\,], 3, 0, s^2 + s + 1\right]$$

Appendix B

Copyright Release

- A version of Chapter 2 has been published by proceedings of the 2019 International Symposium on Symbolic and Algebraic Computation, ACM, Pages 331-338, 2019. The electronic copyright-permission form ACM which was signed by author states: Owner shall have the right to do the following:

  - Post the Accepted Version of the Work on (1) the Author's home page, (2) the Owner's institutional repository, (3) any repository legally mandated by an agency funding the research on which the Work is based, and (4) any non-commercial repository or aggregation that does not duplicate ACM tables of contents, i.e., whose patterns of links do not substantially duplicate an ACM-copyrighted volume or issue. Non-commercial repositories are here understood as repositories owned by non-profit organizations that do not charge a fee for accessing deposited articles and that do not sell advertising or otherwise profit from serving articles.
  - Post an "Author-Izer" link enabling free downloads of the Version of Record in the ACM Digital Library on (1) the Author's home page or (2) the Owner's institutional repository.

- A version of Chapter 3 has been submitted to Mathematics in Computer Science (Revision requested 15-MAY-2019), A preprint of that is accessible in arXiv with Technical Report arXiv:1903.03727v1 [math.AP], 2019. On the publishers website, under **Copyright Transfer Statement**, they state:

  - Authors may self-archive the Authors accepted manuscript of their articles on their own websites. Authors may also deposit this version of the article in any repository, provided it is only made publicly available 12 months after official publication or later. He/she may not use the publisher's version (the final article), which is posted on SpringerLink and other Springer websites, for the purpose of self-archiving or deposit. Furthermore, the Author may only post his/her version provided acknowledgement is given to the original source of publication and a link is inserted to the published article on Springer's website. The link must be provided by inserting the DOI number of the article in the following sentence: "The final publication is available at Springer via http://dx.doi.org/[insert DOI]".

– Prior versions of the article published on non-commercial pre-print servers like arXiv.org can remain on these servers and/or can be updated with Author's accepted version. The final published version (in pdf or html/xml format) cannot be used for this purpose. Acknowledgement needs to be given to the final publication and a link must be inserted to the published article on Springer's website, by inserting the DOI number of the article in the following sentence: "The final publication is available at Springer via http://dx.doi.org/[insert DOI]." Author retains the right to use his/her article for his/her further scientific career by including the final published journal article in other publications such as dissertations and postdoctoral qualifications provided acknowledgement is given to the original source of publication.

Curriculum Vitae

Name:	Zahra Mohammadi
Post-Secondary Education and Degrees:	The University of Western Ontario London, Ontario, Canada Ph.D. Applied Mathematics, 2015 - 2019
	Chamran University Ahvaz, Iran M.Sc. Pure Mathematics (Analysis), 2005 - 2007
Related Work Experience:	Internship Maplesoft, Waterloo, Ontario, Canada May 2019 - August 2019
	Teaching Assistant The University of Western Ontario 2015 - 2019
	Research Assistant The University of Western Ontario 2015 - 2019
	Lecturer (Faculty Member) Islamic Azad University, Mahshahr, Iran 2008 - 2015

Publications

Z. Mohammadi, G. Reid, and S.-L.T. Huang.
Symmetry-based algorithms for invertible mappings of polynomially nonlinear PDS to linear PDS
Submitted to Mathematics in Computer Science (Revision requested MAY-2019)
Technical Report arXiv:1903.03727v1. 2019

Z. Mohammadi, G. Reid, and S.-L.T. Huang.
Introduction of the MapDE algorithm for determination of mappings relating differential equations
ISSAC '19, 331-338, ACM, 2019

Z. Mohammadi and O. Valero.
New fixed point results in partial quasi-metric spaces
Applied Mathematics and Nonlinear Sciences, 2(2), 415-428, 2017

Z. Mohammadi and O. Valero.
A new contribution to the fixed point theory in partial quasi-metric spaces and its applications to asymptotic complexity analysis of algorithms
Topology and its Applications. 203, 42-56, 2016

Z. Mohammadi.
Mizoguchi-Takahashis fixed point theorem concerning τ-distance
Journal of Mathematical Extension, Vol 4, No 2, 9-14, 2010

Software Packages:

`MapDETools` subpackage of `LieAlgebrasOfVectorFields` library implemented in `Maple`, https://github.com/GregGitHub57/MapDETools

Presentations

"The Lie Algebra of Vector Fields Package with applications to Mappings of Differential Equations ", Maplesoft conference, Waterloo, October 2019.

"Extension of the MapDE algorithm for mappings relating differential equations", Computer Algebra in Scientific Computing, Moscow, August 2019.

"A symbolic-numeric method to determine symmetry of approximate differential equations ", Applications of Computer Algebra, Santiago de Compostela, Spain, June 2018.

Poster Presentations

"Numerical determination of missing constraints in Dynamical systems of DAE and PDAE ", Polynomials, Kinematics, and Robotics, University of Notre Dame, Indiana, June 2017.

"Real solution of DAE and PDAE System ", Western Research Forum, Western university, January 2018.