Vector Fields and Nilpotent Lie Algebras

Matthew Grayson*
University of California, San Diego

Robert Grossman†
University of Illinois at Chicago

Abstract

We describe an infinite dimensional family of flows E with the property that the associated dynamical system

$$\dot{x}(t) = E(x(t)), \qquad x(0) \in \mathbf{R}^N$$

is explicitly integrable in closed form. These flows E are of the form $E = E_1 + E_2$, where E_1 and E_2 are the generators of a nilpotent Lie algebra, which is either free, or satisfies some relations "at a point." These flows can then be used to approximate the flows of more general types of dynamical systems.

*The author is a National Science Foundation Postdoctoral Research Fellow.
†The author is supported in part by NASA–Ames grant NAG2-513.

1 Introduction

This paper is concerned with the efficient symbolic computation of flows of ordinary differential equations of the form

$$\dot{x}(t) = F(x(t)), \qquad x(0) \in \mathbf{R}^N.$$

In other words, we want to write the corresponding trajectory $t \to x(t)$ in closed form. Of course, since this cannot be done for most differential equations, it is important to find a means of approximating a general flow by a flow which is explicitly integrable in closed form. In this paper we describe an infinite dimensional family of explicitly integrable flows E; in a companion paper [6] we give an algorithm which, given a general flow F, will, in some cases, find a flow E which is explicitly integrable in closed

form and is close to the original flow F. More generally, one could imagine approximating a flow by a flow which is explicitly integrable in closed form, flowing for a time step, finding a new approximating flow, and continuing.

The type of integrable flows discussed in this paper arise in a very simple fashion. We describe the basic idea here and refer to the paper for the details. We assume that the differential equation

$$\dot{x}(t) = E(x(t)), \qquad x(0) \in \mathbf{R}^N,$$

has the form

$$\begin{aligned}
\dot{x}_1(t) &= a_1 \\
\dot{x}_2(t) &= a_2(x_1) \\
\dot{x}_3(t) &= a_3(x_1, x_2) \\
&\vdots \qquad \vdots \\
\dot{x}_N(t) &= a_N(x_1, x_2, \ldots, x_{N-1})
\end{aligned}$$

for polynomials $a_1, \ldots, a_N$. Since the first equation can be integrated in closed form to yield x_1 as a function of time, and the jth equation depends only upon the first through $(j-1)$th equations, the entire system can be explicitly integrated in closed form.

In many examples, the vector field E defining the flow has some additional structure so that it can be written as the sum of two or more pieces. For example, suppose E depends upon two controls $t \to u_1(t)$ and $t \to u_2(t)$ and can be written as $E = u_1(t)E_1 + u_2(t)E_2$. Alternatively, suppose that E can be written as the sum of two pieces $E_1 + \epsilon E_2$, where ϵ is a parameter. Observe that if E_1 and E_2 are vector fields with polynomial coefficients which are homogeneous of degree 1, then the flow generated by E is of the form above and, hence, is explicitly integrable in closed form. See §2 and [4] for more details.

Section 2 provides some background on vector fields, flows and Lie algebras. Section 3 gives an algorithm for producing flows associated with free, nilpotent Lie algebras. Section 4 considers flows which are modeled by nilpotent Lie algebras satisfying relations "at a point". The appendix contains a brief description of the MAPLE code which we used to construct the examples.

Nilpotent approximations of this type have been used in control theory by Crouch [1] and [2], Hermes [8], [9], and [10], and Krener [13]; and in

partial differential equations by Folland [3], Hörmander [11], Rockland [14], and Stein [15].

2 Vector fields, Flows and Lie Algebras

In this section, we define homogeneous derivations and the flows they generate. One of the interests of homogeneous derivations is that their flows are integrable explicitly in closed form. Homogeneous derivations may be classified by the algebraic properties of the (Lie) algebras they generate. In the next section, we give an algorithm which yields homogeneous derivations which generate a Lie algebra of special type, called a free, nilpotent Lie algebra. In section four, we describe the analogous algorithm for derivations which satisfy relations "at a point." For background and related material, see Goodman [5], Jacobson [12], and Varadarajan [16].

Let R denote the space of polynomial functions on $\mathbf{R}^N$ and let $E_1, \ldots, E_M$ denote vector fields on $\mathbf{R}^N$ with polynomial coefficients; that, is, E_i is of the form

$$E_i = \sum_{j=1}^{N} a_i^j D_j,$$

where $a_i^j = a_i^j(x_1, \ldots, x_N) \in R$ are polynomials and $D_j = \partial/\partial x_j$. These are also called *derivations*, since they are derivations of the ring of polynomials R. Recall that a derivation E of a ring R is a map $E : R \longrightarrow R$ satisfying the identities

$$\begin{aligned} E(a+b) &= E(a) + E(b) \\ E(ab) &= aE(b) + bE(a), \end{aligned}$$

for elements a and $b \in R$. The *flow generated by the derivation* $E = \sum_{j=1}^{N} a^j D_j$, *with initial point* $x^0 = (x_1^0, \ldots, x_N^0) \in \mathbf{R}^N$ is defined to be the solution of the initial value problem

$$\begin{aligned} \dot{x}_1(t) &= a_1(x_1, \ldots, x_N) \\ &\vdots \\ \dot{x}_N(t) &= a_N(x_1, \ldots, x_N), \end{aligned}$$

$$x_1(0) = x_1^0, \quad \ldots, \quad x_N(0) = x_N^0.$$

Let $t \to x(t)$ denote the solution to this initial value problem. We say that the point $x^1 \in \mathbf{R}^N$ is reached in time s in case $x^1 = x(s)$. Notice that

flows form a semigroup; that, is, if we can flow from $x^0 \in \mathbf{R}^N$ to the point x^1, and from x^1 to the point x^2, then we can flow from x^0 to the point x^2.

We are interested in *homogeneous derivations of weight m*. We define these informally; for a more formal definition, see [5]. First, assign weights to the coordinate variables: say x_1 is of weight ρ_1, ..., x_N is of weight ρ_N. Second, assign weights to monomials, by defining the weight of the monomial to be

$$\mathrm{wt}(x_1^{e_1} \cdots x_N^{e_N}) = {\rho_1}^{e_1} + \cdots + {\rho_N}^{e_N}.$$

Third, a polynomial is called homogeneous of weight m in case it is a sum of monomials, each of weight m. Fourth, define the weight of the coordinate derivation to be

$$\mathrm{wt}(D_j) = \partial/\partial x_j = -\rho_j,$$

and the weight of the derivation $a(x)D_j$ to be $\mathrm{wt}(a(x)) - \mathrm{wt}(D_j)$. Fifth, a general derivation $E_i = \sum_{j=1}^N a_i^j D_j$ is of weight m in case it is the sum of derivations, each of weight m. Sixth, the weight of the second order differential operator is defined as

$$\mathrm{wt}(a(x)D_iD_j) = \mathrm{wt}(a(x)) - \mathrm{wt}(D_i) - \mathrm{wt}(D_j),$$

and the weight of higher order differential operators are defined in an analogous fashion.

If E is a derivation of weight 1 on $\mathbf{R}^N$, then the flow

$$\dot{x}(t) = E(x(t)), \qquad (0) = x^0 \in \mathbf{R}^N$$

is explicitly integrable in closed form. This is easy to prove by induction, since the coefficient of $\partial/\partial x_i$, for example, depends only upon variables which have already been computed.

Given M derivations $E_1, \ldots, E_M$, we define the *Lie algebra generated by* $E_1, \ldots, E_M$ to be the smallest real subspace of derivations of R which is closed under the formation of Lie brackets

$$[E, F] = EF - FE;$$

this is denoted

$$\mathrm{g}(E_1, \ldots, E_M).$$

The *length* of a Lie bracket is defined inductively as follows: the generators all have length 1. If E is a Lie bracket of length less than or equal to k and F is a Lie bracket of length less than or equal to l, then $[E, F]$ has length

less than or equal to $k + l$. The algorithms in this paper use the length of a Lie bracket to define homogeneous derivations in the following way. A vector space decomposition of $\mathbf{R}^N = V_1 \oplus \cdots \oplus V_r$ is defined by setting

$$V_l = \text{span } \{E : E \text{ is a Lie bracket of length } \leq l\}.$$

If $e_j, \ldots, e_{j'}$ is a basis of V_l, then each coordinate x_j in the dual basis of coordinates $x_j, \ldots, x_{j'}$ is defined to have weight l, and homgenous derivations are defined as above.

A Lie algebra is called *nilpotent of step* r in case any Lie bracket of length greater than r is zero. Note that if $E_1, \ldots, E_M$ are homogeneous of weight 1, then the Lie algebra they generate is nilpotent of step r and any bracket of length l is homogeneous of weight l. This is because if F_1 is a derivation homogeneous of weight k and F_2 is a derivation homogeneous of weight l, then $[F_1, F_2]$ is homogeneous of weight $k + l$. A Lie algebra is *free* in case it satisfies as few relations as possible. Note that a Lie algebra always satisfies some relations, since for example

$$[E_1, E_2] = -[E_2, E_1]$$

and

$$[[E_1, E_2], E_3] + [[E_2, E_3], E_1] + [[E_3, E_1], E_2] = 0.$$

Also, there will be additional relations that are a consequence of these. A Lie algebra is free in case any relation is a consequence of relations that are of the form above. Rather than try to make this precise, we give a basis for a free, nilpotent Lie algebra that is due to M. Hall.

Definition 2.1 *Each element of the Hall basis is a monomial in the generators and is defined recursively as follows. The generators* $E_1, \ldots, E_M$ *are elements of the basis and of length* 1. *If we have defined basis elements of lengths* $1, \ldots, r-1$, *they are simply ordered so that* $E < F$ *if* $length(E) < length(F)$. *Also if* $length(E) = s$ *and* $length(F) = t$ *and* $r = s + t$, *then* $[E, F]$ *is a basis element of length* r *if:*

1. E *and* F *are basis elements and* $E > F$, *and*
2. *if* $E = [G, H]$, *then* $F \geq H$.

3 Free, Nilpotent Lie Algebras

In this section we provide motivation by reviewing an algorithm, whose input consists of a rank r, and whose output consists of two vector fields

E_1 and E_2 on $\mathbf{R}^N$ which generate a Lie algebra which is isomorphic to the free, nilpotent Lie algebra $\mathrm{g}_{2,r}$ on two generators of rank r. Here N is the dimension of the Lie algebra. The proof can be found in [4]. In the next section we consider the analogous problem when the Lie algebra satisfies relations. Recall that the flows of these vector fields can all be integrated explicitly in closed form.

Fix the rank $r \geq 1$ of the free, nilpotent Lie algebra $\mathrm{g}_{2,r}$, and number the basis elements for the Lie algebra by the ordering from Definition 2.1, *i.e.*, $E_3 = [E_2, E_1]$, $E_4 = [E_3, E_1]$, $E_5 = [E_3, E_2]$, *etc.* Consider a basis element E_k as a bracket in the lower order basis elements, $[E_{i_1}, E_{j_1}]$, where $i_1 > j_1$. If we repeat this process with E_{i_1}, we get $E_k = [[E_{i_2}, E_{j_2}], E_{j_1}]$, where $j_2 \leq j_1$ by the Hall basis conditions. Continuing in this fashion, we get

$$E_k = [[\cdots[[E_{i_m}, E_{j_m}], E_{j_{m-1}}], \cdots, E_{j_2}], E_{j_1}],$$

where $i_m = 2$, $j_m = 1$, and $j_{n+1} \leq j_n$ for $1 \leq n \leq m-1$. This defines a partial ordering of the basis elements. We say that E_k is a *direct descendant* of each E_{i_j}, and we indicate this by writing $k \succ i_j$.

Define monomials $P_{2,k}$ inductively by $P_{2,k} = -x_j P_{2,i}/(\deg_j P_{2,i} + 1)$, whenever $E_k = [E_i, E_j]$ is a Hall basis element, and where $\deg_j P$ is the highest power of x_j which divides P. If $l \succeq m$, then define $P_{l,m}$ to be the quotient $P_{2,m}/P_{2,l}$. Note that $P_{i,i} = 1$. The coefficients of these monomials guarantee that $-\frac{\partial}{\partial x_j} P_{i,l} = P_{k,l}$ whenever $i \prec k \prec l$. For example, if $E_k = [[[[[[[E_2, E_1], E_1], E_1], E_2], E_4], E_4], E_7]$, and if $E_i = [[[E_2, E_1], E_1], E_1]$, then

$$P_{2,k} = -\frac{x_1^3 x_2 x_4^2 x_7}{3!2!}, \quad \text{and} \quad P_{i,k} = \frac{x_2 x_4^2 x_7}{2!}.$$

The following theorem summarizes the properties of the generators.

Theorem 3.1 *Fix $r \geq 1$ and let N denote the dimension of the free, nilpotent Lie algebra on 2 generators of rank r. Then the vector fields*

$$\begin{aligned} E_1 &= \frac{\partial}{\partial x_1} \\ E_2 &= \frac{\partial}{\partial x_2} + \sum_{i \succ 2} P_{2,i} \frac{\partial}{\partial x_i} \end{aligned}$$

have the following properties:

1. *they are homogeneous of weight one with respect to the grading*

$$\mathbf{R}^N = V_1 \oplus \cdots \oplus V_r,$$

where V_i = the span of Hall basis elements of length i.

2. *the Hall basis elements E_k they generate satisfy $E_k(0) = \frac{\partial}{\partial x_k}$;*
3. *the flow $E_1 + E_2$ is explicitly integrable in closed form;*
4. *the Lie algebra they generate is isomorphic to $g_{2,r}$.*

4 Vector Field Models With Relations at the Origin

In the previous section, we gave an algorithm which generates the explicitly integrable flows associated with free, nilpotent Lie algebras. Our eventual goal is to find as large a class as possible of explicitly integrable flows. In this section, we generalize the algorithm of the previous section by giving an algorithm which generates the explicitly integrable flows associated with nilpotent Lie algebras satisfying "relations at a point."

Let $\mathcal{E}_1, \ldots, \mathcal{E}_N$ denote the basis of the free, nilpotent Lie algebra on two generators $\mathcal{E}_1$ and $\mathcal{E}_2$ of rank r and dimension N, and let $E_1, \ldots, E_N$ denote the vector fields obtained by replacing each occurrence of the generators $\mathcal{E}_1$ and $\mathcal{E}_2$ in $\mathcal{E}_i$ by E_1 and E_2. Recall that a *homogeneous relation* is a relation of the form

$$\mathcal{E}_i = \sum_{j=1}^{N} \tilde{r}_{ij} \mathcal{E}_j,$$

where $\tilde{r}_{ij} = 0$, unless $\mathrm{wt}(E_i) = \mathrm{wt}(E_j)$, and $\tilde{r}_{ij}$ are scalars. We are interested in quotient algebras g defined by imposing homogeneous relations among the generators. Our eventual goal is to find a vector field model for an arbitrary quotient algebra g; that is, vector fields $E_1, \ldots, E_N$ corresponding to the Lie elements $\mathcal{E}_1, \ldots \mathcal{E}_N$ and satisfying the same bracket relations as the quotient algebra. The reason is simple: many questions about the structure of the algebra and about the associated dynamical system $\dot{x}(t) = E_1(x(t)) + E_2(x(t))$ are reduced to easily computed questions about the flows of the E_j's.

In this paper, we solve just a "local" version of this problem. To describe this problem, we fix the basis $\mathcal{E}_1, \ldots, \mathcal{E}_N$ in the domain, and the basis $D_1, \ldots, D_N$ in the range, and use the homogeneous relations to induce a map $R : \mathbf{R}^N \to \mathbf{R}^N$. The problem we solve is to find vector fields E_1 and E_2 such that the associated Lie brackets evaluated at the origin $E_1(0), \ldots, E_N(0)$ are the columns of the matrix R. We prove the theorem assuming the following technical hypothesis.

Assumption. The map R is the identity when restricted to the subspace spanned by the derivations $\mathcal{E}_i$ of weight less than or equal to $r/2$. In other words, the components r_{ij} of R satisfy $r_{ij} = \delta_{ij}$, whenever the weights of $\mathcal{E}_i$ and $\mathcal{E}_j$ are both less than $r/2$. Here δ_{ij} is the Kronecker delta.

Since the number of basis elements grows very quickly with the weight, this requirement affects very few of them. This restriction reduces the arguments essentially to the proof of the free case; under this condition, the images of the free nilpotent Lie algebra vector fields E_1 and E_2 under the map R generate the desired quotient algebra with relations at the origin. Indeed, the same map R takes the basis elements for the free nilpotent Lie algebra to a spanning set for the quotient algebra. Of course, in general, the operations of bracket and non-isomorphic linear mapping do not commute. With the above condition, it follows that they at least commute at the origin.

We conclude this introduction with a simple remark. Let B denote the indices in the index set $\{1, \ldots, N\}$ which are of weight less than $r/2$; and let A denote the remaining indices. Also, let $\tilde{R} = (\tilde{r}_{ij})$ denote the matrix defining the homogeneous relations satisfied by the generators $\mathcal{E}_i$, and assume that

1. $\tilde{r}_{ij} = \delta_{ij}, \qquad$ for $i, j \in B$

2. for each fixed $i \in A$, $r_{ij} = 0$, for all $j \in A$ and $j \neq i$.

It is easy to see that, with these hypotheses, the matrix of relations $\tilde{R}$ is equal to the matrix R defined above using the bases $\mathcal{E}_i$ and D_j.

Define polynomials $P_{2,k}$ inductively by $P_{2,k} = -x_j P_{2,i}/(\deg_j P_{2,i} + 1)$, whenever $E_k = [E_i, E_j]$ is a Hall basis element, and where $\deg_j P$ is the highest power of x_j which divides P. If $l \succeq m$, then define $P_{l,m}$ to be the quotient $P_{2,m}/P_{2,l}$. Note that $P_{i,i} = 1$.

Theorem 4.1 *Choose $R = (r_{ij})$ satisfying the assumption above, and set*

$$E_1 = \frac{\partial}{\partial x_1}$$

and

$$E_2 = \sum_{i \geq 2} P_{2,i} \sum_{j=1}^{N} r_{ji} \frac{\partial}{\partial x_j}.$$

Then the Lie brackets E_j have the properties:

1. *they are homogeneous of weight* 1 *with respect to the grading*

$$\mathbf{R}^N = V_1 \oplus \cdots \oplus V_r,$$

where V_i = *the span of Hall basis elements of length* i*;*

2. *the flows of the system*

$$\dot{x}(t) = E_1(x(t)) + E_2(x(t)), \qquad x(0) = x^0$$

are explicitly integrable in closed form;

3. *at the origin, the vector fields satisfy*

$$E_i(0) = \sum_{j=1}^{N} r_{ji} \frac{\partial}{\partial x_j},$$

for all $1 \leq i \leq N$*, and so they satisfy the desired relations.*

Proof of theorem. By construction the vector fields are homogeneous of weight 1; therefore their flows are explicitly integrable in closed form; hence, we need only prove assertion three. As in the proof of the free case, this is proved using a lemma which is actually stronger than the theorem and which illuminates the structure of the vector fields E_j. We must introduce the minimum order of a polynomial which was defined in [4].

Definition 4.2 *Let* A *be any non-constant monomial in the variables* x_1, $\ldots$, x_n. *Define the minimum order of* A *by* $m(A) = \min\{j : x_j | A\}$. *If* A *is a polynomial with no constant term, then define* $m(A)$ *to be the maximum of the minimum orders of the monomials of* A.

Lemma 4.3 *If* $E_k = [E_i, E_j]$, *then*

$$E_k = \sum_{l \succeq k} P_{k,l} \sum_m r_{ml} \frac{\partial}{\partial x_m} + \sum_n Q_{k,n} \frac{\partial}{\partial x_n},$$

where $j \leq m(P_{k,l}) < k$, *and* $Q_{k,n}$ *is a polynomial with no constant term with minimum order* $< j$.

Assertion three follows easily from the lemma. The only coefficients which do not vanish at the origin are from the first term when $l = k$. This implies $E_k(0) = \sum_{m=1}^{N} r_{mk} \frac{\partial}{\partial x_m}$, as desired.

Proof of lemma. The proof is by induction. Direct computation yields

$$E_3 = \sum_{i \geq 3} P_{3,i} \sum_j r_{ji} \frac{\partial}{\partial x_j},$$

since $P_{2,i} = -x_1 P_{3,i}$ for all $i \geq 3$.

Consider the parents of E_k. By the inductive hypothesis:

$$E_i = \sum_{a \succeq i} P_{i,a} \sum_b r_{ba} \frac{\partial}{\partial x_b} + \sum_c Q_{i,c} \frac{\partial}{\partial x_c},$$

and

$$E_j = \frac{\partial}{\partial x_j} + \sum_{d \succ j} P_{j,d} \sum_e r_{de} \frac{\partial}{\partial x_e} + \sum_f Q_{j,f} \frac{\partial}{\partial x_f},$$

since the weight of $\mathcal{E}_j < \rho/2$, and so $E_j(0) = \frac{\partial}{\partial x_j}$.

Now $E_i = [E_p, E_q]$ and so $q \leq j$ by the basis condition. Therefore, we have the relations:

$$q \leq m(P_{i,a}) < i \quad \text{and} \quad m(Q_{i,c}) < q \leq j.$$

And trivially,

$$m(P_{j,d}) < j \quad \text{and} \quad m(Q_{j,f}) < j.$$

Examine the terms in the bracket of E_i and E_j.

$$\begin{aligned}
[E_i, E_j] \;=\; & \sum_{\substack{a \succeq i \\ d \succ j}} \sum_{\substack{b \\ e}} P_{i,a} r_{ba} r_{ed} \left(\frac{\partial}{\partial x_b} P_{j,d}\right) \frac{\partial}{\partial x_e} + \sum_{a \succeq i} \sum_{\substack{b \\ f}} P_{i,a} r_{ba} \left(\frac{\partial}{\partial x_b} Q_{j,f}\right) \frac{\partial}{\partial x_f} \\
& + \sum_{\substack{c \\ d \succ j}} \sum_e Q_{i,c} r_{ed} \left(\frac{\partial}{\partial x_c} P_{j,d}\right) \frac{\partial}{\partial x_e} + \sum_c \sum_f Q_{i,c} \left(\frac{\partial}{\partial x_c} Q_{j,f}\right) \frac{\partial}{\partial x_f} \\
& - \sum_{a \succeq i} \sum_b r_{ba} \left(\frac{\partial}{\partial x_j} P_{i,a}\right) \frac{\partial}{\partial x_b} \\
& - \sum_c \left(\frac{\partial}{\partial x_j} Q_{i,c}\right) \frac{\partial}{\partial x_c} \\
& - \sum_{\substack{d \succ j \\ a \succeq i}} \sum_{\substack{e \\ b}} P_{j,d} r_{ed} r_{ba} \left(\frac{\partial}{\partial x_e} P_{i,a}\right) \frac{\partial}{\partial x_b} - \sum_{d \succ j} \sum_{\substack{e \\ c}} P_{j,d} r_{ed} \left(\frac{\partial}{\partial x_e} Q_{i,c}\right) \frac{\partial}{\partial x_c} \\
& - \sum_{\substack{f \\ a \succeq i}} \sum_b Q_{j,f} r_{ba} \left(\frac{\partial}{\partial x_f} P_{i,a}\right) \frac{\partial}{\partial x_b} - \sum_f \sum_c Q_{j,f} \left(\frac{\partial}{\partial x_f} Q_{i,c}\right) \frac{\partial}{\partial x_c}.
\end{aligned}$$

If A and B are any monomials with $m(A) < j$ then for any m, $A\frac{\partial}{\partial x_m}B$ either vanishes or has minimum order less than j. This, and the fact that $r_{ij} = 0$ unless E_i and E_j have the same weight, imply that the non-zero terms in the second, fifth, and sixth lines have minimum orders $< j$. If A is any monomial satisfying $m(A) < j$ and if $m \geq j$, then $\frac{\partial}{\partial x_m}A$ either vanishes or has minimum order less than j. This implies that the non-zero terms in the first and fourth lines have minimum orders $< j$. The remaining terms are

$$-\sum_{a \succeq i}\sum_{b} r_{ba}(\frac{\partial}{\partial x_j}P_{i,a})\frac{\partial}{\partial x_b}.$$

When $a = i$, this term vanishes. If $m(P_{i,a}) = j$, then either $k = a$, in which case

$$-\sum_{b} r_{bk}(\frac{\partial}{\partial x_j}P_{i,k})\frac{\partial}{\partial x_b} = \sum_{b} r_{bk}\frac{\partial}{\partial x_b},$$

or $a \succ k$, and

$$-\sum_{b} r_{ba}(\frac{\partial}{\partial x_j}P_{i,a})\frac{\partial}{\partial x_b} = \sum_{b} r_{ba}P_{k,a}\frac{\partial}{\partial x_b}.$$

If $m(P_{i,a}) < j$, then $\frac{\partial}{\partial x_j}P_{i,a}$ is either zero, or it has minimum order less than j. If $m(P_{i,a}) > j$, then $\frac{\partial}{\partial x_j}P_{i,a} = 0$. We conclude that

$$-\sum_{a \succeq i}\sum_{b} r_{ba}(\frac{\partial}{\partial x_j}P_{i,a})\frac{\partial}{\partial x_b} = \sum_{l \succeq k}\sum_{m} r_{ml}P_{k,l}\frac{\partial}{\partial x_l} + \sum_{g} Q_{k,g}\frac{\partial}{\partial x_g},$$

where $m(Q_{k,g}) < j$. This proves the lemma.

A Appendix: Maple Code

The algorithms described in this paper were the result of computer experimentation using the symbolic packages MACSYMA and MAPLE. In this appendix, we give examples of some of the very simple MAPLE code that we wrote. Figure 1 gives the MAPLE code to compute Lie brackets, and Figures 2, 3 and 4 give the MAPLE code to compute vector fields which satisfy relations at a point. This code is illustrated by computing two vector fields which satisfy the relation

$$E_6(0) + 2E_7(0) = E_8(0).$$

```
brac:= proc(v,w)
  #returns the Lie bracket of the vector fields v and w
  local temp,t,i,j,l;

  if not type(v,'vector') then
      ERROR('Not a vector')
  elif not type(w,'vector') then
      ERROR('Not a vector')
  fi;
  if vectdim(v)<>vectdim(w) then
    ERROR('Different dimensions!')
  fi;

  temp:=array(1..vectdim(v));
  t:=array(1..vectdim(v));

  for j from 1 to vectdim(v) do
    for i from 1 to vectdim(v) do
      t[i]:= v[i]*diff(w[j],x[i])-w[i]*diff(v[j],x[i])
    od;
    temp[j] := sum(t[l],l=1..vectdim(v));
  od;
  eval(temp);
end;
```

Figure 1: Maple code to compute Lie brackets.

Finally, in Figure 7, we give the code which will flow along vector fields, and illustrate it by flowing along the sum of the two vector fields already computed.

Our example is a quotient algebra of $g_{2,4}$. This has a basis given by E_1, E_2 (the generators), $E_3 = [E_2, E_1]$, $E_4 = [E_3, E_1]$, $E_5 = [E_3, E_2]$, $E_6 = [E_4, E_1]$, $E_7 = [E_4, E_2]$, and $E_8 = [E_5, E_1]$. We impose the relation that

$$E_7(0) + 2E_8(0) = E_6(0).$$

The r matrix, followed by the two vector fields and their brackets is in Figures 5 and 6. The result of flowing along the $E_1 + E_2$ for time τ using the procedure in Figure 7 is displayed in Figure 8.

```
HallRelations := proc(rho,maxdim)
  #rho is the maximum weight, maxdim is an upper bound for the dimension
  local left,right,i,j,a,b,w,w0,wptr,k,poly;

  left:= array(1..maxdim,sparse,[(3)=2]);
  right:= array(1..maxdim,[(3)=1]);
  poly:= array(1..maxdim,1..maxdim,sparse,[(3,1)=1]);
  w:=array(1..maxdim,sparse,[(1)=1,(2)=1,(3)=2]);
  wptr:=array(1..rho+1,sparse,[(1)=1,(2)=3,(3)=4]);
  #wptr(i) is the index of the first element of weight i.
```

Figure 2: Maple code to compute vector fields satisfying relations at point: Part 1.

References

[1] P. E. Crouch, *Dynamical realizations of finite Volterra series,* SIAM J. Control Optim., **19** (1981), 177–202.

[2] P. E. Crouch *Graded vector spaces and applications to the approximations of nonlinear systems,* to appear.

[3] G. B. Folland and E. M. Stein, *Estimates for the $\bar{\partial}_b$ complex and analysis of the Heisenberg group,* Comm. Pure Appl. Math. **27** (1974), 429–522.

[4] M. Grayson and R. Grossman, *Models for free, nilpotent lie algebras,* Center for Pure and Applied Mathematics, University of California, Berkeley, PAM 397.

[5] R. W. Goodman, "Nilpotent Lie Groups: Structure and Applications to Analysis," Springer-Verlag, New York, 1976, Lecture Notes in Mathematics, No. 562.

[6] R. Grossman and R. G. Larson, *Solving nonlinear equations from higher order derivations in linear stages,* Center for Pure and Applied Mathematics, University of California, Berkeley, PAM 396.

[7] M. Hall, *A basis for free Lie rings and higher commutators in free groups,* Proc. Amer. Math. Soc., 1 (1950), 575–581.

[8] H. Hermes, *Control systems which generate decomposible Lie algebras,* J. Diff. Eqns., 44 (1982), 166–187.

```
l:= 4:
print(e.3=[e.2,e.1]);
for w0 from 3 to rho do
                    #w0 is the weight of the produced brackets
  for i from round(w0/2) to (w0-1) do
                    #i is the weight of the left parent
    for j from wptr[i] to (wptr[i+1]-1) do
                    #j is the index of the left parent
      for k from wptr[w0-i] to (wptr[w0-i+1]-1) do
                    #k is the index of the right parent
        if k < right[j] then next fi;
                    #These are the Hall Conditions
        if j <= k then break fi;
        left[l]:= j;
                    #assign the left and right parents
        right[l]:= k;
                    #to the new bracket.
        print(e.l=[e.j,e.k]);
        w[l] := w[j]+w[k];
                    #This is its weight.
        for m from 1 to k-1 do
                    #This loop assigns the multiindex for the coefficient.
          poly[l,m]:= poly[j,m] od;
        poly[l,k]:= poly[j,k]+1;
        l:= eval(l+1);
      od;
    od;
   od;
   wptr[w0+1]:= l;
od;
```

Figure 3: Maple code to compute vector fields satisfying relations at point: Part 2.

```
  e1:=array(1..wptr[rho+1]-1,sparse,[(1)=1]):
  e2:=array(1..wptr[rho+1]-1,sparse,[(2)=1]);
  for i from 2 to rho do
                    #Sum over the weights
    for j from wptr[i] to wptr[i+1]-1 do
                    #Sum over elements of the same weight
      for q from wptr[i] to wptr[i+1]-1 do
        e2[j]:= eval(e2[j])+
            (product((-x[p])^poly[q,p]/(poly[q,p]!), p=1..right[q])*r[j,q])
      od;
    od;
  od;
  print(e1=eval(e1));
  print(e2=eval(e2));
  for i from 3 to wptr[rho+1]-1 do
    a:= left[i]:
    b:=right[i]:
    e.i:= brac(e.a,e.b);
    print(e.i=eval(''));
  od;
end;
```

Figure 4: Maple code to compute vector fields satisfying relations at point: Part 3.

```
r := array ( 1 .. 8, 1 .. 8,
                              [1, 0, 0, 0, 0, 0, 0, 0]
                              [0, 1, 0, 0, 0, 0, 0, 0]
                              [0, 0, 1, 0, 0, 0, 0, 0]
                              [0, 0, 0, 1, 0, 0, 0, 0]
                              [0, 0, 0, 0, 1, 0, 0, 0]
                              [0, 0, 0, 0, 0, 0, 0, 0]
                              [0, 0, 0, 0, 0, 1, 1, 0]
                              [0, 0, 0, 0, 0, 2, 0, 1]
)
>
HallRelations(4,8);
                                   e3 = [e2, e1]
                                   e4 = [e3, e1]
                                   e5 = [e3, e2]
                                   e6 = [e4, e1]
                                   e7 = [e4, e2]
                                   e8 = [e5, e2]
```

Figure 5: The relation matrix, the vector fields, and their brackets: part 1.

```
e1 = (array ( sparse, 1 .. 8, [1, 0, 0, 0, 0, 0, 0, 0]))

e2 = (array ( sparse, 1 .. 8,

                          2                                3          2
[0, 1, - x[1], 1/2 x[1]  , x[1] x[2], 0, - 1/6 x[1]  - 1/2 x[1]  x[2],

            3                 2
   - 1/3 x[1]  - 1/2 x[1] x[2] ]))

e3 = (array ( 1 .. 8,
                                          2                    2          2
   [0, 0, 1, - x[1], - x[2], 0, 1/2 x[1]  + x[1] x[2], x[1]  + 1/2 x[2] ]))

e4  = (array ( 1 .. 8,

                [0, 0, 0, 1, 0, 0, - x[1] - x[2], - 2 x[1]]))

e5 = (array ( 1 .. 8, [0, 0, 0, 0, 1, 0, - x[1], - x[2]]))

e6 = (array ( 1 .. 8, [0, 0, 0, 0, 0, 0, 1, 2]))

e7 = (array ( 1 .. 8, [0, 0, 0, 0, 0, 0, 1, 0]))

e8 = (array ( 1 .. 8, [0, 0, 0, 0, 0, 0, 0, 1]))
```

Figure 6: The relation matrix, the vector fields, and their brackets: part 2.

```
flow:=proc(v,p0)
  #flows returns the point reached after flowing for time tau
  #along the vector field v from the point p0

  local xt,r,x1,i,j,n;

  n := vectdim(v);
  xt := array(1..n);
  x1 := array(1..n);
  r := array(1..n);
  r[1]:=v[1];
  for i from 1 to n-1 do
    xt[i] := int(r[i],t)+p0[i];
    r[i+1]:= subs( (x[j]=xt[j]) $ j=1..i,v[i+1]);
  od;
  xt[n] := int(r[n],t)+p0[n];
  for i from 1 to n do
    x1[i] := subs(t=tau,xt[i])
  od;
  eval(x1);
end;
```

Figure 7: Maple code to flow along a vector field.

```
>
 flow(add(e1,e2),array(1..8,sparse));

    array ( 1 .. 8,
   [tau, tau, - 1/2 tau , 1/6 tau , 1/3 tau , 0, - 1/6 tau , - 5/24 tau ])
>
 subs(tau=1,'');

    array ( 1 .. 8, [1, 1, -1/2, 1/6, 1/3, 0, -1/6, -5/24])
```

Figure 8: The result of flowing along the vector field $e1 + e2$.

[9] H. Hermes, *Nilpotent approximations of control systems and distributions,* SIAM J. Control Optim., **24** (1986), 731–736.

[10] H. Hermes, A. Lundell, and D. Sullivan, *Nilpotent bases for distributions and control systems,* J. Diff. Equations, **55** (1984), 385–400.

[11] L. Hörmander, *Hypoelliptic second order differential equations,* Acta. Math., **119** (1968), 147–171.

[12] N. Jacobson, "Lie Algebras," John Wiley and Sons, New York, 1962.

[13] A. Krener, *Bilinear and nonlinear realizations of input-output maps,* SIAM J. Control Optim., **13** (1975), 827–834.

[14] C. Rockland, *Intrinsic nilpotent approximation,* Acta Applicandae Math. **8** (1987), 213-270.

[15] L. P. Rothschild and E. M. Stein, *Hypoelliptic differential operators and nilpotent groups,* Acta. Math., **37** (1977), 248–315.

[16] V. S. Varadarajan, "Lie Groups, Lie Algebras, and their Representations," Prentice-Hall, Inc., Englewood Cliffs, 1974.