The symbolic computation of vector field expressions

Robert Grossman[*] and Richard Larson[†]
University of Illinois at Chicago

February, 1991

This is a draft of a paper which later appeared in Algebraic Computing in Control 1991, edited by G. Jacob and F. Lamnabhi-Lagarrigue, Springer-Verlag, Berlin, 1991, pp. 1-10.

Abstract

This is an expository paper explaining how trees can be used to compute effectively the vector field expressions which arise in nonlinear control theory. It also describe the mathematical structure that sets of trees carry.

1 Introduction

This is an expository paper explaining how trees can be used to compute effectively the vector field expressions which arise in nonlinear control theory. It also describe the mathematical structure that sets of trees carry.

This paper is based upon [7] and [15]. It also has been influenced by joint work with P. Crouch described in [5] and [6]. The mathematical structure underlying sets of trees was worked out in [10]. This exposition derives, in part, from [8], which is an expository account of symbolic expressions which arise in the study of differential equations. The figures are taken from [9].

[*]This research is supported in part by NASA grant NAG2-513, NSF Grant DMS-8904740, and the Laboratory for Advanced Computing.

[†]This research is supported by NSF Grant DMS-8904740.

Figure 1: The trees associated with the vector fields E_j.

2 The basic idea

In this section, we describe the basic idea of how trees can be used to organize computations involving vector fields following [7] and [8]. Consider a control system

$$\dot{x}(t) = E_1(x(t)) + u_1(t)E_2(x(t)) + u_2(t)E_3(x(t)), \quad x(0) = x^0 \in \mathbf{R}^N, \tag{1}$$

where E_1, E_2 and E_3 are vector fields defined in a neighborhood of $x^0 \in \mathbf{R}^N$ and $t \mapsto u_i(t)$ are controls. Our goal is to describe a class of algorithms for the effective symbolic computation of expressions built from the vector fields that describe the local behavior of the control system. These expressions include iterated Lie brackets and the generating series of the system.

The starting point is to assign trees to vector fields as illustrated in Figure 1, and then to impose a multiplication on the trees which is compatible with the composition of vector fields. Assume that the vector fields E_j have the form:

$$E_j = \sum_{\mu=1}^{N} a_j^\mu D_\mu, \quad j = 1, 2, 3, \quad j = 1, \ldots, M, \tag{2}$$

where a_j^μ are smooth functions on $\mathbf{R}^N$ and $D_\mu = \partial/\partial x_\mu$. Now

$$E_2 \cdot E_1 = \sum b_j (D_j a_i) D_i + \sum b_j a_i D_j D_i$$

and $E_3 \cdot E_2 \cdot E_1$ is equal to

$$\sum a_3^{\mu_k} (D_k a_2^{\mu_j})(D_j a_1^{\mu_i}) D_i + \sum a_3^{\mu_k} a_2^{\mu_j} (D_k D_j a_1^{\mu_i}) D_i + \sum a_3^{\mu_k} a_2^{\mu_j} (D_j a_1^{\mu_i}) D_k D_i$$
$$+ \sum a_3^{\mu_k} a_2^{\mu_j} a_1^{\mu_i} D_k D_j D_i + \sum a_3^{\mu_k} a_2^{\mu_j} (D_k a_1^{\mu_i}) D_j D_i + \sum a_3^{\mu_k} a_1^{\mu_i} (D_k a_2^{\mu_j}) D_j D_i. \tag{3}$$

Here the sum is for $i, j, k = 1, \ldots, N$ and hence involves $O(N^3)$ differentiations. It is convenient to keep track of the terms that arise in this way using labeled trees: we indicate in Figure 2 the trees that are associated with the six sums in this expression.

Figure 2: The trees associated with Equation 3.

An iterated Lie bracket such as

$$[E_3, [E_2, E_1]] = E_3E_2E_1 - E_3E_1E_2 - E_2E_1E_3 + E_1E_2E_3 \qquad (4)$$

gives rise in this fashion to 24 trees corresponding to the $24N^3$ differentiations that a naive computation of this expression requires. On the other hand, 18 of the trees cancel, saving us from computing $18N^3$ terms. We are left with $6N^3$ terms of the form $(junk)D_{\mu_1}$. A careful examination of this correspondence between labeled trees and expressions involving the E_j's shows that the composition of the vector fields E_j's, viewed as first order differential operators, corresponds to a multiplication on trees. This multiplication is illustrated in Figure 3. It turns out that this construction yields an algebra, which we call the *algebra of Cayley trees.*

Let R denote a space of smooth, that is C^∞, observation functions and let A denote the space of differential operators generated by E_1 and E_2. For many applications it is important to have efficient algorithms to compute the actions of these differential operators on observation functions:

$$p \cdot f, \quad p \in A, \quad f \in R.$$

For example, the generating series associated to the system is an infinite sum of terms of this type. Depending upon the representation of the function f, additional cancellations can be expected.

Figure 3: An example of multiplying two trees.

3 Bialgebras

As we will see in the next section, the space of trees has the structure of a bialgebra, which gives the ring of smooth test functions R the structure of a module algebra. By exploiting this algebraic structure, it is possible to derive more efficient algorithms. In this section we review the basic facts about bialgebras and module algebras which will we will need, following [5] and [6].

At the end of this section, we give the definition of *differentially produced* which is fundamental the theorem of Section 6.

Let k denote any field of characteristic 0. By an *algebra* we mean a vector space A over the field k with an associative multiplication and unit. The multiplication can be represented by a linear map $\mu : A \otimes_k A \to A$; the unit can be represented by a linear map $k \to A$ (the map sending $1 \in k$ to $1 \in A$). The facts that the multiplication is associative, and that $1 \in A$ is a unit, can be expressed by the commutativity of certain diagrams. For example, the commutativity of the diagram

$$\begin{tikzcd}
A \otimes_k A \otimes_k A \arrow[r] \arrow[d] & A \otimes_k A \arrow[d] \\
A \otimes_k A \arrow[r] & A
\end{tikzcd}$$

where the upper horizontal arrow is the map $\mu \otimes I$, the left vertical arrow is the map $I \otimes \mu$, and the remaining two arrows are the map μ, expresses the associativity of multiplication.

The dual notion to an algebra is a *coalgebra*: a vector space C over the field k together with a coassociative coproduct $\Delta : C \to C \otimes_k C$ and a counit $\epsilon : C \to k$. The fact that Δ is coassociative and that ϵ is a counit is expressed by diagrams which are dual to the diagrams which express the facts that the multiplication of an algebra is associative, and that $1 \in A$ is a unit: they are the same diagrams, with the direction of all arrows reversed. For example, coassociativity is expressed by the commutativity of the diagram

$$\begin{tikzcd}
C \otimes_k C \otimes_k C & C \otimes_k C \arrow[l] \\
C \otimes_k C \arrow[u] & C \arrow[l] \arrow[u]
\end{tikzcd}$$

where the upper horizontal arrow is the map $\Delta \otimes I$, the left vertical arrow is the map $I \otimes \Delta$, and the remaining two arrows are the map Δ. Often the element $\Delta(c) \in C \otimes_k C$ is written $\sum_{(c)} c_{(1)} \otimes c_{(2)}$.

A *bialgebra* is a vector space H over k which has both an algebra and a coalgebra structure, such that the coalgebra structure maps are algebra homomorphisms, or equivalently, the algebra structure maps are coalgebra homomorphisms. (This equivalence can be seen by expressing the assertion that the coalgebra structure maps are algebra homomorphisms as a set of commutative diagrams: this set of diagrams is self-dual.)

Some examples of bialgebras are the following:

1. Let G be a group, and let kG be the group algebra of G: the vector space kG has the elements of G as a basis, with multiplication defined by extending the multiplication on G linearly. The coproduct and counit of kG are defined by

$$\left.\begin{array}{rcl} \Delta(g) & = & g \otimes g \\ \epsilon(g) & = & 1 \end{array}\right\} \qquad g \in G.$$

2. Let G be an affine algebraic group, and let $k[G]$ be the algebra of representative functions on G. The algebra structure of $k[G]$ is the usual algebra structure of functions with point-wise multiplications. The coproduct arises from the group multiplication $G \times G \to G$, which induces the map $k[G] \to k[G \times G] \cong k[G] \otimes_k k[G]$. The counit arises from the map $\{e\} \to G$, where $\{e\}$ is the single-element group.

3. Let L be a Lie algebra over k, and let $U(L)$ be the universal enveloping algebra of L. The coproduct and counit of $U(L)$ are defined by

$$\left.\begin{array}{rcl} \Delta(x) & = & 1 \otimes x + x \otimes 1 \\ \epsilon(x) & = & 0 \end{array}\right\} \qquad x \in L,$$

and extended to all of $U(L)$ using the fact that Δ and ϵ are algebra homomorphisms.

Usually, in studying bialgebras, an additional condition is imposed which is analogous to the assertion that a semigroup is a group. Such bialgebras are called *Hopf algebras.* The bialgebras which we consider in this paper (such as the universal enveloping algebra of a Lie algebra) satisfy this condition automatically.

A coalgebra is said to be *cocommutative* if it satisfies $\Delta = T \circ \Delta$, where T is the map $T : C \otimes_k C \to C \otimes_k C$ defined by $T(x \otimes y) = y \otimes x$. Note that the bialgebras in Examples 1 and 3 are cocommutative.

A vector space V over k is said to be *graded* if there is a sequence of subspaces V_0, V_1, $\ldots$ such that

$$V \cong \bigoplus_{n=0}^{\infty} V_n.$$

A graded vector space V is said to be *connected* if $V_0 \cong k$.

Let H be a bialgebra. A *H-module algebra* is an algebra R which is an H-module such that the action satisfies

$$h \cdot (fg) = \sum_{(h)} (h_{(1)} \cdot f)(h_{(2)} \cdot g), \qquad \text{for all } h \in H, \quad f, g \in R.$$

An *augmentation* of an algebra R over k is an algebra homomorphism $\epsilon : R \to k$. If $R = k[[x_1, \ldots, x_n]]$ is a power series algebra, the map $f \mapsto f(0)$ is an augmentation. If H is a bialgebra, then the dual space H^* is an algebra with multiplication defined by $pq(h) = (p \otimes q)\Delta(h)$ for $h \in H$ and p, $q \in H^*$. The algebra H^* is a H-module algebra with respect to the action $\rightharpoonup$ of H on H^* defined by $(h \rightharpoonup p)(k) = p(kh)$ for h, $k \in H$ and $p \in H^*$.

We close this section with a remark and two fundamental definitions:

- If $g \in H$ satisfies $\Delta(g) = g \otimes g$ and R is an H-module algebra, then g acts as an endomorphism of R; if $x \in H$ satisfies $\Delta(x) = 1 \otimes x + x \otimes 1$ and R is an H-module algebra, then x acts as a derivation of R.

- If H is a bialgebra, we say that $p \in H^*$ is *differentially produced by the algebra R with the augmentation ϵ* if there is a H-module algebra structure on R and there exists $f \in R$ satisfying

$$p(h) = \epsilon(h \cdot f).$$

- If $H = U(L)$ is a bialgebra which is the universal enveloping algebra of a Lie algebra L, we say that $p \in H^*$ has *finite Lie rank* if $L \rightharpoonup p$ is finite dimensional.

4 The algebra of Cayley trees

In this section, we follow [13] and define a bialgebra structure on spaces of trees. The relation between trees and differential operators goes back at least as far as Cayley [3] and [4]. Of this literature, the work most closely related to the view point taken here is Butcher's use of trees to analyze Runge-Kutta algorithms [1] and [2].

Let k will denote a field of characteristic 0 such as the real numbers or complex numbers. By a *tree* we will mean a finite rooted tree. Let $\mathcal{T}$ be the set of finite rooted trees, and let $k\{\mathcal{T}\}$ be the k-vector space which has $\mathcal{T}$ as a basis.

We first define the coalgebra structure on $k\{\mathcal{T}\}$. If $t \in \mathcal{T}$ is a tree whose root has children $s_1, \ldots, s_r$, the coproduct $\Delta(t)$ is the sum of the 2^r terms $t_1 \otimes t_2$, where the children of the root of t_1 and the children of the root of t_2 range over all 2^r possible partitions of the children of the root of t into two subsets. The augmentation ϵ which sends the trivial tree to 1 and every other tree to 0 is a counit for this coproduct. It is immediate that comultiplication is cocommutative.

We next define the algebra structure on $k\{\mathcal{T}\}$. Suppose that $t_1, t_2 \in \mathcal{T}$ are trees. Let $s_1, \ldots, s_r$ be the children of the root of t_1. If t_2 has $n+1$ nodes (counting the root), there are $(n+1)^r$ ways to attach the r subtrees of t_1 which have $s_1, \ldots, s_r$ as roots to the tree t_2 by making each s_i the child of some node of t_2. The product $t_1 t_2$ is defined to be the sum of these $(n+1)^r$ trees. It can be shown that this product is associative, and that the trivial tree consisting only of the root is a right and left unit for this product. It can also be shown that the maps defining the coalgebra structure are algebra homomorphisms, so that $k\{\mathcal{T}\}$ is a bialgebra. For details, see [10].

The bialgebra $k\{\mathcal{T}\}$ is graded: $k\{\mathcal{T}\}_n$ has as basis all trees with $n+1$ nodes. Because the bialgebra $k\{\mathcal{T}\}$ is graded connected, it is a Hopf algebra. We summarize the above discussion in the following theorem.

Theorem 4.1 *The vector space $k\{\mathcal{T}\}$ with basis the set of finite rooted trees is a cocommutative graded connected Hopf algebra.*

Assume now that each node of the tree (except for the root) is labeled with a symbol from the set $\{E_1, \ldots, E_M\}$. We can define the product and coproduct as above, and, once again, the resulting space is a bialgebra. See [10] for details. Let $k\{\mathcal{LT}\}$ denote this algebra.

Let R denote a subring of the commutative ring of smooth functions on

$\mathbf{R}^N$. We now define an action of the algebra of Cayley trees

$$B = k\{\mathcal{LT}\}$$

on the ring R, making R a B-module algebra, which captures the action of trees as higher derivations. This requires that we interpret the formal symbols E_j as derivations of R using Equations 2. The action is defined using the map

$$\psi : k\{\mathcal{LT}\} \to \operatorname{End}_k R,$$

as follows:

1. Given a labeled, ordered tree t with $m+1$ nodes, assign the root the number 0 and assign the remaining nodes the numbers $1, \ldots, m$. We identify the node with the number assigned to it. To the node k associate the summation index μ_k. Denote $(\mu_1, \ldots, \mu_m)$ by μ.

2. For the labeled tree t, let k be a node of t, labeled with E_{γ_k} if $k > 0$, and let $l, \ldots, l'$ be the children of k. Define

$$\begin{aligned} R(k;\mu) &= D_{\mu_l} \cdots D_{\mu_{l'}} a^{\mu_k}_{\gamma_k}, \quad \text{if } k > 0 \text{ is not the root;} \\ &= D_{\mu_l} \cdots D_{\mu_{l'}}, \quad \text{if } k = 0 \text{ is the root.} \end{aligned}$$

   Note that if $k > 0$, then $R(k;\mu) \in R$.

3. Define
$$\psi(t) = \sum_{\mu_1,\ldots,\mu_m = 1}^{N} R(m;\mu) \cdots R(1;\mu) c(0;\mu).$$

4. Extend ψ to all of $k\{\mathcal{LT}\}$ by linearity.

It is straightforward to check that this action of B on R makes R into a B-module algebra.

We summarize with the following theorem.

Theorem 4.2 *Let R the algebra of smooth functions on $\mathbf{R}^N$. Let B denote the algebra of Cayley trees $k\{\mathcal{LT}\}$. Then R is a B-module algebra with respect to the action defined by ψ.*

5 Symbolic evaluation of vector field expressions

When expressions involving vector fields, such as Lie brackets, are written out in coordinates, there is typically a lot of cancellation, as we have seen above. Similar cancellation occurs in expressions involving Poisson brackets and when flows are concatenated, as in Campbell-Baker-Hausdorf expansions. In this section, we use the algebra of Cayley trees to exploit this cancellation in order to compute more efficiently formal expressions involving vector fields.

The standard action of the algebra A of differential operators generated by E_1, …, E_M on the algebra of smooth functions R gives R the structure of a A-module algebra. It is easy to relate these two H-module algebra structures on R and this observation is the basis for our algorithms.

Let

$$\phi : A \longrightarrow B$$

denote the map sending the generator E_j of the algebra A to the tree consisting of two nodes: the root and a single child labeled E_j. This map is illustrated in Figure 1. Extend ϕ to be an algebra homomorphism. Let χ denote the map

$$A \rightarrow \mathrm{End}_k\, R$$

defined by using the substitution (2) and simplifying to obtain an endomorphim of R. We have the following diagram:

$$\begin{tikzcd} A \arrow[r] \arrow[dr] & B \arrow[d] \\ & \mathrm{End}_k\, R \end{tikzcd} \qquad (5)$$

Theorem 5.1 *(i) The maps χ, ϕ and ψ are related by $\chi = \psi \circ \phi$. (ii) Fix a function $f \in R$ and a differential operator $p \in A$. Then*

$$p \cdot f = \phi(p) \cdot f.$$

Here the action on the left views R as an A-module algebra, while the action on the right views R as B-module algebra.

The first assertion is proved in [14] and the second assertion follows from the first assertion and the definitions. From this theorem, we get:

No. of terms	Form of terms
$8N^3$	coeff. D_{μ_1}
$12N^3$	coeff. $D_{\mu_2}D_{\mu_1}$
$4N^3$	coeff. $D_{\mu_3}D_{\mu_2}D_{\mu_1}$

Table 1: Naive computation of the differential operator corresponding to p.

Algorithm 5.1 *To rewrite expressions in the first order differential operators E_j in terms of the basis*

$$\frac{\partial}{\partial x_{\mu_1}}, \quad \frac{\partial^2}{\partial x_{\mu_1}\partial x_{\mu_2}}, \ldots, \qquad \mu_1, \mu_2, \ldots = 1, \ldots, N,$$

compute the composition of the rightward and downward pointing arrows in the diagram above.

In [11], [12] and [14], we show that the algorithm is much more efficient than naive substitution, which corresponds to computing the diagonal arrow directly. In some common cases, the improvement in efficiency is exponential. We have implemented this and related algorithms in Maple, Mathematica and Snobol.

We illustrate this algorithm by working the example of the last section following [7]: consider a higher order derivation of the form

$$p = E_3E_2E_1 - E_3E_1E_2 - E_2E_1E_3 + E_1E_2E_3.$$

Naive simplification requires computing $24N^3$ terms of the form described in Table 1. The image of p in the algebra of Cayley trees contains 24 trees, six for each of the four terms of p. For example, the six labeled trees corresponding to the first term are given in Figure 2. Eighteen of these trees cancel, leaving the six trees in Figure 4. The corresponding differential operator is equal to

$$\begin{aligned}
&\sum a_3^{\mu_3}(D_{\mu_3}a_2^{\mu_2})(D_{\mu_2}a_1^{\mu_1})D_{\mu_1} - \sum a_3^{\mu_3}(D_{\mu_3}a_1^{\mu_2})(D_{\mu_2}a_2^{\mu_1})D_{\mu_1} \\
&- \sum a_2^{\mu_3}(D_{\mu_3}a_1^{\mu_2})(D_{\mu_2}a_3^{\mu_1})D_{\mu_1} + \sum a_1^{\mu_3}(D_{\mu_3}a_2^{\mu_2})(D_{\mu_2}a_3^{\mu_1})D_{\mu_1} \\
&+ \sum a_3^{\mu_3}a_2^{\mu_2}(D_{\mu_3}D_{\mu_2}a_1^{\mu_1})D_{\mu_1} - \sum a_3^{\mu_3}a_1^{\mu_2}(D_{\mu_3}D_{\mu_2}a_2^{\mu_1})D_{\mu_1},
\end{aligned}$$

and contains $18N^3$ fewer terms than does the naive computation of p. Notice that all the terms of degree 2 and 3 cancel, as well as some of the first order terms. An example of the cancellation of labeled trees is given in Figure 5.

Figure 4: The surviving labeled trees.

Figure 5: The term $E_3E_1E_2$ contributes the first labeled tree and the term $E_1E_2E_3$ contributes the second, which cancel.

6 The algebraic realization of input-output maps

In this section we state a theorem which uses bialgebras to describe the realizability of input-output maps of control systems. If L is a Lie algebra of derivations of R which is generated by $\{E_1, \ldots, E_M\}$, then $H = U(L)$ has a basis of monomials in the E_i. If $x(t)$ is a solution to the control system

$$\dot{x}(t) = \sum_{i=1}^{M} u_i(t) E_i(x(t)), \tag{6}$$

and $f \in R$ is an observation function, then the formal generating series

$$p = \sum p_\mu \mu, \quad \text{where } p_\mu = E_{\mu_1} \cdots E_{\mu_k} f(x(0)),$$

can be thought of as a functional on the bialgebra $H = U(L)$. Using the terminology introduced above, p is differentially produced by R. The following theorem gives a condition which is stated in terms of the action of H on H^* for $p \in H^*$ to be the generating series associated with an input-output map. (See [15] for a complete exposition.)

Theorem 6.1 *Let L be a Lie algebra, and let $H = U(L)$. Then p is of finite Lie rank if and only if p is differentially produced by an augmented algebra R with* $\ker\epsilon/(\ker\epsilon)^2$ *finite dimensional.*

It can be shown that R can be taken to be isomorphic to a power series ring in finitely many variables.

References

[1] J. C. Butcher, "An order bound for Runge-Kutta methods," *SIAM J. Numerical Analysis,* Vol. 12, pp. 304–315, 1975.

[2] J. C. Butcher, *The Numerical Analysis of Ordinary Differential Equations,* John Wiley, 1986.

[3] A. Cayley, "On the theory of analytical forms called trees", *Collected Mathematical Papers of Arthur Cayley,* Cambridge University Press, Vol. 3, pp. 242–6, 1890.

[4] A. Cayley, "On the analytical forms called trees, second part", in *Collected Mathematical Papers of Arthur Cayley,* Cambridge University Press, Vol. 4, pp. 112–5, 1891.

[5] P. Crouch, R. Grossman, and R. G. Larson, “Computations involving differential operators and their actions on functions,” *Laboratory for Advanced Computing Technical Report,* Number LAC91-R2, University of Illinois at Chicago, January, 1991.

[6] P. Crouch, R. Grossman, and R. Larson, “Trees, bialgebras, and intrinsic numerical integrators,” *Laboratory for Advanced Computing Technical Report,* Number LAC90-R23, University of Illinois at Chicago, May, 1990.

[7] R. Grossman, “The evaluation of expresssions involving higher order derivations,” *Journal of Mathematical Systems, Estimation, and Control,* Vol. 1 (1991), pp. 91–106.

[8] R. Grossman, “Using trees to compute approximate solutions of ordinary differential equations exactly,” *Computer Algebra and Differential Equations,* M. F. Singer, editor, Academic Press, New York, 1991, in press.

[9] R. Grossman, *Analytic Computation: An Introduction,* Laboratory for Advanced Computing Technical Report, Number LAC90-R17, University of Illinois at Chicago, April, 1990.

[10] R. Grossman and R. G. Larson, “Hopf algebraic structures of families of trees,” *J. Algebra,* Vol. 26, pp. 184–210, 1989.

[11] R. Grossman and R. G. Larson, “Labeled trees and the algebra of differential operators,” *Algorithms and Graphs,* B. Richter, editor, American Mathematical Society, Providence, pp. 81–87, 1989.

[12] R. Grossman and R. G. Larson, “Labeled trees and the efficient computation of derivations,” in *Proceedings of 1989 International Symposium on Symbolic and Algebraic Computation,* ACM, pp. 74–80, 1989.

[13] R. Grossman and R. G. Larson, “Hopf-algebraic structure of combinatorial objects and differential operators,” *Israeli J. Math.,* to appear.

[14] R. Grossman and R. G. Larson, “The symbolic computation of derivations using labeled trees,” *J. Symbolic Comptutation,* to appear.

[15] R. Grossman and R. G. Larson, “The realization of input-output maps using bialgebras,” to appear.