Efficient computation of the Zassenhaus formula

Fernando Casas[1*]   Ander Murua[2†]   Mladen Nadinic[3]

June 8, 2012

Abstract

A new recursive procedure to compute the Zassenhaus formula up to high order is presented, providing each exponent in the factorization directly as a linear combination of independent commutators and thus containing the minimum number of terms. The recursion can be easily implemented in a symbolic algebra package and requires much less computational effort, both in time and memory resources, than previous algorithms. In addition, by bounding appropriately each term in the recursion, it is possible to get a larger convergence domain of the Zassenhaus formula when it is formulated in a Banach algebra.

[1]Institut de Matemàtiques i Aplicacions de Castelló and Departament de Matemàtiques, Universitat Jaume I, E-12071 Castellón, Spain.

[2]Konputazio Zientziak eta A.A. saila, Informatika Fakultatea, EHU/UPV, Donostia/San Sebastián, Spain.

[3] Departamento de Matemática, Facultad de Ciencias, Universidad de Santiago de Chile, Avda. Bernardo O'Higgins 3363, Estación Central, Santiago, Chile.

1   Introduction

Products of exponentials of non-commuting indeterminate variables are of fundamental importance in physics and mathematics. As is well known, the Baker–Campbell–Hausdorff theorem states that $\mathrm{e}^X \mathrm{e}^Y = \mathrm{e}^Z$, with

$$Z = \log(\mathrm{e}^X \mathrm{e}^Y) = X + Y + \sum_{m=2}^{\infty} Z_m(X,Y). \tag{1.1}$$

Here $Z_m(X,Y)$ is a homogeneous Lie polynomial in the non-commuting variables X and Y. In other words, Z_m is a linear combination (with rational coefficients) of commutators of the form $[V_1, [V_2, \ldots, [V_{m-1}, V_m]\ldots]]$ with $V_i \in \{X,Y\}$

*Email: Fernando.Casas@uji.es

†Email: Ander.Murua@ehu.es

for $1 \le i \le m$. We recall that $[X, Y] \equiv XY - YX$. The first terms in the series (1.1) read explicitly

$$Z_2 = \frac{1}{2}[X, Y], \quad Z_3 = \frac{1}{12}[X, [X, Y]] - \frac{1}{12}[Y, [X, Y]], \quad Z_4 = \frac{1}{24}[X, [Y, [Y, X]]].$$

The expression $\mathrm{e}^X \, \mathrm{e}^Y = \mathrm{e}^Z$ is then properly called the *Baker–Campbell–Hausdorff formula* (BCH for short) and plays a fundamental role in many fields of mathematics (theory of linear differential equations [1], Lie groups [2], numerical analysis [3]), theoretical physics (perturbation theory [4], Quantum Mechanics [5], Statistical Mechanics [6, 7], quantum computing [8]), control theory (analysis and design of nonlinear control laws, nonlinear filters, stabilization of rigid bodies [9]), etc. (see [10] for a comprehensive treatment of the algebraic aspects of the BCH formula).

Although the BCH theorem establishes the precise algebraic structure of the exponent Z in (1.1), it does not provide simple ways to compute explicitly this series. As a matter of fact, the problem of effectively computing the BCH series up to arbitrary degree has a long history, and different procedures have been proposed along the years, starting with the work of Richtmyer and Greenspan in 1965 (see [11] for a review). Most of the procedures lead to expressions where not all the iterated commutators are linearly independent (due to the Jacobi identity and other identities appearing at higher degrees). Equivalently, the resulting expressions are not formulated directly in terms of a basis of the free Lie algebra $\mathcal{L}(X, Y)$ generated by the symbols X and Y. Of course, it is always possible to write these expressions in terms of a basis, but this rewriting process is time consuming and require considerable memory resources. This in addition is made more difficult due to the exponential growth of the number of terms with the degree m. Recently, a new efficient algorithm has been proposed which allows one to get closed expressions for Z_m up to a very high degree in terms of both the classical Hall basis and the Lyndon basis of $\mathcal{L}(X, Y)$ [11].

In the paper dealing with the expansion bearing his name, Magnus [1] cites an unpublished reference by Zassenhaus, reporting that there exists a formula which may be called the dual of the (Baker–Campbell–)Hausdorff formula. This result can be stated as follows.

Theorem 1.1 *(Zassenhaus Formula). Let $\mathcal{L}(X, Y)$ be the free Lie algebra generated by X and Y. Then, e^{X+Y} can be uniquely decomposed as*

$$\mathrm{e}^{X+Y} = \mathrm{e}^X \, \mathrm{e}^Y \prod_{n=2}^{\infty} \mathrm{e}^{C_n(X,Y)} = \mathrm{e}^X \, \mathrm{e}^Y \, \mathrm{e}^{C_2(X,Y)} \, \mathrm{e}^{C_3(X,Y)} \cdots \mathrm{e}^{C_n(X,Y)} \cdots, \quad (1.2)$$

where $C_n(X, Y) \in \mathcal{L}(X, Y)$ is a homogeneous Lie polynomial in X and Y of degree n.

The existence of such a formula is an immediate consequence of the BCH theorem. In fact, it is clear that $\mathrm{e}^{-X} \mathrm{e}^{X+Y} = \mathrm{e}^{Y+D}$, where D involves Lie polynomials of degree > 1. Now $\mathrm{e}^{-Y} \mathrm{e}^{Y+D} = \mathrm{e}^{C_2+\tilde{D}}$, where $\tilde{D}$ involves Lie polynomials of degree > 2 and the process is repeated again. Induction allows one to get the general result.

By comparing with the BCH formula it is possible to obtain the first terms of the formula (1.2) as

$$C_2(X,Y) = -\frac{1}{2}[X,Y], \qquad C_3(X,Y) = \frac{1}{3}[Y,[X,Y]] + \frac{1}{6}[X,[X,Y]].$$

Although less familiar than the BCH formula, the Zassenhaus formula constitutes nevertheless a standard tool in several fields, since the disentangling of the exponential of the sum of two non-commuting operators into an (in general infinite) product of exponential operators arises for instance in statistical mechanics, many-body theories, quantum optics, path integrals, q-analysis in quantum groups, etc. [12]. Also in particle accelerators physics, Dragt and his collaborators have used the Zassenhaus formula to compute the relevant maps both in Taylor and factored product form [13]. Yet in another context, very recently Iserles and Kropielnicka [14] have proposed a new family of high-order splitting methods for the numerical integration of the time-dependent Schrödinger equation based on a symmetric version of the Zassenhaus formula.

Several systematic computations of the terms C_n for $n > 3$ in the Zassenhaus formula have been carried out in the literature, starting with the work of Wilcox [7], where a recursive procedure is presented that has been subsequently used to get explicit expressions up to C_6 in terms of nested commutators [12]. On the other hand, Volkin [15] proposed a general technique to express a function of non-commuting operators as an expansion in successively higher commutators of the operators involved. In particular, he was able to get recursive formulae up to C_9. By following an idea already suggested by Wilcox in [7], Suzuki [16] obtained the successive terms $C_n(X,Y)$ in

$$e^{\lambda(X+Y)} = e^{\lambda X} e^{\lambda Y} e^{\lambda^2 C_2} e^{\lambda^3 C_3} \cdots \tag{1.3}$$

by differentiating both sides with respect to λ and setting $\lambda = 0$ after each differentiation. In this way

$$\begin{aligned}
C_2 &= \frac{1}{2}\left(\frac{d^2}{d\lambda^2}(e^{-\lambda Y}e^{-\lambda X}e^{\lambda(X+Y)})\right)_{\lambda=0} = \frac{1}{2}[Y,X] \\
C_3 &= \frac{1}{3!}\left(\frac{d^3}{d\lambda^3}(e^{-\lambda^2 C_2}e^{-\lambda Y}e^{-\lambda X}e^{\lambda(X+Y)})\right)_{\lambda=0} = \frac{1}{3}[C_2, X + 2Y]
\end{aligned}$$

and in general, for $n \geq 3$,

$$C_n = \frac{1}{n!}\left(\frac{d^n}{d\lambda^n}(e^{-\lambda^{n-1}C_{n-1}}\cdots e^{-\lambda^2 C_2}e^{-\lambda Y}e^{-\lambda X}e^{\lambda(X+Y)})\right)_{\lambda=0}. \tag{1.4}$$

Finally, Baues [17] gave explicit formulae for the Zassenhaus terms via homotopy theory and more recently Kurlin [18] obtained a closed expression for $\sum_{n\geq 2} C_n$ in the metabelian case.

All of these proposals give results for C_n as a linear combination of nested commutators. In contrast, Scholz and Weyrauch [19] proposed a recursive procedure based on upper triangular matrices that can be easily implemented in a symbolic algebra package. In this case, however, the expressions for C_n are

not explicitly written down in terms of homogeneous Lie polynomials. More recently [20], the same authors have applied a technique related to one previously used by Witschel [21] to get C_n up to $n = 15$ in less than 2 minutes of CPU time. Here also the Zassenhaus terms are written as

$$C_n = \sum_{w,|w|=n} g_w\, w, \qquad (1.5)$$

where g_w is a rational coefficient and the sum is taken over all words w with length $|w| = n$ in the symbols X and Y, i.e., $w = a_1 a_2 \cdots a_n$, each a_i being X or Y. Of course, one may always apply the Dynkin–Specht–Wever theorem [22], and express C_n as

$$C_n = \frac{1}{n} \sum_{w,|w|=n} g_w\, [w], \qquad (1.6)$$

that is, the individual terms are the same as in the associative series (1.5) except that the word $w = a_1 a_2 \ldots a_n$ is replaced with the right nested commutator $[w] = [a_1, [a_2, \ldots [a_{n-1}, a_n] \ldots]]$ and the coefficient g_w is divided by the word length n. In this way C_n is constructed as a linear combination of nested commutators of homogeneous degree n, that is, as a linear combination of elements of the homogeneous subspace $\mathcal{L}(X,Y)_n$ of degree n of the free Lie algebra $\mathcal{L}(X,Y)$. As a matter of fact, another representation of (1.5) in terms of nested commutators is proposed in [20] which, it is claimed, contains less terms than the Dynkin–Specht–Wever representation. In any case, it should be stressed that the set of nested commutators $[w]$ for words w of length n in either representation is *not* a basis of the homogeneous subspace $\mathcal{L}(X,Y)_n$.

The purpose of this paper is twofold. First, to present a new recurrence that allows one to express the Zassenhaus terms C_n directly as a linear combination of independent elements of the homogeneous subspace $\mathcal{L}(X,Y)_n$. In other words, the procedure, which can be easily implemented in a symbolic algebra package, gives C_n up to a prescribed degree directly in terms of independent commutators involving n operators X and Y. In this way, no rewriting process in a basis of $\mathcal{L}(X,Y)$ is necessary, thus saving considerable computing time and memory resources. Moreover, we are able to express directly C_n with the minimum number of commutators required at each degree n.

The second aspect we are dealing with concerns the convergence of the Zassenhaus formula when it is formulated in a Banach algebra. As far as we know, there are only two previous results in the literature. The first one was obtained by Suzuki [16] starting with the recurrence (1.4). Specifically, he was able to prove that if $|\lambda|(\|X\| + \|Y\|) \le \log 2 - 1/2$ the infinite product (1.3) converges. Subsequently, Bayen [23] generalized the analysis, showing that the product (1.3) converges if $|\lambda|(\|X\| + \|Y\|) \le r$, where $r \approx 0.596705$ is a root of a certain transcendental equation. In the present work, we obtain sharper bounds for the terms of the Zassenhaus formula which show that the product (1.2) converges in an enlarged domain.

A simple but important remark is in order here. In some applications, the "left-oriented" Zassenhaus formula

$$e^{X+Y} = \cdots\, e^{\hat{C}_4(X,Y)}\, e^{\hat{C}_3(X,Y)}\, e^{\hat{C}_2(X,Y)}\, e^{Y}\, e^{X} \qquad (1.7)$$

4

is used instead of (1.2). A simple observation shows that the exponents $\hat{C}_i$ and C_i are related through

$$\hat{C}_i(X, Y) = (-1)^{i+1} C_i(X, Y), \qquad i \geq 2$$

and so we may restrict ourselves to analyzing the "right-oriented" formula (1.2).

2 Constructing the Zassenhaus terms

2.1 A new recurrence

To derive our recursive procedure, it is convenient to introduce a parameter λ as in (1.3),

$$e^{\lambda(X+Y)} = e^{\lambda X} e^{\lambda Y} e^{\lambda^2 C_2} e^{\lambda^3 C_3} e^{\lambda^4 C_4} \cdots \tag{2.1}$$

so that the original Zassenhaus formula (1.2) is recovered when $\lambda = 1$. Moreover, we consider the compositions

$$R_1(\lambda) = e^{-\lambda Y} e^{-\lambda X} e^{\lambda(X+Y)} \tag{2.2}$$

and for each $n \geq 2$,

$$R_n(\lambda) = e^{-\lambda^n C_n} \cdots e^{-\lambda^2 C_2} e^{-\lambda Y} e^{-\lambda X} e^{\lambda(X+Y)} = e^{-\lambda^n C_n} R_{n-1}(\lambda). \tag{2.3}$$

It is then clear that

$$R_n(\lambda) = e^{\lambda^{n+1} C_{n+1}} e^{\lambda^{n+2} C_{n+2}} \cdots \tag{2.4}$$

Finally, we introduce

$$F_n(\lambda) \equiv \left(\frac{d}{d\lambda} R_n(\lambda) \right) R_n(\lambda)^{-1}, \qquad n \geq 1. \tag{2.5}$$

To determine the series $F_n(\lambda)$ we proceed as follows. On the one hand, a simple calculation starting from (2.3), leads for $n \geq 2$ to

$$
\begin{aligned}
F_n(\lambda) &= -n\, C_n\, \lambda^{n-1} + e^{-\lambda^n C_n} \left(\frac{d}{d\lambda} R_{n-1}(\lambda) \right) R_{n-1}(\lambda)^{-1} e^{\lambda^n C_n} \\
&= -n\, C_n\, \lambda^{n-1} + e^{-\lambda^n C_n}\, F_{n-1}(\lambda)\, e^{\lambda^n C_n} \\
&= -n\, C_n\, \lambda^{n-1} + e^{-\lambda^n \operatorname{ad}_{C_n}} F_{n-1}(\lambda) \\
&= e^{-\lambda^n \operatorname{ad}_{C_n}} \left(F_{n-1}(\lambda) - n\, C_n\, \lambda^{n-1} \right),
\end{aligned}
\tag{2.6}
$$

where we have used the well known formula

$$e^A B e^{-A} = e^{\operatorname{ad}_A} B = \sum_{n \geq 0} \frac{1}{n!} \operatorname{ad}_A^n B$$

with

$$\operatorname{ad}_A B = [A, B], \qquad \operatorname{ad}_A^j B = [A, \operatorname{ad}_A^{j-1} B], \qquad \operatorname{ad}_A^0 B = B.$$

On the other hand, differentiating expression (2.4) with respect to λ and taking into account (2.5) we arrive at

$$F_n(\lambda) = (n+1)\,C_{n+1}\,\lambda^n + \sum_{j=n+2}^{\infty} j\,\lambda^{j-1}\,\mathrm{e}^{\lambda^{n+1}\mathrm{ad}_{C_{n+1}}} \cdots \mathrm{e}^{\lambda^{j-1}\mathrm{ad}_{C_{j-1}}}C_j. \quad (2.7)$$

In other words,

$$F_n(\lambda) = (n+1)C_{n+1}\,\lambda^n + G_{n+1}(\lambda), \qquad n \geq 1,$$

where $G_{n+1}(0) = G_{n+1}^{(1)}(0) = \cdots = G_{n+1}^{(n)}(0) = 0$. In consequence, we have, for $n \geq 1$,

$$\begin{aligned}
F_{n+1}(\lambda) &= \mathrm{e}^{-\lambda^{n+1}\mathrm{ad}_{C_{n+1}}}\,G_{n+1}(\lambda), &(2.8)\\
C_{n+1} &= \frac{1}{(n+1)!}\,F_n^{(n)}(0), &(2.9)\\
G_{n+1}(\lambda) &= F_n(\lambda) - \frac{\lambda^n}{n!}F_n^{(n)}(0). &(2.10)
\end{aligned}$$

Expressions (2.8)–(2.10) allow one to compute recursively the Zassenhaus terms C_n starting from $F_1(\lambda)$. The sequence is

$$F_n(\lambda) \;\longrightarrow\; C_{n+1} \;\longrightarrow\; G_{n+1}(\lambda) \;\longrightarrow\; F_{n+1}(\lambda) \;\longrightarrow\; \cdots, \quad n \geq 1.$$

Let us analyze more in detail this procedure, with the goal of providing an algorithm well adapted from a computational point of view.

For $n = 1$, and taking into account (2.2), we get

$$\begin{aligned}
F_1(\lambda) &= -Y - \mathrm{e}^{-\lambda Y}X\mathrm{e}^{\lambda Y} + \mathrm{e}^{-\lambda Y}\mathrm{e}^{-\lambda X}(X+Y)\mathrm{e}^{\lambda X}\mathrm{e}^{\lambda Y}\\
&= -Y - \mathrm{e}^{-\lambda\mathrm{ad}_Y}X + \mathrm{e}^{-\lambda\mathrm{ad}_Y}\mathrm{e}^{-\lambda\mathrm{ad}_X}(X+Y)\\
&= \mathrm{e}^{-\lambda\mathrm{ad}_Y}(\mathrm{e}^{-\lambda\mathrm{ad}_X} - I)Y,
\end{aligned}$$

that is,

$$F_1(\lambda) = \sum_{i=0}^{\infty}\sum_{j=1}^{\infty} \frac{(-\lambda)^{i+j}}{i!j!}\mathrm{ad}_Y^i\,\mathrm{ad}_X^j Y \quad (2.11)$$

or equivalently

$$F_1(\lambda) = \sum_{k=1}^{\infty} f_{1,k}\,\lambda^k, \qquad \text{with} \qquad f_{1,k} = \sum_{j=1}^{k} \frac{(-1)^k}{j!(k-j)!}\mathrm{ad}_Y^{k-j}\,\mathrm{ad}_X^j Y. \quad (2.12)$$

In general, from (2.6) a straightforward calculation shows that for $n \geq 2$,

$$F_n(\lambda) = \sum_{k=n}^{\infty} f_{n,k}\,\lambda^k, \qquad \text{with} \qquad f_{n,k} = \sum_{j=0}^{[k/n]-1} \frac{(-1)^j}{j!}\mathrm{ad}_{C_n}^j\,f_{n-1,k-nj}, \qquad k \geq n.$$

$$(2.13)$$

6

Here $[k/n]$ denotes the integer part of k/n. Moreover, a closer examination of (2.7) reveals that

$$
\begin{aligned}
F_n(\lambda) &= (n+1)C_{n+1}\lambda^n + (n+2)e^{\lambda^{n+1}\mathrm{ad}_{C_{n+1}}}C_{n+2}\lambda^{n+1} + \cdots \\
&= (n+1)C_{n+1}\lambda^n + (n+2)C_{n+2}\lambda^{n+1} + \cdots \\
&\quad + (2n+2)C_{2n+2}\lambda^{2n+1} + \lambda^{2n+2}[C_{n+1}, C_{n+2}] + \cdots \\
&= \sum_{k=n+1}^{2n+2} k\, C_k\, \lambda^{k-1} + \lambda^{2n+2}H_n(\lambda),
\end{aligned} \tag{2.14}
$$

where $H_n(\lambda)$ involves commutators of C_j, $j \ge n+1$. Notice that the terms $C_{n+1}, \ldots, C_{2n+2}$ of the Zassenhaus formula can be then directly obtained from $F_n(\lambda)$. In particular, one directly gets from (2.11)

$$
C_{n+1} = \frac{1}{n+1}f_{1,n} = \frac{1}{n+1}\sum_{i=0}^{n-1}\frac{(-1)^n}{i!(n-j)!}\mathrm{ad}_Y^i\,\mathrm{ad}_X^{n-j}Y, \quad \text{for} \quad n=1,2,3. \tag{2.15}
$$

Explicitly,

$$
\begin{aligned}
C_2 &= -\frac{1}{2}[X,Y], \\
C_3 &= \frac{1}{3}[Y,[X,Y]] + \frac{1}{6}[X,[X,Y]], \\
C_4 &= -\frac{1}{8}([Y,[Y,[X,Y]]] + [Y,[X,[X,Y]]]) - \frac{1}{24}[X,[X,[X,Y]]].
\end{aligned}
$$

Taking into account (2.13) and (2.14) we have in general

$$
C_{n+1} = \frac{1}{n+1}\,f_{[n/2],n} \qquad n \ge 5, \tag{2.16}
$$

where the expressions of $f_{n,k}$ are given recursively by (2.13).

In summary, the algorithm we propose for computing the Zassenhaus terms is the following:

$$
\begin{array}{l}
\text{Define } f_{1,k} \text{ by eq. (2.12)} \\
C_n = (1/n)\,f_{1,n-1}, \quad n=2,3,4 \\
\text{Define } f_{n,k} \quad n \ge 2,\ k \ge n \text{ by eq. (2.13)} \\
C_n = (1/n)f_{[(n-1)/2],n-1} \quad n \ge 5.
\end{array} \tag{2.17}
$$

This constitutes a new recursive way for obtaining directly the term C_n as a homogeneous Lie polynomial in X, Y of arbitrarily large degree n which can be easily implemented with a symbolic algebra package.

2.2 Linear independence

Algorithm (2.17), or equivalently the procedure given by the identities (2.8)–(2.10), provides expressions for C_n that, by construction, involve only independent commutators. In other words, they cannot be simplified further by using the Jacobi identity and the antisymmetry property of the commutator.

In order to prove this assertion, it is convenient to get a more explicit expression of $F_n(\lambda)$ and C_{n+1} from (2.8)–(2.10). To this end, consider for $n \geq 1$ the sets $\mathcal{J}_n$ and $\mathcal{I}_n$ of $(n+1)$-tuples of non-negative integers recursively defined as follows:

$$
\begin{aligned}
\mathcal{J}_1 &= \{(i_0, i_1) \in \mathbb{N}^2 \ : \ i_0 \geq 1\}, \\
\mathcal{I}_n &= \{(i_0, i_1, \ldots, i_n) \in \mathcal{J}_n \ : \ i_0 + i_1 + 2i_2 + \cdots + ni_n = n\}, \\
\mathcal{J}_{n+1} &= (\mathcal{J}_n \backslash \mathcal{I}_n) \times \mathbb{N}.
\end{aligned}
$$

The set $\mathcal{I}_n$ can be directly defined as the set of $(n+1)$-tuples of non-negative integers satisfying that $i_0 + i_1 + 2\,i_2 + \cdots + n\,i_n = n$ and

$$ j + 1 \leq i_0 + i_1 + 2\,i_2 \cdots + j\,i_j \quad \text{for} \quad j = 0, \ldots, n-1. \tag{2.18} $$

Thus, in particular, $\mathcal{I}_1 = \{(1,0)\}$, $\mathcal{I}_2 = \{(1,1,0),(2,0,0)\}$, etc. Observe that, by construction, each $(i_0, i_1, \ldots, i_n) \in \mathcal{I}_n$ is such that $i_m = 0$ if $m > n/2$.

From (2.8)–(2.10), one can then prove by induction on n that, for $n \geq 1$,

$$
\begin{aligned}
F_n(\lambda) &= \sum_{(i_0, i_1, \ldots, i_n) \in \mathcal{J}_n} \frac{(-1)^{i_0 + \cdots + i_n} \lambda^{i_0 + i_1 + 2i_2 + \cdots + ni_n}}{i_0! i_1! \cdots i_n!} \mathrm{ad}_{C_n}^{i_n} \cdots \mathrm{ad}_{C_2}^{i_2} \mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y, \\
C_{n+1} &= \frac{1}{n+1} \sum_{(i_0, i_1, \ldots, i_n) \in \mathcal{I}_n} \frac{(-1)^{i_0 + \cdots + i_n}}{i_0! i_1! \cdots i_n!} \mathrm{ad}_{C_n}^{i_n} \cdots \mathrm{ad}_{C_2}^{i_2} \mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y. \tag{2.19}
\end{aligned}
$$

In fact, this is clearly true for $n = 1$ (equations (2.11) and (2.15), respectively), whereas successive application of (2.8)–(2.10) leads to the general result.

Now, repeated application of Lazard elimination principle [24], together with $\mathcal{I}_1 = \{(1,0)\}$, $\{C_2\} = \{-\frac{1}{2}[X,Y]\} = \{-\frac{1}{2}\mathrm{ad}_Y^{i_1}\mathrm{ad}_X^{i_0}Y \ : \ (i_0, i_1) \in \mathcal{I}_1\}$, shows that, as a vector space,

$$
\begin{aligned}
\mathcal{L}(X,Y) &= \mathrm{span}(\{X\}) \oplus \mathcal{L}(\{\mathrm{ad}_X^j Y \ : \ j \geq 0\}) \\
&= \mathrm{span}(\{X\}) \oplus \mathrm{span}(\{Y\}) \oplus \mathcal{L}(\{\mathrm{ad}_Y^i \mathrm{ad}_X^i Y \ : \ i \geq 0, j \geq 1\}) \\
&= \mathrm{span}(\{X,Y\}) \oplus \mathcal{L}(\{\mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y \ : \ (i_0, i_1) \in \mathcal{J}_1\}) \\
&= \mathrm{span}(\{X,Y\}) \oplus \mathcal{L}(\{C_2\} \cup \{\mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y \ : \ (i_0, i_1) \in \mathcal{J}_1 \backslash \mathcal{I}_1\}) \\
&= \mathrm{span}(\{X,Y,C_2\}) \oplus \mathcal{L}(\{\mathrm{ad}_{C_2}^{i_2} \mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y \ : \ (i_0, i_1, i_2) \in \mathcal{J}_2\}).
\end{aligned}
$$

More generally, application of Lazard elimination together with (2.19) gives

$$
\begin{aligned}
\mathcal{L}(X,Y) &\subset \mathrm{span}(\{X,Y,C_2,\ldots,C_n\}) \\
&\quad \oplus \mathcal{L}(\{\mathrm{ad}_{C_n}^{i_n} \cdots \mathrm{ad}_{C_2}^{i_2} \mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y \ : \ (i_0, \ldots, i_n) \in \mathcal{J}_n\}) \\
&\subset \mathrm{span}(\{X,Y,C_2,\ldots,C_n\}) \\
&\quad \oplus \mathcal{L}(\{C_{n+1}\} \cup \{\mathrm{ad}_{C_n}^{i_n} \cdots \mathrm{ad}_{C_2}^{i_2} \mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y \ : \ (i_0, \ldots, i_n) \in \mathcal{J}_n \backslash \mathcal{I}_n\}) \\
&\subset \mathrm{span}(\{X,Y,C_2,\ldots,C_{n+1}\}) \\
&\quad \oplus \mathcal{L}(\{\mathrm{ad}_{C_{n+1}}^{i_{n+1}} \cdots \mathrm{ad}_{C_2}^{i_2} \mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y \ : \ (i_0, \ldots, i_{n+1}) \in \mathcal{J}_{n+1}\}).
\end{aligned}
$$

In consequence, the terms $\{\mathrm{ad}_{C_m}^{i_m} \cdots \mathrm{ad}_{C_2}^{i_2} \mathrm{ad}_Y^{i_1} \mathrm{ad}_X^{i_0} Y \ : \ (i_0, i_1, \ldots, i_m) \in \mathcal{J}_n\}$ are linearly independent in the free Lie algebra $\mathcal{L}(X,Y)$ and the same is true for the representation (2.19) of the Zassenhaus terms.

2.3   Computational aspects

We have implemented the recursive procedure (2.17) in $Mathematica^{\text{TM}}$ as the following algorithm.

```
Clear[Cmt, ad, ff, cc];
$RecursionLimit= 1024;
Cmt[a_, a_]:= 0;
Cmt[a___, 0, b___]:= 0;
Cmt[a___, c_ + d_, b___] := Cmt[a, c, b] + Cmt[a, d, b];
Cmt[a___, n_ c_Cmt, b___]:= n Cmt[a, c, b];
Cmt[a___, n_ X, b___]:= n Cmt[a, X, b];
Cmt[a___, n_ Y, b___]:= n Cmt[a, Y, b];
Cmt /: Format[Cmt[a_, b_]]:= SequenceForm["[", a, ",", b, "]"];

ad[a_, 0, b_]:= b;
ad[a_, j_Integer, b_]:= Cmt[a, ad[a, j-1, b]];
ff[1, k_]:= ff[1, k] =
    Sum[((-1)^k/(j! (k-j)!)) ad[Y, k-j, ad[X, j, Y]], {j, 1, k}];
cc[2] = (1/2) ff[1, 1];
ff[p_, k_]:= ff[p, k] =
    Sum[((-1)^j/j!) ad[cc[p], j, ff[p-1, k - p j]], {j, 0,
        IntegerPart[k/p] - 1}];
cc[p_Integer]:= cc[p] =
    Expand[(1/p) ff[IntegerPart[(p-1)/2], p-1]];
```

The object $\text{Cmt}[x_1, x_2, \ldots, x_{n-1}, x_n]$ refers to the nested commutator $[x_1, [x_2, \ldots [x_{n-1}, x_n] \cdots]]$. It has attached just the linearity property (there is no need to attach to it the antisymmetry property and the Jacobi identity). The symbol ad represents the adjoint operator and its powers $\text{ad}_a^j b$, whereas ff[1,k], ff[p,k] and cc[p] correspond to expressions (2.12), (2.13) and (2.16), respectively. Proceeding in this way, we have obtained the expressions of C_n up to $n = 20$ with a reasonable computational time and memory requirements. Thus, for instance, constructing the terms up to degree $n = 20$ with a personal computer (2.4 GHz Intel Core 2 Duo processor with 2 GBytes of RAM) takes less than 20 seconds of CPU time and 35 MBytes of memory. The expression for C_{20} has 48528 terms, all of them independent. The resulting expressions up to C_8 are identical to those expressed in the classical Hall basis.

In Table 1 we collect the CPU time (in seconds) and memory (in MBytes) needed to construct the terms $C_2, C_3, \ldots, C_n$ up to a given value of n both with the recurrence (2.17) (*New*) and the implementation provided in [20] using a variant of the so-called comparison method previously introduced in [21] (*W-S*). Notice that with this method, which is the most efficient of all the procedures analyzed in [20], the Zassenhaus exponents C_j are expressed as linear combinations of words of length j and not directly in terms of independent commutators (although this is always possible by applying Dynkin–Specht–Wever theorem or Theorem 2 in [20], and then simplifying the resulting expressions by taking into

account the Jacobi identity and the antisymmetry property of the commutator). For comparison, C_{16} has 54146 terms when expressed as combinations of words, but only 3711 terms with the new formulation. This translates directly into the memory requirements of both algorithms, as is evident from the results collected in the table.

n	CPU time (seconds)		Memory (MBytes)	
	W-S	*New*	*W-S*	*New*
14	29.18	0.14	122.90	0.88
16	203.85	0.59	764.32	4.09
18		3.01		11.12
20		19.18		35.27

Table 1: CPU time and memory required for the computation of the Zassenhaus terms $C_2, C_3, \ldots, C_n$ up to the given value of n using the algorithm presented in [20] (*W-S*) and recurrence (2.17) (*New*).

3   Convergence of the Zassenhaus formula

Suppose now that X and Y are defined in a Banach algebra $\mathcal{A}$, that is to say, an algebra that is also a complete normed linear space whose norm is submultiplicative,

$$\|X\,Y\| \leq \|X\|\,\|Y\| \tag{3.1}$$

for any two elements of $\mathcal{A}$. Notice that for the commutator one has $\|[X,Y]\| \leq 2\,\|X\|\,\|Y\|$. Then it makes sense to analyze the convergence of the Zassenhaus formula (1.2).

As stated in the introduction, we are aware of only two previous results establishing sufficient conditions for convergence of the form $\|X\| + \|Y\| < r$ with a given $r > 0$. Specifically, Suzuki [16] obtained $r_s = \log 2 - \frac{1}{2} \approx 0.1931$, whereas Bayen [23] proved that the domain of convergence can be enlarged up to a value of r_b given by the unique positive solution of the equation

$$z^2 \left(1 + 2\int_0^z \frac{\mathrm{e}^{2w} - 1}{w}dw\right) = 4(2\log 2 - 1).$$

A numerical computation shows that $r_b = 0.59670569\ldots$. Thus for $\|X\| + \|Y\| < r_b$ one has

$$\lim_{n\to\infty} \mathrm{e}^X\,\mathrm{e}^Y\,\mathrm{e}^{C_2}\cdots\mathrm{e}^{C_n} = \mathrm{e}^{X+Y}. \tag{3.2}$$

In the following, we use recursion (2.8)–(2.10) to show that (3.2) holds indeed for $(x,y) \equiv (\|X\|, \|Y\|) \in \mathbb{R}^2$ in a domain that is larger than $\{(x,y) \in \mathbb{R}^2 \ : \ 0 \leq x + y < r_b\}$.

Clearly, (3.2) holds if

$$\lim_{n\to\infty} \|R_n(1)\| = 1, \tag{3.3}$$

10

where $R_n(\lambda)$ is given by (2.4), and thus is the solution of the initial value problem

$$\frac{d}{d\lambda}R_n(\lambda) = F_n(\lambda)R_n(\lambda), \qquad R_n(0) = I. \tag{3.4}$$

It is well known that, if $\int_0^1 \|F_n(\lambda)\|d\lambda < \infty$, then there exists a unique solution $R_n(\lambda)$ of (3.4) for $0 \le \lambda \le 1$, and that $\|R_n(1)\| \le \exp(\int_0^1 \|F_n(\lambda)\|d\lambda)$. In consequence, convergence (3.3) will be guaranteed whenever $(x,y) = (\|X\|, \|Y\|) \in \mathbb{R}^2$ is such that

$$\lim_{n\to\infty} \int_0^1 \|F_n(\lambda)\|d\lambda = 0.$$

From (2.19) we have that $\|C_{n+1}\| \le \delta_{n+1}$, where $\delta_2 = x\,y$ and for $n \ge 2$,

$$\delta_{n+1} = \frac{1}{n+1} \sum_{(i_0,i_1,\dots,i_n)\in\mathcal{I}_n} \frac{2^{i_0+\cdots+i_n}}{i_0!i_1!\cdots i_n!} \delta_n^{i_n} \cdots \delta_2^{i_1} y^{i_1} x^{i_0} y.$$

Similarly, $\|F_n(\lambda)\| \le f_n(\lambda)$, where

$$f_1(\lambda) \;=\; \sum_{i_1=0}^{\infty}\sum_{i_0=1}^{\infty} \frac{(2\lambda)^{i_0+i_1}}{i_0!i_1!} y^{i_1} x^{i_0} y = \mathrm{e}^{2\lambda y}(\mathrm{e}^{2\lambda x} - 1)y, \tag{3.5}$$

and for $n \ge 2$,

$$f_n(\lambda) = \sum_{(i_0,i_1,\dots,i_n)\in\mathcal{J}_n} \frac{2^{i_0+\cdots+i_n}\lambda^{i_0+i_1+2i_2+\cdots+ni_n}}{i_0!i_1!\cdots i_n!} \delta_n^{i_n} \cdots \delta_2^{i_2} y^{i_1} x^{i_0} y.$$

Note that this implies

$$\int_0^1 f_n(\lambda)d\lambda \le \sum_{k=n}^{\infty} \delta_k,$$

so that (3.3) is ensured if the series $\sum_{k=2}^{\infty} \delta_k$ converges. Let us analyze each term of this series by mimicking the recursive procedure given by (2.17). From (2.12) (or alternatively from (3.5)) and (2.13), we get

$$\|f_{1,k}\| \;\le\; d_{1,k} \equiv 2^k y \sum_{j=1}^{k} \frac{1}{j!(k-j)!} x^j y^{k-j} = \frac{2^k}{k!} y\big((x+y)^k - y^k\big)$$

$$\|f_{n,k}\| \;\le\; d_{n,k} = \sum_{j=0}^{[k/n]-1} \frac{2^j}{j!} \delta_n^j d_{n-1,k-nj} \tag{3.6}$$

whence

$$\|C_n\| \le \delta_n = \frac{1}{n} d_{[(n-1)/2],n-1}, \qquad n \ge 3. \tag{3.7}$$

A sufficient condition for convergence is obtained by imposing

$$\lim_{n\to\infty} \frac{\delta_{n+1}}{\delta_n} < 1. \tag{3.8}$$

11

At this point it is worth remarking that, although not reflected by the notation, both $d_{n,k}$ and δ_n depend on $(x,y) = (\|X\|, \|Y\|)$, so condition (3.8) implies in fact a constraint on the convergence domain $(x,y) \in \mathbb{R}^2$ of the Zassenhaus formula. In Figure 1, we show graphically the (numerically computed) domain $\mathcal{D}$ of such points (x,y). This has been obtained by computing for each point the coefficients $d_{n,k}$ and δ_n up to $n = 1000$ (in fact, considering a smaller value of n the figure does no change significantly). We have also included for comparison the previous results $x + y < 0.1931$ and $x + y < 0.5967$ of Suzuki and Bayen, respectively. Observe that the new convergence domain is considerably larger. In particular, it contains the region $x + y < 1.054$, but it is not restricted to that. For instance, the convergence domain contains the sets $\{(x, 2.9216) \ : \ x < 0.00292\}$ and $\{(2.893, y) \ : \ y < 0.0145\}$, and also the points $(x, 0)$ and $(y, 0)$ with arbitrarily large value of x or y.
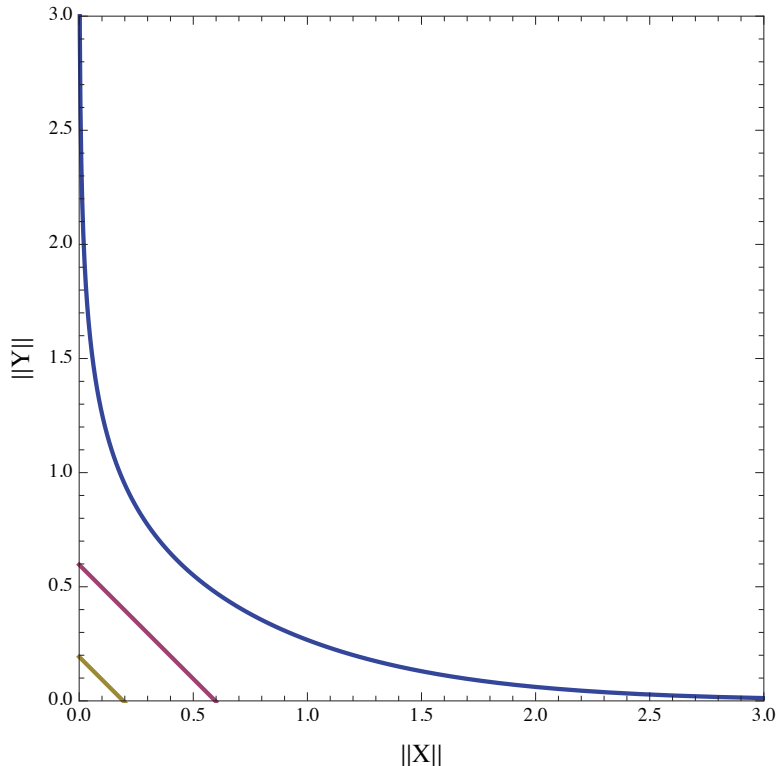

Figure 1: Upper boundary of convergence domain for the Zassenhaus formula obtained with the procedure (3.6)-(3.7). Previous results $0 < \|X\| + \|Y\| < 0.1931$ and $0 < \|X\| + \|Y\| < 0.5967$ are also depicted for comparison. The new domain contains the region $\|X\| + \|Y\| < 1.054$.

In summary, we have presented a new recursive procedure that not only allows us to get the expressions of the Zassenhaus exponents C_n directly in terms of independent commutators in an efficient way and can be easily implemented in any symbolic algebra package, but also shows, by bounding appropriately each term in the recursion, that the convergence domain of the Zassenhaus

formula is considerably larger than the domain guaranteed by previously known results.

Acknowledgments

References

[1] W. Magnus, On the exponential solution of differential equations for a linear operator, *Commun. Pure Appl. Math.* **7** (1954), 649-673.

[2] V.V. Gorbatsevich, A.L. Onishchik, and E.B. Vinberg. *Foundations of Lie Theory and Lie Transformation Groups.* Springer, 1997.

[3] E. Hairer, Ch. Lubich, and G. Wanner. *Geometric Numerical Integration. Structure-Preserving Algorithms for Ordinary Differential Equations.* Springer-Verlag, Second edition, 2006.

[4] A. J. Dragt and J. M. Finn, Lie series and invariant functions for analytic symplectic maps, *J. Math. Phys.* **17** (1976), 2215-2227.

[5] G.H. Weiss and A.A. Maradudin, The Baker–Hausdorff formula and a problem in Crystal Physics, *J. Math. Phys.* **3** (1962), 771-777.

[6] K. Kumar, On expanding the exponential, *J. Math. Phys.* **6** (1965), 1928-1934.

[7] R.M. Wilcox, Exponential operators and parameter differentiation in quantum physics, *J. Math. Phys.* **8** (1967), 962-982.

[8] A.T. Sornborger and E.D. Stewart, Higher-order methods for simulations on quantum computers, *Phys. Rev. A* **60** (1999), 1956-1965.

[9] M. Torres-Torriti and H. Michalska, A software package for Lie algebraic computations, *SIAM Review* **47** (2005), 722-745.

[10] A. Bonfiglioli and R. Fulci, *Topics in Noncommutative Algebra. The Theorem of Campbell, Baker, Hausdorff and Dynkin.* Lecture Notes in Mathematics 2034, Springer, 2012.

[11] F. Casas and A. Murua, An efficient algorithm for computing the Baker–Campbell–Hausdorff series and some of its applications, *J. Math. Phys.* **50** (2009), 033513.

[12] C. Quesne, Disentangling q-exponentials: a general approach, *Int. J. Theor. Phys.* **43** (2004), 545-559.

[13] A.J. Dragt, *Lie methods for Nonlinear Dynamics with Applications to Accelerator Physics*, available online at `http://www.physics.umd.edu/dsat/`.

[14] A. Iserles and K. Kropielnicka, Effective approximation for linear time-dependent Schrödinger equation, *Technical Report NA2011/15*, University of Cambridge (2011).

[15] H.C. Volkin, Iterated commutators and functions of operators, *NASA Technical Note D-4857*, Washington D.C., 1968.

[16] M. Suzuki, On the convergence of exponential operators—the Zassenhaus formula, BCH formula and systematic approximants, *Commun. Math. Phys.* **57** (1977), 193-200.

[17] H.J. Baues, *Commutator Calculus and Groups of Homotopy Classes.* Cambridge University Press, 1981.

[18] V. Kurlin, The Baker–Campbell–Hausdorff formula in the free metabelian Lie algebra, *J. Lie Theory* **17** (2007), 525-538.

[19] D. Scholz and M. Weyrauch, A note on the Zassenhaus product formula, *J. Math. Phys.* **47** (2006), 033505.

[20] M. Weyrauch and D. Scholz, Computing the Baker–Campbell–Hausdorff series and the Zassenhaus product, *Comput. Phys. Comm.* **180** (2009), 1558-1565.

[21] W. Witschel, Ordered operator expansions by comparison, *J. Phys. A: Math. Gen.* **8** (1975), 143-155.

[22] N. Jacobson. *Lie Algebras.* Dover, 1979.

[23] F. Bayen, On the convergence of the Zassenhaus formula, *Lett. Math. Phys.* **3** (1979), 161-167.

[24] C. Reutenauer. *Free Lie algebras.* Oxford University Press, 1993.