Computing Logarithmic Parts by Evaluation Homomorphisms*

Hao Du
School of Sciences,
Beijing University of Posts and Telecommunications
Beijing (102206), China
haodu@bupt.edu.cn

Yiman Gao
Key Lab of Mathematics Mechanization, AMSS,
University of Chinese Academy of Sciences,
Chinese Academy of Sciences, Beijing (100190), China
ymgao@amss.ac.cn

Jing Guo
Data Communication Science and Technology
Research Institute, Beijing (100191), China
JingG@amss.ac.cn

Ziming Li
Key Lab of Mathematics Mechanization, AMSS,
University of Chinese Academy of Sciences,
Chinese Academy of Sciences, Beijing (100190), China
zmli@mmrc.iss.ac.cn

ABSTRACT

We present two evaluation-based algorithms: one for computing logarithmic parts and the other for determining complete logarithmic parts in transcendental function integration. Empirical results illustrate that the new algorithms are markedly faster than those based respectively on resultants, the contraction of ideals, subresultants and Gröbner bases. They may be used to accelerate Risch's algorithm for transcendental integrands, and help us to compute elementary integrals over logarithmic towers efficiently.

CCS CONCEPTS

• **Computing methodologies → Algebraic algorithms**.

KEYWORDS

Additive decomposition, Elementary integral, Evaluation homomorphism, Logarithmic part, Symbolic integration

ACM Reference Format:

Hao Du, Yiman Gao, Jing Guo, and Ziming Li. 2023. Computing Logarithmic Parts by Evaluation Homomorphisms. In *International Symposium on Symbolic and Algebraic Computation 2023 (ISSAC 2023), July 24–27, 2023, Tromsø, Norway*. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3597066.3597067

1 INTRODUCTION

Developing methods for indefinite integration has been active and challenging ever since the invention of calculus. It is neatly-formulated in terms of differential algebra by Ritt in [16] and Rosenlicht in [17, 18]. Risch [14, 15] presents a systematic approach to determining whether an elementary function has an elementary integral. See [13] for commentaries and details. His papers contain a complete algorithm for transcendental elementary integrands, in which computing logarithmic parts is a fundamental building block. The algorithm is described, refined, improved and extended in [1, 3, 6, 11, 19, 20]. Its implementations are available in computer algebra systems, e.g. Maple and Mathematica.

The integral of a rational function in $\mathbb{Q}(x)$ is the sum of another rational function and a linear combination of logarithmic functions over $\overline{\mathbb{Q}}$. Such a linear combination is called the logarithmic part of the integral. It can be found by expanding a Rothstein-Trager resultant and performing gcd-computation over several algebraic number fields. See [21], [6, §11.5] and [1, §2.4] for details. Two alternative algorithms are presented to avoid gcd-computation over algebraic number fields in [9, 10] and [2], respectively. The former uses the subresultant algorithm. The latter needs to compute a Gröbner basis of some zero-dimensional ideal.

Logarithmic parts, Rothstein-Trager resultants and the above-mentioned algorithms are valid for monomial extensions of various kinds due to the results in [20, Theorem 2], [1, §5.6] and [12, Theorem 4]. Moreover, Lemma 6 in [12] leads to another algorithm using the contraction of ideals. Elements of monomial extensions are multivariate polynomials and their fractions. So intermediate expression swell frequently occurs when resultants, subresultants or Gröbner bases are computed in such extensions.

For an element f in a monomial extension, the logarithmic part of its integral can be constructed by the Rothstein-Trager resultant of f, which is a univariate polynomial over a differential field. The resultant can be factored as a product $u\,v$, where u is a monic polynomial with constant coefficients, and each nontrivial monic factor of v has nonconstant coefficients. The logarithmic part is determined by u, which is much smaller in size than the resultant. This observation makes it possible to control intermediate expression swell. We present two evaluation-based algorithms for computing logarithmic parts and for determining complete logarithmic parts, respectively. Some preliminary results of this paper are contained in the doctoral dissertation of the third author [7].

We had focused merely on determining complete logarithmic parts in Risch's algorithm. James Davenport raised a question about how to compute logarithmic parts in the same manner when part of this work was presented at the 27th International Conference on Applications of Computer Algebra, Gebze-Istanbul, 2022. His question widened the scope of this project and simplified our results.

The rest of this paper is organised as follows. We review basic notions for symbolic integration in Section 2, and define logarithmic parts in terms of residues in Section 3. Evaluation-based algorithms

*H. Du was partially supported by an NSFC grant (# 12201065). Y. Gao, J. Guo and Z. Li were partially supported by two NFSC grants (# 11971029 , 12271511).

ISSAC 2023, July 24–27, 2023, Tromsø, Norway

ACM ISBN 979-8-4007-0039-2/23/07.
https://doi.org/10.1145/3597066.3597067

and their comparison with known ones are presented in Section 4. With the help of the additive decomposition in [4], we compute elementary integrals over logarithmic towers, and compare our method with the Maple function `int` in Section 5.

2 PRELIMINARIES

Let F be a field. For a nonzero polynomial $p \in F[t]$, $\deg_t(p)$ and $\mathrm{lc}_t(p)$ stand for its degree and leading coefficient, respectively. We say that p is *monic* if $\mathrm{lc}_t(p) = 1$. The *monic associate* of p is defined to be $p/\mathrm{lc}_t(p)$, which is denoted by $\mathrm{ma}_t(p)$. For $p, q \in F[t] \setminus \{0\}$ with $\max(\deg_t(p), \deg_t(q)) > 0$, their Sylvester resultant is denoted by $\mathrm{resultant}_t(p, q)$. An element of $F(t)$ is said to be *t-proper* if the degree of its numerator is lower than that of the denominator. In particular, zero is t-proper.

A derivation δ on a field F is an additive map from F to itself such that for all $x, y \in F$, $\delta(xy) = \delta(x)y + x\delta(y)$. The pair (F, δ) is called a *differential field*. An element c of F is called a *constant* if $\delta(c) = 0$. Denote $\{c \in F \mid \delta(c) = 0\}$ by C_F, which is a subfield of F. Let (F, δ) and (E, Δ) be two differential fields. We say that E is a *differential field extension* of F, or, equivalently, F is a *differential subfield* of E if F is a subfield of E and $\delta = \Delta|_E$. We still denote the derivation Δ on E by δ when there is no confusion arising.

Notation. Throughout this paper we assume that $(F, ')$ is a differential field of characteristic zero.

The derivation of F can be uniquely extended to its algebraic closure $\overline{F}$, and $C_{\overline{F}} = \overline{C_F}$ by [1, Corollary 3.3.1]. An element f of F is called a *logarithmic derivative* in F if it is equal to g'/g for some nonzero $g \in F$. Denote by $\mathcal{L}(F)$ the linear subspace spanned by all logarithmic derivatives over C_F. For $f \in \mathcal{L}(F)$, there exist $c_1, \ldots, c_n \in C_F$ and $g_1, \ldots, g_n \in F \setminus \{0\}$ such that $f = \sum_{i=1}^n c_i g_i'/g_i$. Then the integral of f can be expressed as $\sum_{i=1}^n c_i \log(g_i)$, where $\log(g_i)$ stands for an element in some differential field extension of F whose derivative is equal to g_i'/g_i.

Let t belong to a differential field extension of F. Then t is *primitive* if $t' \in F$, and t is *hyperexponential* if $t'/t \in F$. We call t a *monomial* over F if it is transcendental over F and its derivative belongs to $F[t]$.

Let t be a monomial over F. Then $F(t)$ is called a *monomial extension* of F, and t is said to be *regular* if $C_F = C_{F(t)}$. The ring $F[t]$ is closed under the derivation $'$. A polynomial $p \in F[t]$ is *normal* if $\gcd(p, p') = 1$, and it is *special* if $p \mid p'$ according to [1, Definition 3.4.2]. Normal polynomials are squarefree. Squarefree polynomials are normal if t is both primitive and regular. An element $f \in F(t)$ is said to be *t-simple* if it is t-proper and has a normal denominator. Zero is t-simple because 1 is both normal and special.

Example 2.1. *Let C be a field of characteristic 0. Then $(C(x), d/dx)$ is a differential field whose subfield of constants equals C.*

The subspace $\mathcal{L}(\mathbb{C}(x))$ consists of all x-simple functions.

Let $t = \exp(x)$. Then $t+1$ is normal and t is special as polynomials in $\mathbb{C}(x)[t]$. So $1/(t+1)$ is t-simple but $1/t$ is not.

For a nonzero polynomial $p \in F[z]$, we can uniquely decompose p as the product of p_S and p_N, where p_S is a monic polynomial in $C_F[z]$ and p_N is either an element of F or a polynomial whose monic factors have nonconstant coefficients. Regard z as a constant indeterminate. Then p_S is special, and every squarefree factor of p_N is normal. We call p_S and p_N the *special and non-special parts* of p, respectively.

Example 2.2. *Let F be given in Example 2.1, and z be an indeterminate over F. We set $z' = 0$. Then $F[z]$ is a differential ring. Monic special polynomials in $F[z]$ are the elements of $C[z]$.*

Let $p = xz^2 - z/x + z - xz + 1/x - 1$. Then its special and non-special parts are $z - 1$ and $xz - 1/x + 1$, respectively. They can be computed by either Algorithm **SplitFactor** *in [1, §3.5] or by taking the content and primitive part of the numerator of p with respect to x.*

3 LOGARITHMIC PARTS

Let t be a monomial over F, and $f \in F(t)$ be nonzero and t-simple. Write f as a/b, where $a, b \in F[t]$ and $\gcd(a, b) = 1$. For a root α of b, the *residue* of f at α is defined to be

$$\mathrm{residue}_t(f, \alpha) := \frac{a(\alpha)}{b'(\alpha)} \in \overline{F}. \tag{1}$$

The normality of b implies $b'(\alpha) \neq 0$. The residue is independent of the choices of denominators by a straightforward verification. Moreover, $\gcd(a, b) = 1$ implies $\mathrm{residue}_t(f, \alpha) \neq 0$. Our definition of residues is consistent with [1, Definition 4.4.1], which defines a residue as a natural projection. It appears that canonical images are more convenient to describe algorithms.

Let β be a residue of f and $\alpha_1, \ldots, \alpha_k$ be the distinct roots of b. The number of the appearances of β in the sequence:

$$\mathrm{residue}_t(f, \alpha_1),\ \mathrm{residue}_t(f, \alpha_2),\ \ldots,\ \mathrm{residue}_t(f, \alpha_k)$$

is called the *multiplicity* of β in [9]. Let z be a constant indeterminate over $F(t)$. The *Rothstein-Trager resultant* of f is defined to be $\mathrm{resultant}_t(a - zb', b)$ and is denoted by R_f. It is a nonzero polynomial in $F[z]$ by the assumption that $\gcd(a, b) = \gcd(b, b') = 1$.

The following lemma collects relevant results in [1, Theorem 4.4.3] and [9, Proposition 2]. It also describes the degree and leading coefficient of a Rothstein-Trager resultant.

Lemma 3.1. *Let t be a monomial over F, and f be a nonzero and t-simple element of $F(t)$. Write f as a/b with $a, b \in F[t]$ and $\gcd(a, b) = 1$. Assume that $k = \deg_t(b)$ and that $\alpha_1, \ldots, \alpha_k \in \overline{F}$ are the roots of b. Then the following assertions hold.*

(i) $f = \sum_{i=1}^{k} \mathrm{residue}_t(f, \alpha_i) \dfrac{(t - \alpha_i)'}{t - \alpha_i} + u$ *for some $u \in F[t]$.*

(ii) *Let $\beta_1, \ldots, \beta_\ell$ be the distinct residues of f with respective multiplicities $m_1, \ldots, m_\ell$, and let g_j be the monic greatest common divisor of $a - \beta_j b'$ and b in $F(\beta_j)[t]$ for $j = 1, \ldots, \ell$. With u given in (i), we have*

$$f = \sum_{j=1}^{\ell} \beta_j \frac{g_j'}{g_j} + u \quad \text{and} \quad m_j = \deg_t(g_j).$$

(iii) [1] $\deg_z(R_f) = k$. *With the β_j and m_j in (ii), we have*

$$R_f = (-1)^n \,\mathrm{lc}_t(b)^d \,\mathrm{resultant}_t(b', b) \prod_{j=1}^{\ell} (z - \beta_j)^{m_j},$$

where $n = k(d + 1)$ and $d = \deg_t(a - zb') - \deg_t(b')$.

[1] Shaoshi Chen reminded us that the assertions in (iii) were obtained in a seminar on symbolic integration at our lab in 2008.

Proof. (i) The irreducible partial fraction decomposition of f is

$$\sum_{i=1}^{k} \text{residue}_t(f, \alpha_i)\frac{\gamma_i}{t-\alpha_i}, \quad \text{where } \gamma_i = (t-\alpha_i)'|_{t=\alpha_i},$$

by (1) and a direct calculation. Then

$$u = \sum_{i=1}^{k} \text{residue}_t(f, \alpha_i)\frac{\gamma_i - (t-\alpha_i)'}{t-\alpha_i}.$$

So $u \in \overline{F}[t]$ by $(t-\alpha_i) \mid \gamma_i - (t-\alpha_i)'$ for all i with $1 \le i \le k$. Moreover, $u \in F[t]$ because u is symmetric in $\alpha_1, \ldots, \alpha_k$ over F.

(ii) For all j with $1 \le j \le \ell$, we let

$$h_j = \prod_{\text{residue}_t(f,\alpha_i)=\beta_j} (t-\alpha_i). \tag{2}$$

By (i), $f = \sum_{j=1}^{\ell} \beta_j h_j'/h_j + u$. Note that $h_j = g_j$, because they are monic, squarefree and have the same roots. So (ii) holds.

(iii) Let $\lambda = \text{lc}_t(b)$. Expressing R_f by the roots of b yields

$$R_f = (-1)^{ke}\lambda^e \prod_{i=1}^{k}(a(\alpha_i) - zb'(\alpha_i)),$$

where $e = \deg_t(a - zb')$. By (1), we have

$$R_f = (-1)^{ke+k}\lambda^e \left(\prod_{i=1}^{k} b'(\alpha_i)\right)\left(\prod_{i=1}^{k}(z - \text{residue}_t(f,\alpha_i))\right).$$

Since b is normal, $\deg_z(R_f) = k = \deg_t(b)$. Moreover,

$$R_f = (-1)^{ke+k}\lambda^e \left(\prod_{i=1}^{k} b'(\alpha_i)\right)\left(\prod_{j=1}^{\ell}(z-\beta_j)^{m_j}\right),$$

which, together with

$$\text{resultant}_t(b', b) = (-1)^{k\deg_t(b')}\lambda^{\deg_t(b')}\left(\prod_{i=1}^{k} b'(\alpha_i)\right),$$

implies that (iii) holds. □

With the notation introduced in Lemma 3.1, we assume further that $\beta_1, \ldots, \beta_s \in \overline{C}_F$ and $\beta_{s+1}, \ldots, \beta_\ell \notin \overline{C}_F$. Then

$$f = \sum_{j=1}^{s} \beta_j \frac{g_j'}{g_j} + \sum_{j=s+1}^{\ell} \beta_j \frac{g_j'}{g_j} + u$$

for some $u \in F[t]$ by Lemma 3.1 (ii). Then

$$\int f = \sum_{j=1}^{s} \beta_j \log(g_j) + \sum_{j=s+1}^{\ell} \int \beta_j \frac{g_j'}{g_j} + \int u.$$

Definition 3.2. *We call* $\sum_{j=1}^{s} \beta_j \log(g_j)$ *the* logarithmic part of the integral of f with respect to t. *When $s = \ell$, the logarithmic part is said to be* complete.

Proposition 3.3. *Let $C = C_F$, t be a monomial over F, and f be a nonzero and t-simple element of $F(t)$. Then the following assertions are equivalent.*

(i) *The integral of f has a complete logarithmic part.*
(ii) *All residues of f belong to $\overline{C}$.*
(iii) *All roots of R_f belong to $\overline{C}$.*
(iv) *The monic associate of R_f belongs to $C[z]$.*

Assume further that t is primitive and regular over F. Then the above assertions are equivalent to each of the following assertions.

(v) *The integral of f is equal to its logarithmic part.*
(vi) *The integral of f is elementary over $F(t)$.*

Proof. The equivalences among (i), (ii), (iii) and (iv) are immediate from Lemma 3.1.

In the rest of this proof, we set $f = a/b$, where $a, b \in F[t]$ and $\gcd(a, b) = 1$. Furthermore, let $\beta_1, \ldots, \beta_\ell$ be the distinct residues of f, where $\beta_1, \ldots, \beta_s \in \overline{C}$ and $\beta_{s+1}, \ldots, \beta_\ell \notin \overline{C}$. Furthermore, let t be primitive and regular over F.

Since t is primitive, $(t-\alpha)'/(t-\alpha)$ is t-simple for every $\alpha \in \overline{F}$. By Lemma 3.1 (i) and (ii), $f = \sum_{j=1}^{\ell} \beta_j g_j'/g_j$, where g_j is the monic greatest common divisor of $a - \beta_j b'$ and b. Then (ii) implies (v) due to $s = \ell$. Assume that (v) holds. So does (vi), because the integral of f is equal to $\sum_{j=1}^{s} \beta_j \log(g_j)$. Assume that (vi) holds. Then (vi) implies (ii) by [12, Theorem 3 (ii)], □

Theorem 3 in [12] corrects Theorem 5.6.1 in [1] by adding an assumption on regularity of t. Such an assumption is also indispensable in the above proposition, as illustrated below.

Example 3.4. *Let $F = \mathbb{C}(x)$ and t be a primitive monomial with $t' = 0$. Then t is not regular. It is direct to see that $f(t) = t/(t^2 + x)$ is t-simple and that $R_f = z^2 + x$. So none of the residues of f is a constant. But $\int f = t\log(t^2 + x)$, which is elementary over $F(t)$.*

4 ALGORITHMS

This section consists of three parts. First, we review known algorithms for computing logarithmic parts. Second, we present new algorithms using evaluation homomorphisms. At last, empirical results are given.

In this section, we let $C = C_F$, t be a monomial over F, and f be a nonzero and t-simple element of $F(t)$. To describe algorithms concisely, we further set $f = a/b$, where $a, b \in F[t]$ and $\gcd(a, b) = 1$. Moreover, let z be a constant indeterminate over $F(t)$. For an irreducible polynomial $p \in F[z]$, the monic greatest common divisor of $a - zb'$ and b over $F[z]/(p)$ is denoted by $\gcd(a - zb', b) \mod p$.

All the algorithms for computing logarithmic parts have the same input and output. Their input consists of a monomial extension $F(t)$ and an integrand $f \in F(t)$, and the output consists of the logarithmic part of the integral of f with respect to t and a boolean value indicating whether the logarithmic part is complete.

The first algorithm, named **RT**, expands R_f and computes the special part of $\text{ma}_z(R_f)$ in $F[z]$. It then computes irreducible factors $p_1, \ldots p_k$ of the special part over C, and $g_i(z, t) = \gcd(a - zb', b) \mod p_i$, $i = 1, \ldots, k$. Then the logarithmic part is equal to

$$\sum_{i=1}^{k} \sum_{p_i(\alpha_{i,j})=0} \alpha_{i,j} \log(g_i(\alpha_{i,j}, t)).$$

By Proposition 3.3, the logarithmic part is complete if and only if $\text{ma}_z(R_f)$ belongs to $C[z]$. Algorithm **RT** is essentially the same as Algorithm **ResidueReduce** based on Rothstein-Trager resultant reduction in [1, §5.6].

The second algorithm, named **CI**, is based on [12, Lemma 6], which asserts that the squarefree part of $\text{ma}_z(R_f)$ is the monic generator of $\langle a - zb', b\rangle \cap F[z]$, where $\langle a - zb', b\rangle$ stands for the algebraic ideal generated by $a - zb'$ and b in $F[z,t]$. By [12, Lemma 5], the ideal has a Gröbner basis $\{b, z - pa\}$ with respect to the lexicographic order $t \prec z$, where $pb' \equiv 1 \mod b$. The Gröbner basis enables us to construct the generator by linear algebra. Then we proceed as Algorithm **RT** with the generator instead of $\mathrm{ma}_z(R_f)$.

The third algorithm, named **SR**, is essentially the same as Algorithm **ResidueReduce** based on Lazard-Rioboo-Rothstein-Trager resultant reduction in [1, §5.6]. It computes a subresultant sequence of $a - zb'$ and b with respect to t, and R_f. Then the algorithm extracts the logarithmic part from the subresultant sequence by a carefully-designed process involving splitting factorization, squarefree factorization and gcd-computation in $F[z]$. But gcd-computation over any algebraic extension of C is not needed.

The fourth algorithm, named **GB**, is described in [12, Theorem 8]. It computes a minimal Gröbner basis of $\langle a - zb', b\rangle$ with respect to the lexicographic ordering $z \prec t$ by the half-extended Euclidean algorithm and linear algebra according to remarks on [12, pp. 1294-1295]. Then the logarithmic part can be constructed by taking leading coefficients and performing exact division. Gcd-computation over any algebraic extension of C is not needed either.

Elimination techniques used in the above algorithms cause intermediate expression swell, as illustrated below.

Example 4.1. *Let $F = \mathbb{Q}(x)$ and $t' = 1/x$. Let*

$$a = (64x^4 + 24x^3 - 24x^2 + 6x)t^2 + (32x^4 + 88x^3 - 40x^2 + 8x - 1)t + 16x^3 + 32x^2 - 22x + 2,$$

and b be the product of $x\,(2x-1)\,(4x^2+8x-1)$, $(2x-1)t+1$ and $(4x^2+8x-1)\,t^2 + (4x+4)t + 1$. Then $f = a/b$ is t-simple. Using Algorithm **RT***, we find*

$$R_f = p \cdot \underbrace{\left(z + \frac{1}{4}\right) \cdot \left(z^2 - \frac{1}{4}z - \frac{1}{16}\right)}_{\mathrm{ma}_z(R_f)},$$

where $p \in \mathbb{Q}[x]$ is of degree 27 and is irrelevant to the logarithmic part. The integral of f has a complete logarithmic part

$$-\frac{1}{4}\log\left(t + \frac{1}{2x-1}\right) + \sum_{\beta^2 - \frac{1}{4}\beta - \frac{1}{16} = 0} \beta \log\left(t + \frac{2x - 8\beta + 3}{4x^2 + 8x - 1}\right).$$

Applying Algorithm **CI** *to f, we need to compute the inverse of b' modulo b. It is a quadratic polynomial in t whose coefficients are fractions of dense polynomials in $\mathbb{Q}[x]$ with degrees up to 10. Similarly, R_f is computed in Algorithm* **SR***, and the same modular inverse is computed in Algorithm* **GB**.

On the other hand, for almost all $\alpha \in \mathbb{Q}$,

$$R_f(\alpha, z) = \mathrm{resultant}_t(a(\alpha,t) - zb'(\alpha,t), b(\alpha,t)).$$

Moreover, R_f and $R_f(\alpha,z)$ have the same monic associate with respect to z whenever α is not a root of p. So a substitution for x may enable us to compute the monic associate by operations in $\mathbb{Q}[z,t]$.

This example motivates us to compute the logarithmic part without fully expanding R_f. Our idea is to choose a subring of $F[z,t]$ and a homomorphism from the subring to $C[z,t]$ properly. Then we compute the homomorphic image of R_f in $C[z,t]$. Proposition 4.7 to be given in the sequel will guide us to find the logarithmic part by the image, factorization over C and gcd-computation over some algebraic extensions of C.

To this end, we impose some restrictions on F. From now on, let F be the field of rational functions over C in several indeterminates, say $y_1, \ldots, y_n$. For example, $C(x, \log(x))$ is understood as $C(y_1, y_2)$, where $y_1 = x$ and $y_2 = \log(x)$. The numerator and denominator of an element in $F(t)$ are taken to be two coprime polynomials in $C[y_1, \ldots, y_n, t]$, respectively.

Definition 4.2. *Let $\mathbf{v} \in C^n$ and the multiplicative subset*

$$S_{\mathbf{v}} = \{p \in C[y_1, \ldots, y_n] \mid p(\mathbf{v}) \neq 0\}.$$

We call

$$\begin{array}{rccc} \phi_{\mathbf{v}}: & S_{\mathbf{v}}^{-1}C[y_1, \ldots, y_n, z, t] & \longrightarrow & C[z,t] \\ & g(y_1, \ldots, y_n, z, t) & \mapsto & g(\mathbf{v}, z, t). \end{array}$$

the (evaluation) homomorphism *for $\mathbf{v}$. We say that $\phi_{\mathbf{v}}$ is* lucky *for f if the following three conditions are satisfied:*

(i) *the denominator of b' belongs to $S_{\mathbf{v}}$,*
(ii) $\mathrm{lc}_t(a), \mathrm{lc}_t(b), \mathrm{lc}_t(b') \notin \ker(\phi_{\mathbf{v}})$,
(iii) $\mathrm{resultant}_t(b', b) \notin \ker(\phi_{\mathbf{v}})$.

Remark 4.3. *There is an $(n-1)$-dimensional algebraic set in C^n containing every point $\mathbf{v} \in C^n$ such that $\phi_{\mathbf{v}}$ is unlucky for f.*

The first and second conditions can be verified easily. The next lemma provides a way to verify the third.

Lemma 4.4. *Let $f \in F(t)$ be nonzero and t-simple. Let $\mathbf{v} \in C^n$ satisfy (i) and (ii) in Definition 4.2. Then $\phi_{\mathbf{v}}$ is a lucky homomorphism for f if and only if $\deg_z(\phi_{\mathbf{v}}(R_f)) = \deg_t b$.*

Proof. Let $k = \deg_t(b)$. By Definition 4.2 (i), $\phi_{\mathbf{v}}$ is applicable to both b' and R_f. By Lemma 3.1 (iii),

$$R_f = \pm\,\mathrm{resultant}_t(b, b')\,\mathrm{lc}_t(b)^m z^k + \text{terms of degrees} < k$$

for some nonnegative integer m. Thus, Definition 4.2 (ii) implies that $\deg_z(\phi_{\mathbf{v}}(R_f)) = k$ if and only if Definition 4.2 (iii) holds. □

Below are some useful properties of lucky homomorphisms.

Lemma 4.5. *Let $\phi_{\mathbf{v}}$ be a lucky homomorphism for f. Then the following assertions hold.*

(i) $\phi_{\mathbf{v}}(R_f) = \mathrm{resultant}_t(\phi_{\mathbf{v}}(a - zb'), \phi_{\mathbf{v}}(b))$.
(ii) $\phi_{\mathbf{v}}(\mathrm{ma}_z(R_f)) = \mathrm{ma}_z(\phi_{\mathbf{v}}(R_f))$.
(iii) *Let p_S be the special part of $\mathrm{ma}_z(R_f)$. Then p_S is a factor of $\mathrm{ma}_z(\phi_{\mathbf{v}}(R_f))$ in $C[z]$.*

Proof. (i) By Definition 4.2 (ii), we have

$$\deg_t(a - zb') = \deg_t(\phi_{\mathbf{v}}(a - zb')) \text{ and } \deg_t(b) = \deg_t(\phi_{\mathbf{v}}(b)).$$

Then (i) holds, because the determinant formula for R_f and that for $\mathrm{resultant}_t(\phi_{\mathbf{v}}(a - zb'), \phi_{\mathbf{v}}(b))$ have the same order.

(ii) Let q be the denominator of b'. The denominator of $\mathrm{ma}_z(R_f)$ divides a power of $\mathrm{resultant}_t(b', b)\,\mathrm{lc}_t(b)q$ by Lemma 3.1 (iii). It follows from Definition 4.2 that $\mathrm{ma}_z(R_f) \in S_{\mathbf{v}}^{-1}C[y_1, \ldots, y_n, z]$, that is, $\phi_{\mathbf{v}}$ is applicable to $\mathrm{ma}_z(R_f)$. Consequently,

$$\phi_{\mathbf{v}}(R_f) = \phi_{\mathbf{v}}(\mathrm{lc}_z(R_f))\phi_{\mathbf{v}}(\mathrm{ma}_z(R_f)).$$

Then (ii) holds by taking the monic parts of the both sides of the above equality.

(iii) Let p_N be the non-special part of $\text{ma}_z(R_f)$. We have that $p_N \in S_{\mathbf{v}}^{-1}C[y_1,\dots,y_n][z]$, because p_S belongs to $C[z]$. It follows from $\text{ma}_z(f) = p_S\, p_N$ and (ii) that

$$\text{ma}_z(\phi_{\mathbf{v}}(R_f)) = \phi_{\mathbf{v}}(p_S)\phi_{\mathbf{v}}(p_N) = p_S\phi_{\mathbf{v}}(p_N).$$

Therefore, $p_S \mid \text{ma}_z(\phi_{\mathbf{v}}(R_f))$. □

Example 4.6. *In the situation described in Example 4.1, we further let $C = \mathbb{Q}$, and $y_1 = x$. Then ϕ_1 is lucky for f. Moreover,*

$$\phi_1(a) = 70t^2 + 87t + 28,\ \ \phi_1(b) = 11(t+1)(11t^2 + 8t + 1)$$

and $\phi_1(b') = 957t^3 + 1690t^2 + 925t + 148$. By Lemma 4.5 (i),

$$\phi_1(R_f) = \text{resultant}_t\left(\phi_1(a) - z\phi_1(b'),\ \phi_1(b)\right),$$

which is $363170005(4z+1)(16z^2 - 4z - 1)$. Its monic associate is equal to the special part of $\text{ma}_z(R_f)$ by Lemma 4.5 (ii).

The last step towards our evaluation-based algorithms consists in forming a logarithmic part and deciding whether the logarithmic part is complete.

Proposition 4.7. *Let $f = a/b \in F(t)$ be nonzero and t-simple. Assume that $p \in C[z]$ is the image of $\text{ma}_z(R_f)$ under a lucky homomorphism for f, and that the irreducible factorization of p over C is $p_1^{n_1}\cdots p_d^{n_d}$. Set $g_i(z,t)$ to be $\gcd(a - zb', b) \mod p_i$, $i = 1,\dots,d$. Then the logarithmic part of the integral of f is*

$$\sum_{i=1}^{d}\sum_{p_i(\beta)=0}\beta\log(g_i(\beta,t)),$$

where $\log(g_i(\beta,t))$ is set to be 0 if $g_i(\beta,t) = 1$. Moreover, we have three equivalent assertions:

(i) *the integral of f has a complete logarithmic part,*
(ii) $\sum_{i=1}^{d}\deg_z(p_i)\deg_t(g_i) = \deg_t(b)$,
(iii) $\deg_t(g_i) = n_i$, $i = 1,\dots,d$.

Proof. Let q_S and q_N be, respectively, the special and non-special parts of $\text{ma}_z(R_f)$. By Lemma 4.5 (iii), q_S is a factor of p. So we further assume that the irreducible factors of q_S are $p_1,\dots,p_e$, and that each of $p_{e+1},\dots,p_d$ is coprime with q_S. Since every monic factor of q_N has a nonconstant coefficient, each of $p_{e+1},\dots,p_d$ is coprime with $\text{ma}_z(R_f)$. In other words, none of the $p_{e+1},\dots,p_d$ divides R_f. It follows that $g_j(z,t) = 1$ for all j with $e+1 \le j \le d$. Then the logarithmic part of the integral of f is

$$\sum_{i=1}^{e}\sum_{p_i(\beta)=0}\beta\log(g_i(\beta,t)) = \sum_{i=1}^{d}\sum_{p_i(\beta)=0}\beta\log(g_i(\beta,t)).$$

It remains to show that (i), (ii) and (iii) are equivalent.

By Lemma 3.1 (ii) and (iii), $q_S = \prod_{i=1}^{e}\prod_{p_i(\beta)=0}(z-\beta)^{\deg_t(g_i)}$. It follows from Lemma 4.5 (iii) and $\gcd(p_j, R_f) = 1$ with $e+1 \le j \le d$ that $q_S \mid p_1^{n_1}\cdots p_e^{n_e}$. In addition, $g_j = 1$ for j with $e+1 \le j \le d$. So

$$\deg_t(g_i) \le n_i, \quad i = 1,\dots,d. \tag{3}$$

Moreover, Lemma 3.1 (iii) and Lemma 4.4 imply that

$$\deg_t(b) = \deg_z(R_f) = \deg_z(p). \tag{4}$$

Assume that (i) holds. Then e and d are equal. So $p = \text{ma}_z(R_f)$ and $q_S = \text{ma}_z(R_f)$ since $\text{ma}_z(R_f) \in C[z]$. Consequently, we have that $\deg_z(p) = \deg_z(q_S)$, which, together with (4), implies (ii).

Assume that (ii) holds. By (4), we have

$$\sum_{i=1}^{d}\deg_z(p_i)\deg_t(g_i) = \sum_{i=1}^{d}\deg_z(p_i)n_i.$$

So $\deg_t(g_i) = n_i$ for all i with $1 \le i \le d$ by (3), and thus (iii) holds.

Assume that (iii) holds. Then $\deg_t(g_i) > 0$ for all i with $1 \le i \le d$. So $d = e$. By Lemma 3.1 (ii), every root of p_i is a residue of f with multiplicity n_i. It follows from Lemma 3.1 (iii) that p is a divisor of R_f. Hence, $p = \text{ma}_z(R_f)$ by (4) and $\text{lc}_t(p) = 1$. Therefore, (i) holds by Proposition 3.3 (iv). □

We are ready to present an evaluation-based algorithm for computing logarithmic parts.

Algorithm EH.

Input: a monomial extension $F(t)$,
a nonzero and t-simple element $f \in F(t)$
Output: L, the logarithmic part of $\int f$, and $B \in \{0,1\}$ such that $B = 1$ if L is complete, and $B = 0$ otherwise

1. $a \leftarrow$ numerator of f, $b \leftarrow$ denominator of f, $w \leftarrow 0$
2. [*choose a lucky homomorphism*]
   for i **from** 1 **to** 10 **do**
     choose a point $\mathbf{v} \in C^n$ randomly
     if $\phi_{\mathbf{v}}$ satisfies both (i) and (ii) in Definition 4.2 **then**
       $r \leftarrow \text{resultant}_t(\phi_{\mathbf{v}}(a - zb'), \phi_{\mathbf{v}}(b))$
       if $\deg_z(r) = \deg_t(b)$ **then**
         $p \leftarrow \text{ma}_z(r)$, $w \leftarrow 1$, break the loop
       end if
     end if
   end do
3. [*handle the unlucky case*] **if** $w = 0$ **then return** the result of Algorithm **RT**$(F(t), f)$ **end if**
4. find the irreducible factors $p_1,\dots,p_d$ of p over C
5. [*form a logarithmic part*] $B \leftarrow 0$, $L \leftarrow 0$, $m \leftarrow 0$,
   for i **from** 1 **to** d **do**
     $g_i(z,t) \leftarrow \gcd(a - zb', b) \mod p_i$
     $L \leftarrow L + \sum_{p_i(\beta)=0}\beta\log(g_i(\beta,t))$
     $m \leftarrow m + \deg_z(p_i)\deg_t(g_i)$
   end do
6. [*check completeness*] **if** $m = \deg_t(b)$ **then** $B \leftarrow 1$ **end if**
7. **return** L, B

In step 2 of Algorithm **EH**, we try to choose a lucky homomorphism for f. The verification of lucky homomorphisms in step 2 is correct by Lemma 4.4. If we have failed to choose any lucky homomorphisms for ten times, then the algorithm will end by calling Algorithm **RT**$(F(t), f)$ to compute the logarithmic part in step 3. In fact, Algorithms **CI**, **SR** and **GB** can also be applied in step 3. We choose Algorithm **RT**, because it performs better than other algorithms in our experiments. The correctness of steps 4, 5 and 6 is immediate from Proposition 4.7.

We are not aware of any way to find a point $\mathbf{v} \in C^n$ such that $\phi_{\mathbf{v}}(\text{resultant}_t(b, b')) \neq 0$ without expanding the resultant. So we opt for choosing points in C^n randomly and verify if there is a point leading to a lucky homomorphism. This strategy succeeds with probability one by Remark 4.3. We choose an evaluation point for ten times without any particular reason. Usually, the first choice leads to a lucky homomorphism.

Next, we determine complete logarithmic parts. By Proposition 3.3, we modify Algorithms **RT**, **CI**, **SR** and **GB** as follows. Whenever $\mathrm{ma}_z(R_f)$ or its squarefree part is obtained, we check whether it belongs to $C[z]$. If the answer is negative, then "false" is returned. Otherwise, they proceed in the same way. The modified algorithms are named **RT***, **CI***, **SR*** and **GB***, respectively.

An evaluation-based algorithm, named **EH***, determines complete logarithmic parts. It can be regarded as Algorithm **EH** equipped with some early detections of the nonexistence of complete logarithmic parts. In its pseudo-code, "[]" stands for the empty list, $\mathrm{len}(S)$ for the length of a list S, and $S[i]$ for the ith element of S.

Algorithm EH*.

Input: a monomial extension $F(t)$,
a nonzero and t-simple element $f \in F(t)$
Output: FALSE if f has no complete logarithmic part;
the complete logarithmic part of $\int f$, otherwise

1. $a \longleftarrow$ numerator of f, $b \longleftarrow$ denominator of f, $S \longleftarrow []$
2. [*choose lucky homomorphisms*]
   for i **from** 1 **to** 10 **do**
   choose a point $\mathbf{v} \in C^n$ randomly
   if $\phi_\mathbf{v}$ satisfies both (i) and (ii) in Definition 4.2 **then**
   $r \longleftarrow \mathrm{resultant}_t\left(\phi_\mathbf{v}\left(a - zb'\right), \phi_\mathbf{v}(b)\right)$
   if $\deg_z(r) = \deg_t(b)$ **then** append $\mathrm{ma}_z(r)$ to S
   if $\mathrm{len}(S) = 2$ **then** break the loop **end if**
   end if
   end if
   end do
3. [*handle the unlucky case*] **if** $\mathrm{len}(S) < 2$ **then return** the result of Algorithm **RT**$^*(F(t), f)$ **end if**
4. [*detect the nonexistence of complete logarithmic parts*]
   if $S[1] \neq S[2]$ **then return** FALSE **end if**
5. compute the irreducible factorization $p_1^{n_1} \cdots p_d^{n_d}$ of $S[1]$ over C
6. [*form the complete logarithmic part*] set $L \longleftarrow 0$
   for i **from** 1 **to** d **do**
   $g_i(z,t) \longleftarrow \gcd(a - zb', b) \mod p_i$
   [*detect the nonexistence of complete logarithmic parts*]
   if $\deg_t(g_i) \neq n_i$ **then return** FALSE **end if**
   $L \longleftarrow L + \sum_{p_i(\beta)=0} \beta \log(g_i(\beta, t))$
   end do
7. **return** L

In step 2 of Algorithm **EH***, we try to choose two lucky homomorphisms. The result of Algorithm **RT**$^*(F(t), f)$ is returned in step 3 if we have failed to choose for ten times. Assume that two lucky homomorphisms are found. Note that $\mathrm{ma}_z(R_f)$ is invariant under every lucky homomorphism if it belongs to $C[z]$. So Lemma 4.5 (ii) implies that the integral does not have any complete logarithmic part if $S[1]$ and $S[2]$ are unequal. Usually, they are unequal if $\mathrm{ma}_z(R_f)$ has a nonconstant coefficient. Thus, the algorithm filters out most of the integrands that have no complete logarithmic part in step 4. The correctness of steps 5 and 6 follows from Proposition 4.7. Moreover, the nonexistence of complete logarithmic parts is disclosed as long as a degree constraint is not satisfied in step 6 by Proposition 4.7 (iii).

We now present empirical results. Maple scripts of the above algorithms and testing examples are available at
https://haodu007.github.io/publication/logpart-paper.

All timings given in the rest of this section are Maple CPU time and measured in seconds, where "∅" means that Maple CPU time exceeds an hour. Experiments were carried out with Maple 2021 on a computer with imac CPU 3.6GHZ, Intel Core i9, 16G memory.

Our experimental data is generated with the help of the Maple command `randpoly`. Each suite of data contains several groups. A group is indexed by an integer i and consists of five examples.

For the algorithms to compute logarithmic parts, a suite of data was obtained as follows. We set $F = \mathbb{Q}(x, t_1)$, where $t_1 = \log(x)$. Let $t_2 = \log(\log(x))$. Then t_2 was a logarithmic monomial over F. We generated three dense polynomials u_i, v_i and w_i of respective total degrees $\lfloor i/2 \rfloor$ $\lfloor i/2 \rfloor$, and $\lceil i/2 \rceil$ in x, t_1 and t_2. Set f_i to be the t_2-proper part of $2u_i'/u_i - 3v_i'/v_i + 1/w_i$. Then f_i had two constant residues 2 and -3. The average timings for $i = 6, 7, \ldots, 12$ are summarized in Figure 1.

i	6	7	8	9	10	11	12
EH	0.08	0.07	0.10	0.19	0.27	0.45	0.65
RT	0.10	0.17	0.35	1.20	2.52	15.13	32.38
CI	105.21	511.76	1691.64	∅	∅	∅	∅
SR	118.25	276.02	2073.99	∅	∅	∅	∅
GB	547.97	∅	∅	∅	∅	∅	∅

Figure 1: Logarithmic parts (rational residues)

Next, we show the timings for the algorithms to determine complete logarithmic parts.

We set $F = \mathbb{Q}(x)$ and $t = \exp\left(-x^2/2\right)$. Then t was a hyperexponential monomial over F. We generated two dense polynomials u_i and v_i of total degrees i in x and t. Set f_i to be the t-proper part of $4u_i'/u_i - 6v_i'/v_i$. The residues of f_i were 4 and -6. Figure 2 contains the average timings for $i = 11, 12, \ldots, 16$.

i	11	12	13	14	15	16
EH*	0.11	0.14	0.17	0.22	0.28	0.34
RT*	0.29	0.42	0.55	0.89	1.14	1.71
CI*	26.84	52.48	94.58	174.27	324.75	624.26
SR*	643.95	1505.84	3219.16	∅	∅	∅
GB*	114.86	205.93	326.97	632.11	1073.48	∅

Figure 2: Complete logarithmic parts (rational residues)

At last, we set $F = \mathbb{Q}(x, t_1)$ with $t_1 = \log(x)$. Let t_2 be the integral of $1/t_1$. Then t_2 was a primitive monomial over F. Let $p = 5z^4 - z^3 + 2$, which was irreducible over $\mathbb{Q}$. We generated a sparse polynomial u_i of total degrees i in y, x, t_1 and t_2. The option "sparse" was chosen because dense polynomials in four indeterminates occupied too much space when their degrees were high. Set f_i to be the t_2-proper part of $\sum_{p(y)=0} y u_i'/u_i$. The residues of f_i were exactly the roots of p. The average timings are summarized in Figure 3, in which $i = 1, 2, \ldots, 6$.

i	1	2	3	4	5	6
EH*	0.05	0.03	0.04	0.07	0.13	0.16
RT*	0.06	1.14	15.94	13.47	411.21	1767.90
CI*	0.06	1.10	63.47	39.72	3580.75	⊘
SR*	0.05	2.80	38.60	184.75	⊘	⊘
GB*	0.08	297.51	⊘	⊘	⊘	⊘

Figure 3: Complete logarithmic parts (quartic residues)

The high efficiency of Algorithms **EH** and **EH*** relies on good performance of the Maple function `resultant` for expanding the resultant of $a - z\tilde{b}$ and b with $a, b, \tilde{b} \in \mathbb{Q}[t]$, and the function `Gcd` for computing greatest common divisors of univariate polynomials over algebraic number fields [5, 8, 22].

Algorithms **RT** and **RT*** outperform other elimination-based algorithms for most of examples. One reason is that the denominators of our input functions are expressed as the products of several polynomials due to the way to generate them. Resultant computation takes advantage of multiplicative expressions, but the other algorithms ignore any factored form of denominators.

At present, our maple scripts are only applicable to integrands whose constant coefficients are rational numbers. In fact, Algorithms **EH** and **EH*** are both valid as long as one can factor univariate polynomials over C and perform gcd-computation of a finite algebraic extension of C. Assume further that $C = \mathbb{Q}(w_1, \ldots, w_m)$, where $w_1, \ldots, w_m$ are constant indeterminates over $\mathbb{Q}$. We are not yet able to modify Algorithms **EH** and **EH*** so that evaluation homomorphisms can be applied to $w_1, \ldots, w_m$ nontrivially, because the special part of a Rothstein-Trager resultant may have coefficients involving some of the w_i's.

5 APPLICATIONS

In this section, we describe some applications arising from additive decompositions in logarithmic and S-primitive towers.

We denote $\{1, 2, \ldots, n\}$ by $[n]$. Let K_0 be a field of characteristic zero, $t_1, \ldots, t_n$ be n indeterminates, and $K_i = K_0(t_1, \ldots, t_i)$ for $i \in [n]$. An element of K_n is said to be t_i*-proper* if it is free of $t_{i+1}, \ldots, t_n$ and is proper as a univariate rational function in $K_{i-1}(t_i)$.

Set P_0 to be $K_0[t_1, \ldots, t_n]$, P_i to be the additive subgroup consisting of all polynomials in $K_i[t_{i+1}, \ldots, t_n]$ whose coefficients are t_i-proper for $i \in [n-1]$, and P_n to be the additive subgroup consisting of all t_n-proper elements. Then $K_n = \bigoplus_{i=0}^n P_i$. Let π_i be the projection from K_n to P_i with respect to the above direct sum. For an element $f \in K_n$, we have $f = \sum_{i=0}^n \pi_i(f)$, which is called the *Matryoshka decomposition* of f with respect to $t_1, \ldots, t_n$ in [4].

From now on, we assume that $K_0 = C(x)$, where $C(x)$ is the differential field given in Example 2.1. The Matryoshka decomposition of an element in K_n is always with respect to $t_1, \ldots, t_n$. Assume further that t_i is a primitive and regular monomial over K_{i-1} for each $i \in [n]$. Then we have a primitive tower

$$\begin{array}{ccccccc} K_0 & \subset & K_1 & \subset & \cdots & \subset & K_n \\ \shortparallel & & \shortparallel & & & & \shortparallel \\ C(x) & & K_0(t_1) & & & & K_0(t_1, \ldots, t_n) \end{array} \tag{5}$$

whose subfield of constants is equal to C.

An element f of K_n is said to be *simple* if $\pi_0(f)$ is x-simple and $\pi_i(f)$ is t_i-simple in $K_{i-1}(t_i)$ for all $i \in [n]$. Every element of $\mathcal{L}(K_n)$ is simple by [4, Proposition 3.5].

Algorithms in Section 4 helps us determine whether a simple element of K_n belongs to $\mathcal{L}(\overline{C}K_n)$, where $\overline{C}K_n$ stands for the smallest field containing both $\overline{C}$ and K_n. Since each of the t_i is regular over K_{i-1}, the subfield of constants in $\overline{C}K_n$ is equal to $\overline{C}$.

Proposition 5.1. *Let K_n be the tower given in (5), and $r \in K_n$ be simple. Then $r \in \mathcal{L}(\overline{C}K_n)$ if and only if the integral of $\pi_i(r)$ has a complete logarithmic part with respect to t_i for all $i \in [n]$.*

Proof. Assume that the integral of $\pi_i(r)$ has a complete logarithmic part with respect to t_i for all $i \in [n]$. Then the integral equals its complete logarithmic part with respect to t_i by Proposition 3.3. Differentiating the integral, we see that $\pi_i(r) \in \mathcal{L}(\overline{C}K_n)$. In addition, $\pi_0(r) \in \mathcal{L}(\overline{C}K_0)$ by Example 2.1. So $r \in \mathcal{L}(\overline{C}K_n)$.

Conversely, let $r \in \mathcal{L}(\overline{C}K_n)$. By [4, Lemma 2.6 (ii)], there exist a t_n-simple element $s \in \mathcal{L}(\overline{C}K_n) \cap K_n$ and $h \in \mathcal{L}(\overline{C}K_{n-1}) \cap K_{n-1}$ such that $r = s + h$. Then $s = \pi_n(r)$. It follows from a direct induction that $\pi_i(h) \in \mathcal{L}(\overline{C}K_i) \cap K_i$ for $i \in [n-1]$. By the Matryoshka decomposition, $\pi_i(r) = \pi_i(h)$ for $i \in [n-1]$. So the integral of $\pi_i(r)$ has a complete logarithmic part w.r.t. t_i for $i \in [n]$. □

The tower K_n in (5) is said to be *S-primitive* if t_i' is simple for $i \in [n]$. It is *logarithmic* if $t_i' \in \mathcal{L}(K_{i-1})$ for $i \in [n]$. Logarithmic towers are S-primitive by [4, Proposition 3.5].

Let K_n be S-primitive. Then Algorithm **AddDecompInField** in [4] computes two elements $g, r \in K_n$ such that

$$f = g' + r \tag{6}$$

with three properties: (i) r is minimal in some sense, (ii) f is a derivative in K_n if and only if $r = 0$, and (iii) r is simple if f has an elementary integral over K_n. The last property is due to the remark below [4, Theorem 4.10]. We call r a *remainder* of f in K_n.

Let K_n be logarithmic. By [4, Theorem 4.10], $f \in K_n$ has an elementary integral over K_n if and only if r in (6) belongs to $\mathcal{L}(\overline{C}K_n)$, which is equivalent to that the integral of $\pi_i(r)$ has a complete logarithmic part with respect to t_i for $i \in [n]$ by Proposition 5.1.

Example 5.2. *Let $K_0 = \mathbb{C}(x)$ and $t = \arctan(x)$. Since t is a $\mathbb{C}$-linear combination of two logarithmic derivatives, $K_1 = K_0(t)$ is logarithmic. Let $a = -x(2x^2+2)t^3 - x^4t^2 + x(2x^4+5x^2+2)t - (x^3+2x)x$ and $b = t^2(x^2+1)(x^2+2)(t+x)$. Algorithm* **AddDecompInField** *yields $a/b = (x/t)' + r$, where*

$$r = \pi_0(r) + \pi_1(r) = -\frac{2x}{x^2+2} + \frac{-t + x^3 + x}{(x^2+1)t(t+x)}.$$

The remainder r is simple. But the integral of $\pi_1(r)$ has an incomplete logarithmic part , which is $-\log(t+x)$. Hence, f has no elementary integral over K_1. In fact,

$$\int \frac{a}{b} = \frac{x}{t} - \log(x^2+2) - \log(t+x) + \int \frac{1}{\arctan(x)}.$$

Using **AddDecompInField** and **EH***, we present an algorithm for determining elementary integrals over a logarithmic tower.

Algorithm **AddInt_log.**

Input: K_n as in (5), a logarithmic tower over K_0 and $f \in K_n$
Output: FALSE if f has no elementary integral over K_n; an elementary integral of f, otherwise

1. [*decompose*] compute $g, r \in K_n$ such that $f = g' + r$ by Algorithm **AddDecompInField**
2. [*detect in-field and non-elementary integrability*]
   if $r = 0$ **then return** g **end if**
   if r is not simple **then return** FALSE **end if**
3. [*determine complete logarithmic parts*] $s \leftarrow g$
   for i **from** 1 **to** n **do**
      if $\pi_i(r) \neq 0$ **then**
         $u \leftarrow$ Algorithm $\mathbf{EH}^*(K_{i-1}(t_i), \pi_i(r))$
         if u = FALSE then **return** FALSE **end if**
         $s \leftarrow s + u$
      end if
   end do
4. **return** $s + \int \pi_0(r)$

The correctness of this algorithm is due to properties (ii) and (iii) of Algorithm **AddDecompInField**, and Proposition 5.1.

We compared efficiency of the above algorithm with the Maple function `int`. Every integrand in our experimental data had an elementary integral over $\mathbb{Q}(x)$ so that `int` would not need to look for any closed-form beyond elementary functions.

In the first suite of experimental data, we set $K_2 = \mathbb{Q}(x, t_1, t_2)$, where $t_1 = \log(x)$ and $t_2 = \log(\log(x))$. We generated four dense polynomials p_i, q_i, r_i, s_i in x, t_1 and t_2 of respective total degrees $\lceil i/2 \rceil$, $\lfloor i/2 \rfloor$, i and i. Set the integrand $f_i = (p_i/q_i)' - 3r_i'/r_i + 2s_i'/s_i$. The average timings are summarized in Figure 4, in which **A** stands for our maple scripts for Algorithm **AddInt_log**.

i	4	5	6	7	8	9	10
A	0.50	6.86	27.71	17.37	32.65	402.75	506.58
`int`	0.70	7.61	31.35	29.47	51.74	376.05	574.73

Figure 4: Elementary integrals (rational residues)

All residues of the nonzero projections of remainders were rational numbers in this suite. Algorithm **AddInt_log** and `int` performed almost equally well.

In the second suite, the monomial extension of $\mathbb{Q}(x)$ is the same as that in the first. We generated two dense polynomials p_i and q_i of total degree i in x, t_1 and t_2, a sparse polynomial r_i of total degrees $\lfloor i/2 + 1 \rfloor$ in y, x, t_1 and t_2, and a sparse polynomial s_i of total degree $\lceil i/2 + 1 \rceil$ in y, x and t_1. Set the integrand to be

$$f_i = \left(\frac{p_i}{q_i}\right)' + \sum_{3y^2+y-1=0} y\frac{r_i'}{r_i} + \sum_{y^2+1=0} y\frac{s_i'}{s_i}.$$

The average timings are summarized in Figure 5.

The nonzero projections of remainders may have quadratic residues in this suite. Algorithm **AddInt_log** outperformed `int` as the index i was increasing.

For the examples in the two suites, Algorithm **AddInt_log** only slowed down slightly when Algorithm $\mathbf{EH}^*$ was replaced with Algorithm $\mathbf{RT}^*$. But this was not the case for the last suite of data.

i	6	7	8	9	10	11	12
A	4.58	4.33	8.17	26.22	84.77	170.99	492.85
`int`	11.14	16.54	37.31	101.88	⊘	⊘	⊘

Figure 5: Elementary integrals (quadratic residues)

We set $K_1 = \mathbb{Q}(x, t)$ with $t = \log(x)$, and generated two dense polynomials a_i and b_i of total degrees i in x and t. Moreover, a dense polynomial g_i was generated in $\mathbb{Q}[x, y, t]$ whose total degree is i. Set the integrand $f_i = (a_i/b_i)' + \sum_{y^3+y-1=0} y g_i'/g_i$. The average timings are summarized in Figure 6, where **AR** stands for the algorithm that replaces Algorithm $\mathbf{EH}^*$ in step 3 of Algorithm **AddInt_log** by Algorithm $\mathbf{RT}^*$ given in Section 4.

i	11	12	13	14	15	16
A	5.45	11.48	16.61	27.06	49.30	72.42
AR	129.06	233.77	361.06	541.10	901.61	1239.29
`int`	325.64	697.95	1275.67	2048.20	3331.69	⊘

Figure 6: Elementary integrals ($\mathbf{EH}^*$ vs $\mathbf{RT}^*$)

The timings in this figure reveal that Algorithm $\mathbf{EH}^*$ improves the efficiency of algorithms for indefinite integration as far as integrals have dense logarithmic parts involving irrational residues.

REMARK 5.3. *We also used Mathematica 12 and 13.1 to compute the integrals of examples in our data. Unfortunately, the command* `Integrate` *returned unevaluated integrals from time to time. So it is difficult for us to make any further comparison.*

Let K_n be S-primitive but not logarithmic, and $f \in K_n$. By (6) and [4, Theorem 4.10], f has an elementary integral over K_n if and only if $r \in \text{span}_C \{t_1', \ldots, t_n'\} + \mathcal{L}(\overline{C}K_n)$. The latter condition can be verified by [11, Theorem 3.9].

EXAMPLE 5.4. *Let $K_0 = \mathbb{Q}(x)$ and K_3 be generated by $t_1 = \log(x)$, $t_2 = \mathrm{Li}(x)$ and $t_3 = \log(\log(x))$. We determine an elementary integral of f whose additive decomposition is equal to $g' + r$, where*

$$g = xt_3 + \frac{t_2^2}{2} - \frac{t_2 x}{t_1} - \frac{x^2}{t_1} \quad \text{and} \quad r = \frac{2}{x} - \frac{24x - 11}{6xt_1} + \frac{1}{t_1 t_2}.$$

By a minor variation of the algorithm contained in the proof of [11, Theorem 3.9], we see that r belongs to $-4t_2' + \mathcal{L}(\overline{\mathbb{Q}}K_3)$. In other words, $r + 4t_2' \in \mathcal{L}(\overline{\mathbb{Q}}K_3)$. Note that each $\int \pi_j(r + 4t_2')$ has a complete logarithmic part with respect to t_j, $j = 1, 2, 3$. Indeed, Algorithm **EH** *yields the complete logarithmic parts of the integrals of the three projections. It turns out $\int f = g + 2\log(x) + 11/6\log(t_1) + \log(t_2) - 4t_2$. The integral of f is elementary over K_3 but not over K_0.*

ACKNOWLEDGMENTS

We thank Shaoshi Chen, James Davenport, Hui Huang, George Labahn, Clemens Raab and Elaine Wong for their encouragement, timely assistance and helpful comments. Special thanks go to the anonymous referees for their constructive suggestions and careful corrections.

REFERENCES

[1] Manuel Bronstein. 2005. *Symbolic Integration I: Transcendental Functions* (second ed.). Springer, Berlin Heidelberg New York.

[2] Günter Czichowski. 1995. A note on Gröbner bases and integration of rational functions. *Journal of Symbolic Computation* 20 (1995), 163–167.

[3] James H. Davenport. 1986. The Risch differential equation problem. *SIAM J. Comput.* 15 (1986), 903–918.

[4] Hao Du, Jing Guo, Ziming Li, and Elaine Wong. 2020. An additive decomposition in logarithmic towers and beyond. In *Proceedings of the 45th International Symposium on Symbolic and Algebraic Computation (ISSAC'20)*. Lille, France, ACM, New York, 146–153.

[5] Mark J. Encarnación. 1995. Computing GCDs of polynomials over algebraic number fields. *Journal of Symbolic Computation* 20 (1995), 299–313.

[6] Keith O. Geddes, Stephen R. Czapor, and George Labahn. 1992. *Algorithms for Computer Algebra*. Kluwer Academic Publishers.

[7] Jing Guo. 2021. *Additive Decompositions in Primitive Towers with Applications*. Ph.D. Dissertation. Chinese Academy of Sciences, Beijing, China (in Chinese).

[8] Seyed M. M. Javadi and Michael Monagan. 2009. In-place arithmetic for univariate polynomials over an algebraic number field. In *Proceedings of the Joint Conference of ASCM 2009 and MACIS 2009 (COE Lecture Note, 22)*. Kyushu University, 330–341.

[9] Daniel Lazard and Renaud Rioboo. 1990. Integration of rational functions: Rational computation of the logarithmic part. *Journal of Symbolic Computation* 9 (1990), 113–115.

[10] Thomas Mulders. 1997. A note on subresultants and the Lazard/Rioboo/Trager formula in rational function integration. *Journal of Symbolic Computation* 24 (1997), 45–50.

[11] Clemens G. Raab. 2012. *Definite Integration in Differential Fields*. Ph.D. Dissertation. RISC-Linz, Johannes Kepler University, Linz, Austria.

[12] Clemens G. Raab. 2012. Using Gröbner bases for finding the logarithmic part of the integral of transcendental functions. *Journal of Symbolic Computation* 47 (2012), 1290–1296.

[13] Clemens G. Raab and Michael F. Singer. 2022. *Integration in Finite Terms: Fundamental Sources*. Text & Monographs in Symbolic Computation, Springer.

[14] Robert H. Risch. 1969. The problem of integration in finite terms. *Trans. Amer. Math. Soc.* 139 (1969), 167–189.

[15] Robert H. Risch. 1970. The solution of integration in finite terms. *Bull. Amer. Math. Soc.* 76 (1970), 605–608.

[16] Joseph F. Ritt. 1948. *Integration in Finite Terms. Liouville's Theory of Elementary Methods*. Columbia University Press, New York.

[17] Maxwell Rosenlicht. 1968. Integration in finite terms. *Amer. Math. Monthly* 24 (1968), 153–161.

[18] Maxwell Rosenlicht. 1972. Liouville's theorem on functions with elementary integrals. *Pacific J. Math.* 79 (1972), 963–972.

[19] Michael Rothstein. 1976. *Aspects of Symbolic Integration and Simplification of Exponential and Primitive Functions*. Ph.D. Dissertation. University of Wisconsin, Madison, the United States.

[20] Michael Rothstein. 1977. A new algorithm for the integration of exponential and logarithmic functions. In *Proceedings of the 1977 MACSYMA Users Conference*. 263–274.

[21] Barry M. Trager. 1976. Algebraic factoring and rational function integration. In *Proceedings of SYMSAC' 76*. ACM, New York, 219–226.

[22] Mark van Hoeij and Michael Monagan. 2002. A modular GCD algorithm over number fields presented with multiple field extensions. In *Proceedings of the 27th International Symposium on Symbolic and Algebraic Computation (ISSAC' 02)*. Lille, France, ACM, New York, 109–116.