Introduction to symbolic integration

Bruno Salvy
INRIA Rocquencourt

March 22, 1993

[summary by Daniel Augot]

1. General Introduction

Scientific community is becoming aware of the power and utility of symbolic computations softwares. One of the most surprising achievements of these programs is the ability to perform formal integration, that is:

Given a function $f(z)$ (possibly with parameters), find a function $F(z)$ such that:

$$F'_z(z) = f(z). \tag{1}$$

Most mathematical students have learned how to find such primitives, generally with heuristics. The important point here is that symbolic softwares (like Maple) use *algorithms* which can:

(1) prove or disprove that the primitive of $f(z)$ can be expressed with elementary functions;
(2) find a solution to problem (1).

It is these techniques that we want to introduce here.

Our main interest will be the integration of rational functions and of purely transcendental functions (this will be made more precise later). We will focus on the integration of rational functions first, as the main algorithm can be derived (in structure) from this basic case. We consider that we have some basic tools "at hand", which are common to symbolic softwares. This toolkit includes gcd operations, extended Euclidean algorithm, and some linear algebra techniques.

2. Rational Functions

We want to solve the following problem: given $F \in \mathbb{Q}(z)$, we want to find $\int F$. As a preliminary step, we can write

$$F = P + \frac{Q}{R}$$

where P, Q, R are polynomials, Q and R are coprime and $\deg Q < \deg R$. The polynomial P can be easily integrated, so the remaining task is to compute a primitive of Q/R.

2.1. Structural approach. From the theoretical point of view, the following theorem is well known.

Theorem 1. *Let $R = \prod_{i=1\ldots n}(z-a_i)^{n_i}$ be the factorisation of R, the fraction Q/R can be written*

$$\frac{Q}{R} = \sum_{i=1\ldots n} \frac{B_i}{(z-a_i)^{n_i}}, \qquad B_i \in \mathbb{Q}[z], \qquad \deg B_i < n_i.$$

We can rewrite this sum

$$\frac{Q}{R} = \sum_{i=1\ldots n}\sum_{j=1\ldots n_i} \frac{b_{i,j}}{(z-a_i)^j}, \qquad b_{i,j} \in \mathbb{Q}.$$

Then the primitive of Q/R is

$$\int \frac{Q}{R} = \sum_{i=1\ldots n} b_{i,j} \log(z-a_i) - \sum_{i=1\ldots n} \sum_{j=2\ldots n_i} \frac{b_{i,j}}{(j-1)(z-a_i)^{j-1}}.$$

This gives some insight about the result: the simple factors of the denominator give logarithms in the result, while multiple factors give rational functions. However, this formula is not practicable, for many reasons:

The factorization of R has to be known, and it is not easy to compute. We shall see that it is not needed in practice.

Furthermore, this factorisation gives rise to algebraic numbers, whose use is expensive. One rule in computer algebra is "stay in $\mathbb{Q}$ as most as possible".

Consider the following rewriting of Q:

$$F = P + \frac{A}{D} + \frac{B}{E} \tag{2}$$

where A,B,D,E,P are polynomials in $\mathbb{Q}[z]$, D is square-free, and E has only multiple factors. We want to deal separately with A/D, which leads to logarithmic terms, and with B/E which leads to rational functions.

We recall that the square-free decomposition of a polynomial P is the decomposition

$$P = P_1 P_2^2 \cdots P_m^m$$

where the polynomials $P_1, \ldots, P_m$ are pairwise coprime, and each P_i is square-free. We shall call m the *highest multiplicity* of P. This decomposition is not hard to compute, using differentiation of polynomials and gcd operations (Consider it as given in the basic toolkit).

In fact the algorithm computes step by step:

$$F = Q' + \left(\frac{\tilde{A}}{\tilde{D}}\right)' + \frac{\tilde{B}}{\tilde{E}},$$

such that the highest multiplicity of $\tilde{E}$ has been decreased, so that we have to solve the same problem with lower highest multiplicity, until we are left with a fraction whose denominator is square-free.

This process of decreasing the highest multiplicity of the denominator is called the *Hermite reduction*, which gives the rational part, and the logarithmic part (for a square-free denominator) is found by the method of the *Rothstein-Trager* resultant.

2.2. The Hermite reduction. The algorithm dates back to Hermite. We sketch a modern version due to Mack. We present the iterative step of the method.

Input P/Q, $\deg P < \deg Q$, $\gcd(P,Q) = 1$, the highest multiplicity of Q being $m > 1$.

Output $R, B, E \in \mathbb{Q}[X]$ such that $P/Q = (R)' + B/E$, the highest multiplicity m' of E being lower than m.

- Obtain a square-free decomposition of Q, and define G and $G^\star$ as shown:

$$Q = Q_1 Q_2^2 \cdots Q_m^m = Q_1 \underbrace{(Q_2 Q_3 \cdots Q_m)}_{G^\star} \underbrace{Q_2 Q_3^2 \cdots Q_m^{m-1}}_{G}$$

- Compute $QG'/G^2 = Q_1 G^\star G'/G$, which can be seen to be prime to $G^\star$.
- Using the extended Euclidean algorithm, compute the Bezout coefficients A, B of P:

$$P = A Q_1 \frac{G^\star G'}{G} + B G^\star$$

This leads to

$$\frac{P}{Q} = -\left(\frac{A}{G}\right)' + \frac{A' Q_1 + B}{Q_1 G}.$$

– Return $R = -A/G$, $B = A'Q_1 + B$, $E = Q_1 G$. The highest multiplicity of E is now at most $m-1$.

EXAMPLE. We consider the following function

$$f = \frac{z^7 - 24z^4 - 4z^2 + 8z - 8}{z^8 + 6z^6 + 12z^4 + 8z^2} = \frac{P}{Q}$$

Using this algorithm:

$$\begin{aligned} G &= \gcd(Q, Q') = z^5 + 4z^3 + 4z \\ Q_1 G^\star &= Q/G = z^3 + 2z \\ \gcd(G, G^\star) &= z^2 + 2 \\ Q_1 &= (Q_1 G^\star)/(G/\gcd(G, G')) = 1 \\ Q_1 G^\star G'/G &= 5z^2 + 2. \end{aligned}$$

By Bezout, we have $P = -(8z^2+4)(5z^2+2) + (z^4 - 2z^2 + 16z + 4)(z^3 + 2z)$. Thus

$$f = \left(\frac{8z^2+4}{z^5+4z^3+4z}\right)' + \frac{z^4 - 2z^2 + 4}{z^5 + 4z^3 + 4z}.$$

Again, for the second term, the algorithm gives:

$$\begin{aligned} G &= z^2 + 2 \\ Q_1 G^\star &= Q/G = z^3 + 2z \\ \gcd(G, G^\star) &= 1 \\ Q_1 &= z \\ Q_1 G^\star G'/G &= 2z^2. \end{aligned}$$

We get the Bezout coefficients $z^4 - 2z^2 + 4 = -3(2z^2) + (z^2+2)(z^2+2)$. Eventually:

$$f = \left(\frac{8z^2+4}{z^5+4z^3+4z} + \frac{3}{z^2+2}\right)' + \frac{1}{z}$$

2.3. The Hurwitz-Ostrogradsky method. It is possible to compute directly the result by the following remark. The aim is to compute

$$\frac{P}{Q} = \left(-\frac{A}{G}\right)' + \frac{C}{Q_1 \cdots Q_m}$$

i.e.

$$P = AQ_1 \frac{G^\star G'}{G} - A'Q_1 G^\star + CG$$

with the conditions: $\deg A < \deg G$, $\deg C < \deg(Q_1 G^\star)$. This is a linear system in the coefficients of A and C, which can be directly solved by a linear solver (again in the basic machinery of the toolkit).

2.4. The logarithmic part. Now it remains to solve the problem for a fraction $f = P/Q$, where Q is square-free. At this point two difficulties must be pointed out. Consider the example

$$f = \frac{5z^4 + 60z^3 + 255z^2 + 450z + 275}{z^5 + 15z^4 + 85z^3 + 225z^2 + 274z + 120}, \tag{3}$$

whose primitive is

$$\frac{25}{24}\log(z+1) + \frac{5}{6}\log(z+2) + \frac{5}{4}\log(z+3) + \frac{5}{6}\log(z+4) + \frac{25}{24}\log(z+5),$$

given in expanded form. If the factors where factored in simple logarithm, it would be the logarithm of a degree 120 polynomial, with coefficients of order 10^{68}. So the problem of the growth of intermediate numbers is present here.

The other difficulty is that algebraic numbers may be required in the solution, as shown by the example:

$$\int \frac{dz}{z^2-2} = \frac{\sqrt{2}}{4} \log \frac{z-\sqrt{2}}{z+\sqrt{2}}.$$

Then the problem is to compute in the extension of $\mathbb{Q}$ of lowest degree.

The method of Rothstein-Trager gives a good solution. Consider

$$f = \frac{P}{Q} = \sum_{i=1\ldots n} \frac{a_i}{z-\alpha_i}.$$

- One can check that $a_i = P(\alpha_i)/Q'(\alpha_i)$, thus the polynomials Q and $P - a_iQ'$ have a common root α_i. So a_i is a root of $R = \mathrm{Res}_z(Q, P - tQ')$ which belongs to $\mathbb{Q}[t]$. (Res_z stands for the *resultant in z*, another basic tool, which gives algebraic conditions on coefficients for two polynomials in z to have a common root.)
- Then α_i is a root of $G_i = \gcd(Q, P - a_iQ')$ which belongs to $\mathbb{Q}(a_i)[z]$. These roots can be collected for each a_i: for each such a_i, the answer is

$$a_i \sum_{\alpha_j | G_i(\alpha_j)=0} \log(z-\alpha_j) = a_i \log(G_i).$$

- The final answer is

$$\int \frac{P}{Q} = \sum_{a|R(a)=0} a \log(\gcd(Q, P - aQ')).$$

It must be pointed out that this is the expression of the primitive in the lowest degree extension of $\mathbb{Q}$.

Example. We consider the function f given by (3).We have $\mathrm{Res}_z(Q, P-tQ') = (5-4t)(6t-5)^2(24t-25)^2$, and

$$\int f = \frac{5}{4}\log(z+3) + \sum_{\alpha\in\{\frac{25}{24},\frac{5}{6}\}} \alpha \log(z^2+6z+20-72\frac{\alpha}{5}).$$

3. Elementary functions

3.1. Definition. The classical definition of elementary functions, after Liouville's work, is:

Definition 1. Let K be a differential field, y is said to be *elementary* over K if $K(y)$ and K have the same field of constants, and if

(1) either y is algebraic over K,
(2) or there exists $f \in K$ such that $f' = fy'$ (i.e. y is some kind of logarithm),
(3) or there exists $f \in K$ such that $y' = yf'$ (i.e. y is an exponential).

Now the following theorem gives an important characterization of functions which admit an elementary primitive:

Theorem 2 (Liouville). *Let K be a differential field, and C be the field of constants of K, $f \in K$ admits an elementary primitive over K if and only if f can be written*

$$f = v' + \sum_{i=1}^{m} c_i \frac{u_i'}{u_i}$$

where $v \in K$, $c_i \in \overline{C}$, $u_i \in K$.

The integration algorithm in the case of algebraic functions relies on a much more difficult theory, which borrows some tools from algebraic geometry. It is not within the scope of this talk to explain these. So we shall restrict ourselves to transcendental functions, which are amazingly simpler to deal with.

3.2. The strategy of Risch's algorithm; transcendental case. The Risch algorithm appears to be a slight generalisation of the method we have described here for the case of rational functions. We give here Bronstein's version of Risch's algorithm, in the transcendental case:

- Find a tower of extensions of $\mathbb{Q}$ given by $\theta_1 = z, \theta_2, \dots, \theta_k$ such that θ_i is elementary over the previous field $\mathbb{Q}(\theta_1, \dots, \theta_{i-1})$ for every i, and such that $f \in \mathbb{Q}(\theta_1, \dots, \theta_k)$. We note $\theta = \theta_k$.
- Write $f = A(\theta)/D(\theta)$, with $A, D \in \mathbb{Q}(\theta_1, \dots, \theta_{k-1})$.
- Split D into $D = D_s D_n$, where D_s is the product of the square-free factors F of D such that $\gcd(F, F') \neq 1$. We rewrite the function f as follows
$$f = P + \frac{B}{D_s} + \frac{C}{D_n} = f_p + f_s + f_n.$$
Here f_p is called the polynomial part, f_s is the special part, and f_n the normal part.
- A more general case of Hermite's reduction is applied, to eliminate all factors of the denominators whose highest multiplicity is greater than 1. This gives
$$f = f_p + g' + h$$
where the denominator of h has no multiple factors.
- A Rothstein-Trager resultant is computed to find the logarithmic part (or to decide it does not exists).
- The polynomial part f_p is integrated by an appropriate method, whether y is a logarithm or an exponential.

It is understood that it is a recursive method, since an effective method for the case $k = 1$ (i.e. a simple extension) gives a method for the general case, where computations are performed is some tower of extension, where arithmetic is effective, and primitives recursively computed.

3.3. More details. Now we give details about the main steps of Risch's algorithm. For constructing the extension tower, and finding the primitive of the special part, see the references.

3.3.1. The Hermite reduction We are concerned with the normal part A/D_n, which we would like to rewrite $A/D_n = g' + P + B/Q_n$, where Q_n has no multiple factors. As before we describe a single step of the method, for reducing the highest multiplicity at least by one.

Let V^{k+1} be the highest exponent in the square-free decomposition of D_n, that is $D_n = UV^{k+1}$, where V is square-free, $\gcd(U, V) = 1$ and the highest multiplicity of U is less than k.

We seek two polynomials $G, H \in K[\theta]$, such that $\deg G < \deg V$ and
$$\frac{A}{UV^{k+1}} = \left(\frac{G}{V^k}\right)' + \frac{H}{UV^k}.$$
(Remember that our goal is to lower the highest multiplicity of the denominator.) A small computation gives the relation $A = UVG' - kUV'G + VH$, and modulo V, we have
$$G = -\frac{A}{kUV'} \mod V.$$
Again we pick in our basic toolkit the extended Euclidean algorithm to compute $A/kUV' \bmod V$. H is easily computed from the data of A, U, V, G, k.

Example. Consider
$$f = \frac{z - \tan(z)}{\tan^2(z)} \quad \in \mathbb{Q}(z)(\theta),\ \theta' = 1 + \theta^2.$$
The following equation is to be solved for $G, H \in \mathbb{Q}(z)[\theta], \deg G < 1$:
$$\frac{z - \theta}{\theta^2} = \left(\frac{G}{\theta}\right)' + \frac{H}{\theta}.$$

This leads to $G = -z \mod \theta$, and $H = -\theta z$, thus $f = (-z/\theta)' - z$ and $\int f = -z/(\tan z) - z^2/2$.

3.3.2. The Rothstein-Trager resultant We consider $f = f_p + f_s + G/D$, where D is square-free. Again we consider

$$R(z) = \mathrm{Res}_\theta(G - zD', D).$$

The function f has an elementary primitive if and only if $R(z) = \beta P(z)$, where $\beta \in K$ and $P \in K[z]$ is a monic polynomial with constant coefficients. It may happen that the coefficients of $R(z)$ have a non-zero derivative, in which case f has no elementary primitive.

The logarithmic part of the primitive is

$$\int \frac{G}{D} = \sum_{\alpha | R(\alpha)=0} \alpha \log(\gcd(G - \alpha D', D)).$$

3.3.3. Polynomials parts

First we describe the method for a polynomial in $\log(u(z))$. Consider

$$f_p = a_m(z)\log^m u(z) + a_{m-1}(z)\log^{m-1} u(z) + \cdots + a_1(z)\log u(z) + a_0(z).$$

The Liouville principle implies that, if f_p admit a primitive, f_p can be written

$$f_p = \sum_{i=0}^{m} a_i(z)\log^i(z) = \left(\sum_{i=0}^{n} b_i \log^i u(z)\right)' + \sum_{i=1}^{k} \frac{c_i v_i'}{v_i}$$

Identification gives

$$\begin{aligned} 0 &= B'_{m+1}, \\ A_i &= B'_i + (i+1)B_{i+1}\frac{u'}{u}, \quad 1 \le i \ \le m. \end{aligned}$$

Thus each B'_i can be iteratively computed, and again each B'_i is integrated by a recursive application of the algorithm, and is found up to a constant. Furthermore, by identification, the constant of B_{i+1} is found.

Example.

$$F = \left(\frac{3}{2z} + \frac{1}{\log(z+\frac{1}{2})} - 2\frac{2z}{(2z+1)\log(z+\frac{1}{2})}\right)\log^2(z) + \frac{2\log z}{\log(z+\frac{1}{2})} + \frac{2}{(2z+1)\log(z+\frac{1}{2})}$$

We let $\tau = \log(z+1/2)$, and $\theta = \log z$, such that $F = A_2\theta^2 + A_1\theta + A_0$, $A_i \in \mathbb{Q}(z,\tau)$. A primitive has the form $B_2\theta^3 + B_2\theta^2 + B_1\theta + B_0 + \sum c_i v' i/v_i$ and:

(1) $B'_3 = 0$ thus $B_3 = b_3 \in \overline{\mathbb{Q}}$.
(2) $A_2 = B'_2 + 3B_3\theta'$. Recursively we get $\int A_2 = 3\theta/2 + z/\tau$, this implies $b_3 = 1/2$ and $B_2 = z/\tau + b_2$.
(3) $A_1 - 2/\tau = B'_1 + 2b_2\theta'$. Again we get $\int A_1 - 2/\tau = 0$ and $b_2 = 0$, $B_1 = b_1$.
(4) $A_0 = B'_0 + b_1\theta' + \sum c_i v'_i/v_i$. $\int A_0 = \log\log(z+1/2)$, thus $b_1 = 0$, B_0 is a constant, and $c_1 = 1$, $v_1 = \tau$.

In the end, $\int F = \log(z)^3/2 + z\log^2(z)/\log(z+1/2) + \log\log(z+1/2)$.

We explain the algorithm for a polynomials in exponential terms: let f_p be

$$f_p = a_m(z)\exp(mu(z)) + a_{m-1}(z)\exp((m-1)u(z)) + \cdots + a_{-p}(Z)\exp[-pu(z)].$$

The Liouville principle implies that the primitive is of the form $\sum_{i=-p}^{m} b_i(z)\exp(iu(z))$, with

$$b'_i + iu'b_i = a_i$$

which is called the Risch differential equation. This equation can be solved without difficulty since only rational solutions are required.

Example. $\int e^{-z^2}$. The Risch differential equation is

$$b'(z) - 2zb(z) = 1$$

for which a rational solution is to be found. Because there is no pole in the coefficients of the equation and the leading coefficient is constant, $b(z)$ has no pole, and so must be a polynomial. But degree constraints show this equation has no polynomial solution. Conclusion: $\int e^{-z^2}$ is not elementary.

Bibliography

[1] Bronstein (Manuel). – Symbolic integration, 1993. Book in preparation.
[2] Davenport (J. H.), Siret (Y.), and Tournier (E.). – *Calcul formel.* – Paris, Masson, 1986.
[3] Geddes (Keith O.), Czapor (Stephen R.), and Labahn (George). – *Algorithms for computer algebra.* – Kluwer, 1992.